LoadRunner

Analysis User's Guide

Version 7.8

MERCURY INTERACTIVE

LoadRunner Analysis User's Guide, Version 7.8

Mercury Interactive Corporation
1325 Borregas Avenue
Sunnyvale, CA 94089 USA
Tel: (408) 822-5200
Toll Free: (800) TEST-911, (866) TOPAZ-4U
Fax: (408) 822-5300

If you have any comments or suggestions regarding this document, please send them via e-mail to documentation@merc-int.com.

LRANUG7.8/01

Table of Contents

Welcome to LoadRunner

Welcome to LoadRunner, Mercury Interactive's tool for testing the performance of applications. LoadRunner stresses your entire application to isolate and identify potential client, network, and server bottlenecks.

LoadRunner enables you to test your system under controlled and peak load conditions. To generate load, LoadRunner runs thousands of Virtual Users that are distributed over a network. Using a minimum of hardware resources, these Virtual Users provide consistent, repeatable, and measurable load to exercise your application just as real users would. LoadRunner's in-depth reports and graphs provide the information that you need to evaluate the performance of your application.

Online Resources

LoadRunner includes the following online tools:

Read Me First provides last-minute news and information about LoadRunner.

Books Online displays the complete documentation set in PDF format. Online books can be read and printed using Adobe Acrobat Reader, which is included in the installation package. Check Mercury Interactive's Customer Support Web site for updates to LoadRunner online books.

LoadRunner Function Reference gives you online access to all of LoadRunner's functions that you can use when creating Vuser scripts, including examples of how to use the functions. Check Mercury Interactive's Customer Support Web site for updates to the online *LoadRunner Function Reference*.

LoadRunner Context Sensitive Help provides immediate answers to questions that arise as you work with LoadRunner. It describes dialog boxes, and shows you how to perform LoadRunner tasks. To activate this help, click in a window and press F1. Check Mercury Interactive's Customer Support Web site for updates to LoadRunner help files.

Technical Support Online uses your default Web browser to open Mercury Interactive's Customer Support Web site. This site enables you to browse the knowledge base and add your own articles, post to and search user discussion forums, submit support requests, download patches and updated documentation, and more. The URL for this Web site is http://support.mercuryinteractive.com.

Support Information presents the locations of Mercury Interactive's Customer Support Web site and home page, the e-mail address for sending information requests, and a list of Mercury Interactive's offices around the world.

Mercury Interactive on the Web uses your default Web browser to open Mercury Interactive's home page (http://www.mercuryinteractive.com). This site enables you to browse the knowledge base and add your own articles, post to and search user discussion forums, submit support requests, download patches and updated documentation, and more.

LoadRunner Documentation Set

LoadRunner is supplied with a set of documentation that describes how to:

➤ install LoadRunner

➤ create Vuser scripts

➤ use the LoadRunner Controller

➤ use the LoadRunner Analysis

Using the LoadRunner Documentation Set

The LoadRunner documentation set consists of one installation guide, a Controller user's guide, an Analysis user's guide, and two guides for creating Virtual User scripts.

Installation Guide

For instructions on installing LoadRunner Analysis 7.8, refer to the *Installing LoadRunner Analysis Guide.*

Controller User's Guide

The LoadRunner documentation pack includes one Controller user's guide:

The *LoadRunner Controller User's Guide* describes how to create and run LoadRunner scenarios using the LoadRunner Controller in a Windows environment. The Vusers can run on UNIX and Windows-based platforms. The Controller user's guide presents an overview of the LoadRunner testing process.

Analysis User's Guide

The LoadRunner documentation pack includes one Analysis user's guide:

The *LoadRunner Analysis User's Guide* describes how to use the LoadRunner Analysis graphs and reports after running a scenario in order to analyze system performance.

Guides for Creating Vuser Scripts

The LoadRunner documentation pack has two guides that describe how to create Vuser scripts:

➤ The *Creating Vuser Scripts* guide describes how to create all types of Vuser scripts. When necessary, supplement this document with the online *LoadRunner Function Reference* and the following guide.

➤ The *WinRunner User's Guide* describes in detail how to use WinRunner to create GUI Vuser scripts. The resulting Vuser scripts run on Windows platforms. The *TSL Online Reference* should be used in conjunction with this document.

For information on	Look here...
Installing LoadRunner	*Installing LoadRunner* guide
The LoadRunner testing process	*LoadRunner Controller User's Guide*
Creating Vuser scripts	*Creating Vuser Scripts* guide
Creating and running scenarios	*LoadRunner Controller User's Guide*
Analyzing test results	*LoadRunner Analysis User's Guide*

Documentation Updates

Mercury Interactive is continuously updating its product documentation with new information. You can download the latest version of this document from Mercury Interactive's Customer Support Web site (http://support.mercuryinteractive.com).

To download updated documentation:

1 In the Customer Support Web site, click the **Documentation** link.

2 Select the product name. Note that if <product name> does not appear in the list, you must add it to your customer profile. Click "My Account" to update your profile.

3 Click **Retrieve**. The Documentation page opens and lists all the documentation available for the current release and for previous releases. If a document was recently updated, **Updated** appears next to the document name.

4 Click a document link to download the documentation.

Typographical Conventions

This book uses the following typographical conventions:

1, 2, 3	Bold numbers indicate steps in a procedure.
➤	Bullets indicate options and features.
>	The greater than sign separates menu levels (for example, **File** > **Open**).
Stone Sans	The **Stone Sans** font indicates names of interface elements on which you perform actions (for example, "Click the **Run** button.").
Bold	**Bold** text indicates method or function names
Italics	*Italic* text indicates method or function arguments, file names or paths, and book titles.
Arial	The Arial font is used for examples and text that is to be typed literally.
<>	Angle brackets enclose a part of a file path or URL address that may vary from user to user (for example, *<Product installation folder>\bin*).
[ ]	Square brackets enclose optional arguments.
{ }	Curly brackets indicate that one of the enclosed values must be assigned to the current argument.
...	In a line of syntax, an ellipsis indicates that more items of the same format may be included.

1

Introducing Analysis

LoadRunner Analysis provides graphs and reports to help you analyze the performance of your system. These graphs and reports summarize the scenario execution.

This chapter describes:

➤ Analysis Basics

➤ Setting Data Options

➤ Setting General Options

➤ Setting Database Options

➤ Setting Web Page Breakdown Options

➤ Using Templates

➤ Viewing Session Information

➤ Viewing the Scenario Run Time Settings

➤ Analysis Graphs

➤ Opening Analysis Graphs

About Analysis

During scenario execution, Vusers generate result data as they perform their transactions. To monitor the scenario performance *during* test execution, use the online monitoring tools described in the *LoadRunner Controller User's Guide*. To view a summary of the results *after* test execution, you can use one or more of the following tools:

➤ The **Vuser log files** contain a full trace of the scenario run for each Vuser. These files are located in the scenario results directory. (When you run a Vuser script in standalone mode, these files are placed in the Vuser script directory.) For more information on Vuser log files, refer to the *Creating Vuser Scripts* guide.

➤ The **Controller Output window** displays information about the scenario run. If your scenario run fails, look for debug information in this window. For more information, see the *LoadRunner Controller User's Guide*.

➤ The **Analysis graphs** help you determine system performance and provide information about transactions and Vusers. You can compare multiple graphs by combining results from several scenarios or merging several graphs into one.

➤ The **Graph Data** and **Raw Data** views display the actual data used to generate the graph in a spreadsheet format. You can copy this data into external spreadsheet applications for further processing.

➤ The **Report** utilities enable you to view a Summary HTML report for each graph or a variety of Performance and Activity reports. You can create a report as a Microsoft Word document, which automatically summarizes and displays the test's significant data in graphical and tabular format.

This chapter provides an overview of the graphs and reports that can be generated through Analysis.

Analysis Basics

This section describes basic concepts that will enhance your understanding of how to work with Analysis.

Creating Analysis Sessions

When you run a scenario, data is stored in a result file with an *.lrr* extension. Analysis is the utility that processes the gathered result information and generates graphs and reports.

When you work with the Analysis utility, you work within a *session*. An Analysis session contains at least one set of scenario results (*lrr* file). Analysis stores the display information and layout settings for the active graphs in a file with an *.lra* extension.

Starting Analysis

You can open Analysis through as an independent application or directly from the Controller. To open Analysis as an independent application, choose **Analysis** from the LoadRunner Program Group.

To open Analysis directly from the Controller, select **Results** > **Analyze Results**. This option is only available after running a scenario. Analysis takes the latest result file from the current scenario, and opens a new session using these results. You can also instruct the Controller to automatically open Analysis after it completes scenario execution by selecting **Results** > **Auto Load Analysis**.

When creating a new session, Analysis prompts you for the scenario result file (*.lrr* extension) to include in the session. To open an existing Analysis session, you specify an Analysis Session file (*.lra* extension).

Collating Execution Results

When you run a scenario, by default all Vuser information is stored locally on each Vuser host. After scenario execution the results are automatically *collated* or consolidated—results from all of the hosts are transferred to the results directory. You disable automatic collation by choosing **Results** > **Auto Collate Results** from the Controller window, to clear the check mark adjacent to the option. To manually collate results, choose **Results** > **Collate Results**. If your results have not been collated, Analysis will automatically collate the results before generating the analysis data. For more information about collating results, see the *LoadRunner Controller User's Guide*.

Viewing Summary Data

In large scenarios, with results exceeding 100 MBs, it can take a long time for Analysis to process the data. While LoadRunner is processing the complete data, you can view a summary of the data.

To view the summary data, choose **Tools** > **Options**, and select the **Result Collection** tab. Select **Display summary data while generating complete data** if you want Analysis to process the complete data graphs while you view the summary data, or select **Generate summary data only** if you do not want LoadRunner to process the complete Analysis data.

The following graphs are not available when viewing summary data only:

➤ Rendezvous

➤ Data Point (Sum)

➤ Web Page Breakdown

➤ Network Monitor

➤ Error

Note: Some fields are not available for filtering when you work with summary graphs.

Setting Data Options

You can configure Analysis to generate and display summary or complete data. When you choose to generate the complete Analysis data, Analysis aggregates the data. Aggregation reduces the size of the database and decreases processing time in large scenarios.

You can also configure Analysis to store and display data for the complete duration of the scenario, or for a specified time range only. This decreases the size of the database and thereby decreases processing time.

You use the Result Collection tab of the Options dialog box to configure data options.

To configure data aggregation:

1 Select **Tools** > **Options**, and select the **Result Collection** tab.
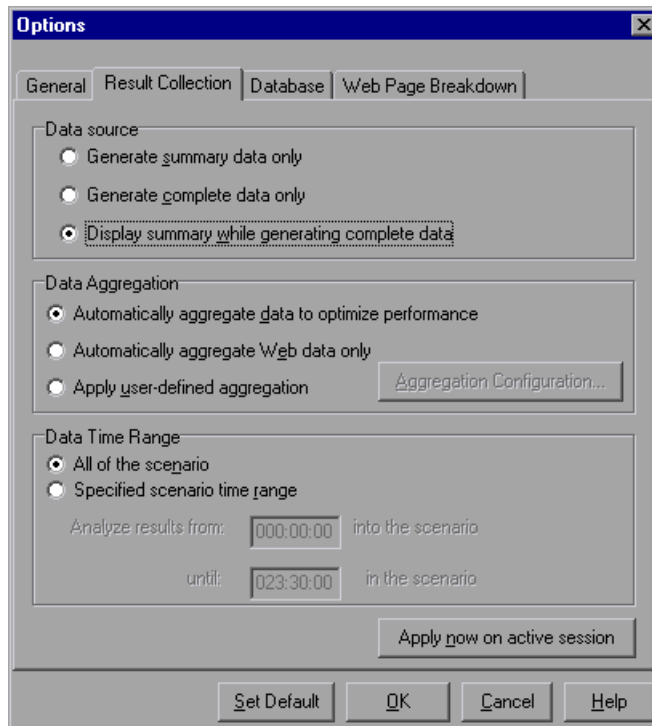
2 Select data source, data aggregation, and data time range options, as described in "Understanding the Options Dialog Box—Result Collection Tab," on page 6.

3 To configure custom aggregation, click **Aggregation Configuration** and set the options as described in "Understanding the Data Aggregation Configuration Dialog Box," on page 9.

Note: All graphs except the Running Vusers graph will be affected by the time range settings, in both the Summary and Complete Data Analysis modes.

---

4 Click **OK**.

To apply the changes to the active session, click **Apply now on active session**.

Understanding the Options Dialog Box—Result Collection Tab

In large scenarios, with results exceeding 100 MB, it will take several minutes for Analysis to process the data. The Result Collection tab of the Options dialog box enables you to indicate to LoadRunner to display a summary of the data, while you wait for the complete data to be processed.

The *complete* data refers to the result data after it has been processed for use within Analysis tool. The graphs can be sorted, filtered, and manipulated. The *summary* data refers to the raw, unprocessed data. The summary graphs contain general information such as transaction names and times, and not all filtering options are available.

If you choose to generate the complete data, Analysis aggregates the data being generated using either built-in data aggregation formulas, or aggregation settings that you define. Data aggregation is necessary in order to reduce the size of the database and decrease processing time in large scenarios.

You can also instruct Analysis to display data for the complete duration of the scenario, or for a specified time range only.

Data Source

> **Generate summary data only:** View the summary data only. If this option is selected, Analysis will not process the data for advanced use with filtration and grouping.

> **Generate complete data only:** View only the complete data after it has been processed. Do not display the summary data.

> **Display summary data while generating complete data:** View summary data while the complete data is being processed. After the processing, view the complete data. A bar below the graph indicates the complete data generation progress.

Data Aggregation

> **Automatically aggregate data to optimize performance:** Aggregates data using built-in data aggregation formulas.

> **Automatically aggregate Web data only:** Aggregates Web data only using built-in data aggregation formulas.

> **Apply user-defined aggregation:** Aggregates data using settings you define. For more information on user-defined aggregation settings, see "Configuring User-Defined Data Aggregation," on page 8.

> **Aggregation Configuration:** Opens the Data Aggregation Configuration dialog box, in which you define custom aggregation settings.

Data Time Range

> **All of the scenario:** Displays data for the complete duration of the scenario.

> **Specified scenario time range:** Displays data only for the specified time range of the scenario.

> **Analyze results from X into the scenario:** Enter the amount of scenario time you want to elapse (in hhh:mm:ss format) before Analysis begins displaying data.

> **until X in the scenario:** Enter the point in the scenario (in hhh:mm:ss format) at which you want Analysis to stop displaying data.

Apply now to active session: Applies the settings in the Result Collection tab to the current session.

Configuring User-Defined Data Aggregation

To configure user-defined aggregation settings:

1 Select **Tools** > **Options**, and select the **Result Collection** tab.

2 Click **Aggregation Configuration**. The Data Aggregation Configuration dialog box opens.
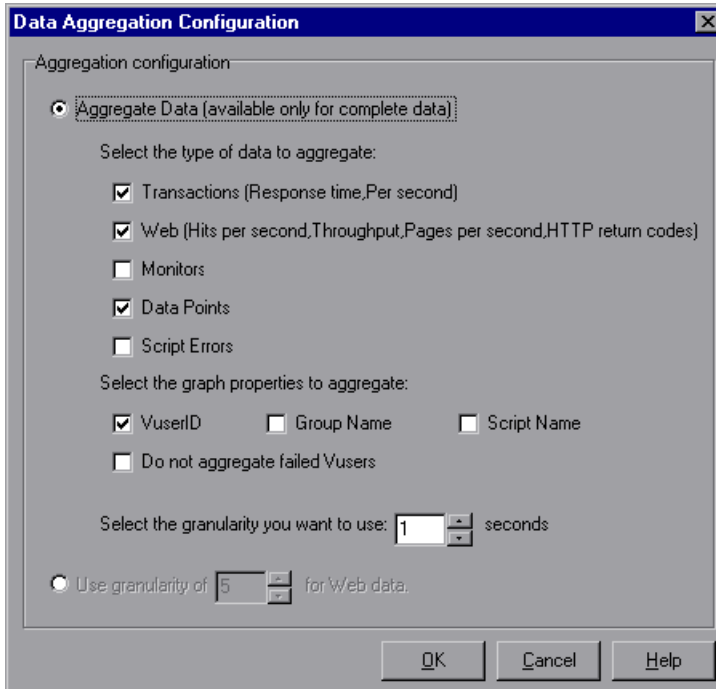
3 Customize the data aggregation and granularity settings as described in "Understanding the Data Aggregation Configuration Dialog Box," on page 9.

Note: You will not be able to drill down on the graph properties you select to aggregate.

---

4 Click **OK**.

Understanding the Data Aggregation Configuration Dialog Box

You can use the Data Aggregation Configuration dialog box to define custom aggregation and granularity settings.

Aggregate Data: Specify the data you want to aggregate in order to reduce the size of the database.

> **Select the type of data to aggregate:** Specify the type(s) of graphs for which you want to aggregate data.

> **Select the graph properties to aggregate:** Specify the graph properties—Vuser ID, Group Name, and Script Name—you want to aggregate. If you do not want to aggregate the failed Vuser data, select **Do not aggregate failed Vusers**.

> **Select the granularity you want to use: X seconds:** Specify a custom granularity for the data. To reduce the size of the database, increase the granularity. To focus on more detailed results, decrease the granularity. Note that the minimum granularity is 1 second.

Use granularity of X seconds for Web data: Specify a custom granularity for Web data. By default, Analysis summarizes Web measurements every 5 seconds. To reduce the size of the database, increase the granularity. To focus on more detailed results, decrease the granularity.

Setting General Options

You can configure the following general options:

➤ Date storage and display format

➤ File browser directory locations

➤ Temporary file locations

➤ Summary report transaction reporting

You use the General tab of the Options dialog box to set general options.

To configure General options:

1 Select **Tools > Options**. The Options dialog box opens displaying the General tab.
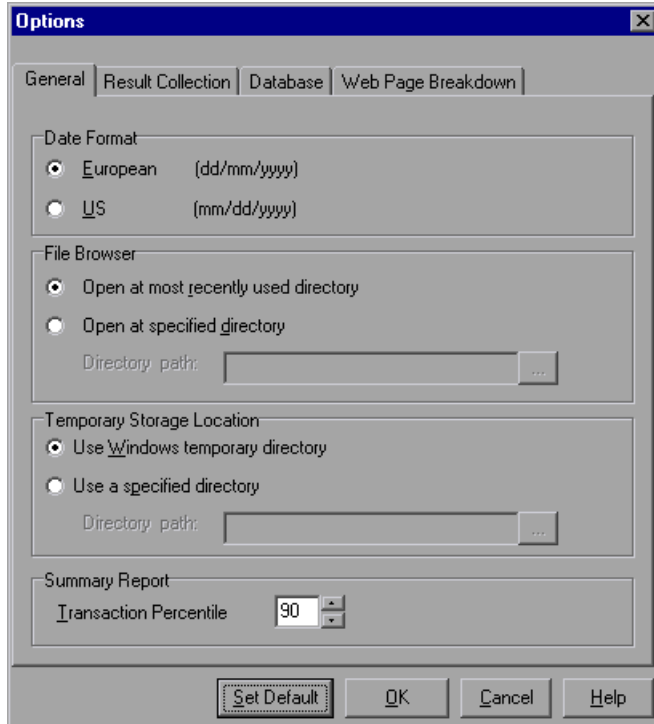
2 Set the general options as described in "Understanding the Options Dialog Box—General Tab," on page 11.

3 Click **OK**.

Understanding the Options Dialog Box—General Tab

You can use the General tab of the Options dialog box to set the following options:

Date Format: Select a date format for storage and display.

> **European:** Displays the European date format.

> **US:** Displays the U.S. date format.

File Browser: Select the directory location at which you want the file browser to open.

> **Open at most recently used directory:** Opens the file browser at the previously used directory location.

> **Open at specified directory:** Opens the file browser at a specified directory.

> **Directory path:** Enter the directory location at which you want the file browser to open.

Temporary Storage Location: Select the directory location in which you want to save temporary files.

> **Use Windows temporary directory:** Saves temporary files in your Windows temp directory.

> **Use a specified directory:** Saves temporary files in a specified directory.

> **Directory path:** Enter the directory location in which you want to save temporary files.

Summary Report: Set the percentage of transactions whose response times are displayed in the Summary Report.

> **Transaction Percentile**: The Summary Report contains a percentile column showing the response time of 90% of transactions (90% of transactions fall within this amount of time). To change the value of the default 90% percentile, enter a new figure in the **Transaction Percentile** box. Since this is an application level setting, the column name changes to the new percentile figure (for example, to "80% Percentile") only on the next invocation of Analysis.

Setting Database Options

You can choose the database in which to store Analysis session result data and you can repair and compress your Analysis results and optimize the database that may have become fragmented.

By default, LoadRunner stores Analysis result data in an Access 2000 database. If your Analysis result data exceeds two gigabytes, it is recommended that you store it on an SQL server or MSDE machine.

Configuring Database Format Options

1 Select **Tools** > **Options**, and select the **Database** tab.
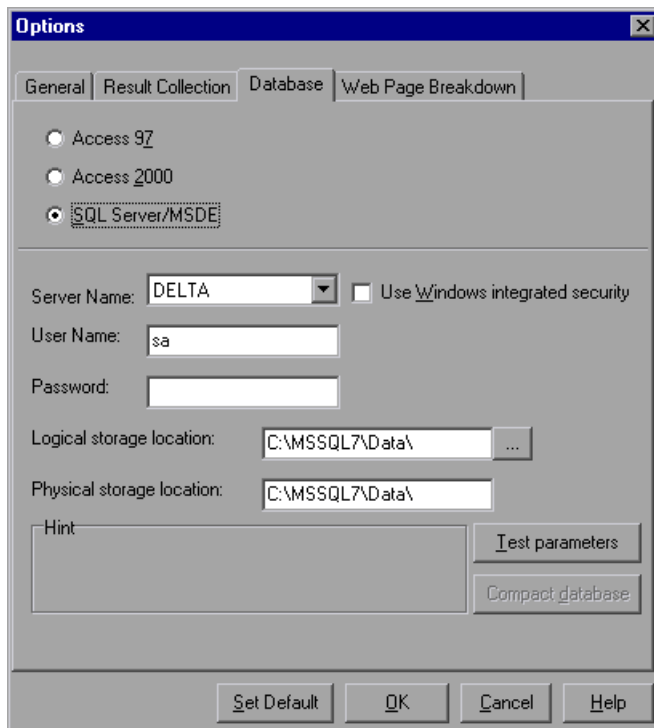
2 Choose database format options, as described in "Understanding the Options Dialog Box—Database Tab," on page 13.

3 To verify that the "list separator" registry options on your machine are the same as those on the database machine, click **Test parameters**.

4 To repair and compress results that may have become fragmented, and to prevent the use of excessive disk space, click **Compact database**.

Note: Long scenarios (duration of two hours or more) will require more time for compacting.

Understanding the Options Dialog Box—Database Tab

The Database tab of the Options dialog box enables you to specify the database in which to store Analysis session result data. If your Analysis result data exceeds two gigabytes, it is recommended that you store it on an SQL server or MSDE machine.

➤ **Access 97:** Saves Analysis result data in an Access 97 database format. This setting is the default.

➤ **Access 2000:** Saves Analysis result data in an Access 2000 database format.

➤ **SQL Server/MSDE:** Instructs LoadRunner to save Analysis result data on an SQL Server or MSDE machine.

Note: You can access the MSDE installation from the **add-ins** directory on the product installation CD.

Server Name: Select or enter the name of the machine on which the SQL server or MSDE is running.

Use Windows-integrated security: Enables you to use your Windows login, instead of specifying a user name and password. By default, the user name "sa" and no password are used for the SQL server.

User Name: Enter the user name for the master database.

Password: Enter the password for the master database.

Logical storage location: Enter a shared directory on the SQL server/MSDE machine in which you want permanent and temporary database files to be stored. For example, if your SQL server's name is fly, enter \\fly\<Analysis Database>\.

Analysis results stored on an SQL server/MSDE machine can only be viewed on the machine's local LAN.

Physical storage location: Enter the real drive and directory path on the SQL server/MSDE machine that correspond to the logical storage location. For example, if the Analysis database is mapped to an SQL server named *fly*, and *fly* is mapped to drive D, enter D:\<Analysis Database>.

If the SQL server/MSDE and Analysis are on the same machine, the logical storage location and physical storage location are identical.

Test parameters (for Access)**:** Lets you connect to the Access database and verify that the "list separator" registry options on your machine are the same as those on the database machine.

Test parameters (for SQL Server/MSDE)**:** Lets you connect to the SQL server/MSDE machine and see that the shared directory you specified exists on the server, and that you have write permissions on the shared server directory. If so, Analysis synchronizes the shared and physical server directories.

Compact Database: When you configure and set up your Analysis session, the database containing the results may become fragmented. As a result, it will use excessive disk space. The Compact Database button enables you to repair and compress your results and optimize your Access database.

Note: If you store Analysis result data on an SQL server/MSDE machine, you must select **File > Save As** in order to save your Analysis session. To delete your Analysis session, you must select **File > Delete Current Session**.

To open a session stored on an SQL Server/MSDE machine, the machine must be running and the directory you defined must exist as a shared directory.

Setting Web Page Breakdown Options

You can choose how to aggregate the display of URLs that include dynamic information, such as a session ID. You can display these URLs individually, or you can unify them and display them as one line with merged data points.

You use the Web Page Breakdown tab of the Options dialog box to set Web page breakdown options.

To set the display of URLs with dynamic data:

1 Select **Tools > Options**, and select the **Web Page Breakdown** tab.
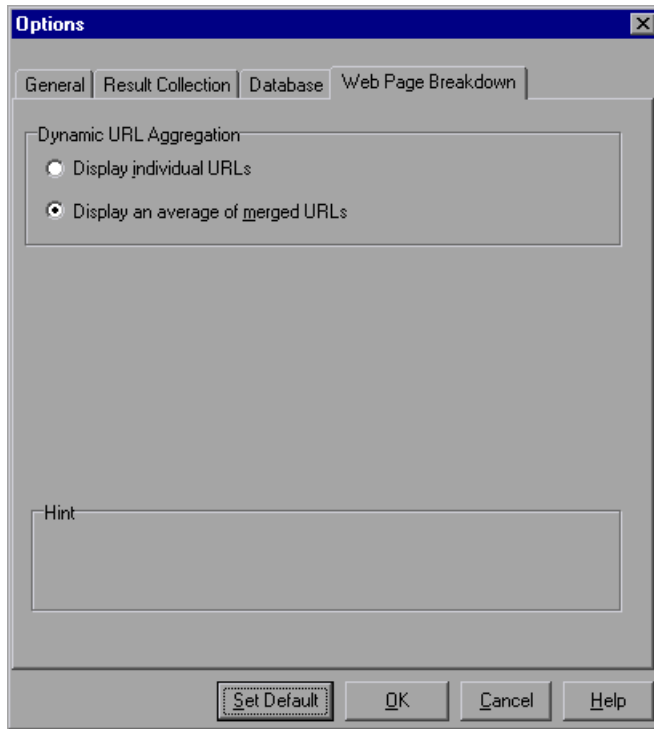
2 Select one of the following URL aggregation options:

➤ **Display individual URLs:** Displays each URL individually.

➤ **Display an average of merged URLs:** Merges URLs from the same script step into one URL, and displays it with merged (average) data points.

3 Click **OK**.

Using Templates

Templates enable you to save the current filter and display options and use them in another session.

You save, apply, and edit templates using the Template dialog box. Its title changes depending on the action being performed.

To use templates:

 1 Choose one of the following:

➤ **Tools** > **Templates** > **Save As Template.**

➤ **Tools** > **Templates** > **Apply/Edit Template.**

The Template dialog box opens.
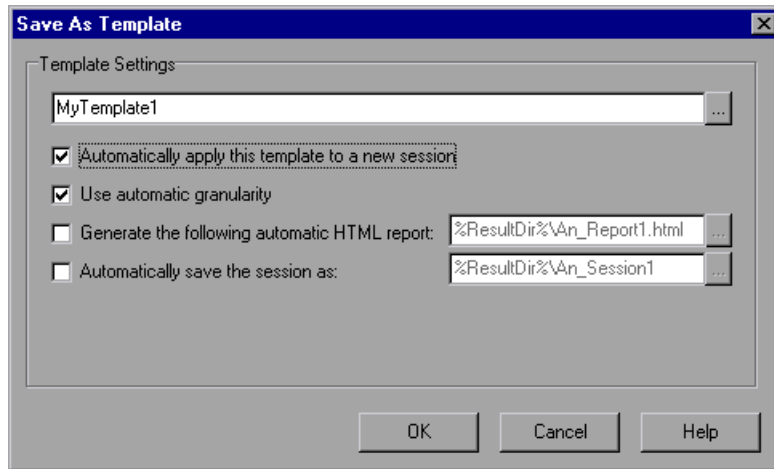
 2 Enter template options, as described in "Understanding the Template Dialog Box," on page 18.

 3 Click **OK**.

Understanding the Template Dialog Box

The Template dialog box enables you to perform the following:

➤ Save filter and display options for use in a another Analysis session.

➤ Apply a saved template to the current Analysis session.

➤ Edit the settings of a saved template.

Template Settings: Enter the name of the template you want to create, use, or edit, or click **Browse** to select a template .

Automatically apply this template to a new session: Applies the template to any new session you open.

Use automatic granularity: Applies the default Analysis granularity (one second) to the template. For information about setting Analysis granularity, see "Changing the Granularity of the Data" on page 55.

Generate the following automatic HTML report: Generates an HTML report using the template. Specify or select a report name. For information about generating HTML reports, see "Creating HTML Reports" on page 338.

Automatically save the session as: Automatically saves the session using the template you specify. Specify or select a file name.

Viewing Session Information

You can view the properties of the current Analysis session in the Session Information dialog box.

Choose **File** > **Session Information**. The Session Information dialog box opens
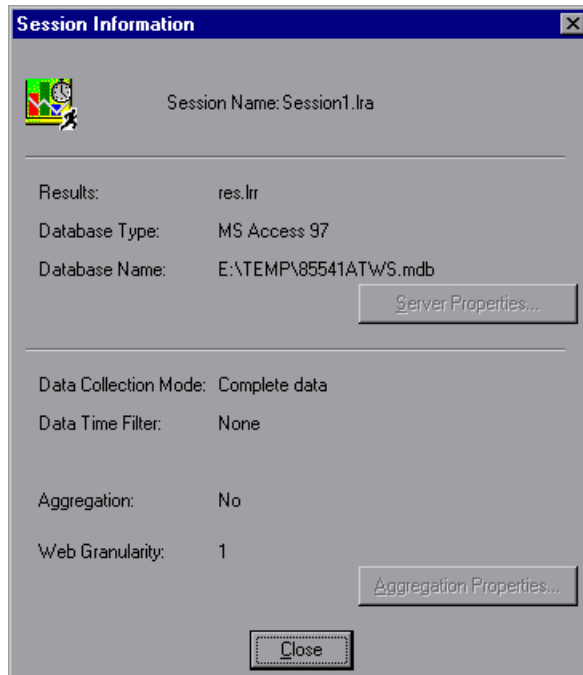
Understanding the Session Information Dialog Box

The Session Information dialog box enables you to view the properties of the current Analysis session.

Session Name: Displays the name of the current session.

Results: Displays the name of the LoadRunner result file.

Database Type: Displays the type of database used to store the scenario data.

Database Name: Displays the name and directory path of the database.

Server Properties: Displays the properties of the SQL Server and MSDE databases.

Data Collection Mode: Indicates whether the session displays complete data or summary data.

Data Time Filter: Indicates whether a time filter has been applied to the session.

Aggregation: Indicates whether the session data has been aggregated.

Web Granularity: Displays the Web granularity used in the session.

Aggregation Properties: Displays the type of data aggregated, the criteria according to which it is aggregated, and the time granularity of the aggregated data.

Viewing the Scenario Run Time Settings

You can view information about the Vuser groups and scripts that were run in each scenario, as well as the run time settings for each script in a scenario, in the Scenario Run Time Settings dialog box.

Note: The run time settings allow you to customize the way a Vuser script is executed. You configure the run time settings from the Controller or VuGen before running a scenario. For information on configuring the run time settings, refer to the *Creating Vuser Scripts* guide.

Choose **File** > **Scenario Run Time Settings**, or click **Run Time Settings** on the toolbar.

The Scenario Run Time Settings dialog box opens, displaying the Vuser groups, scripts, and scheduling information for each scenario. For each script in a scenario, you can view the run time settings that were configured in the Controller or VuGen before scenario execution.
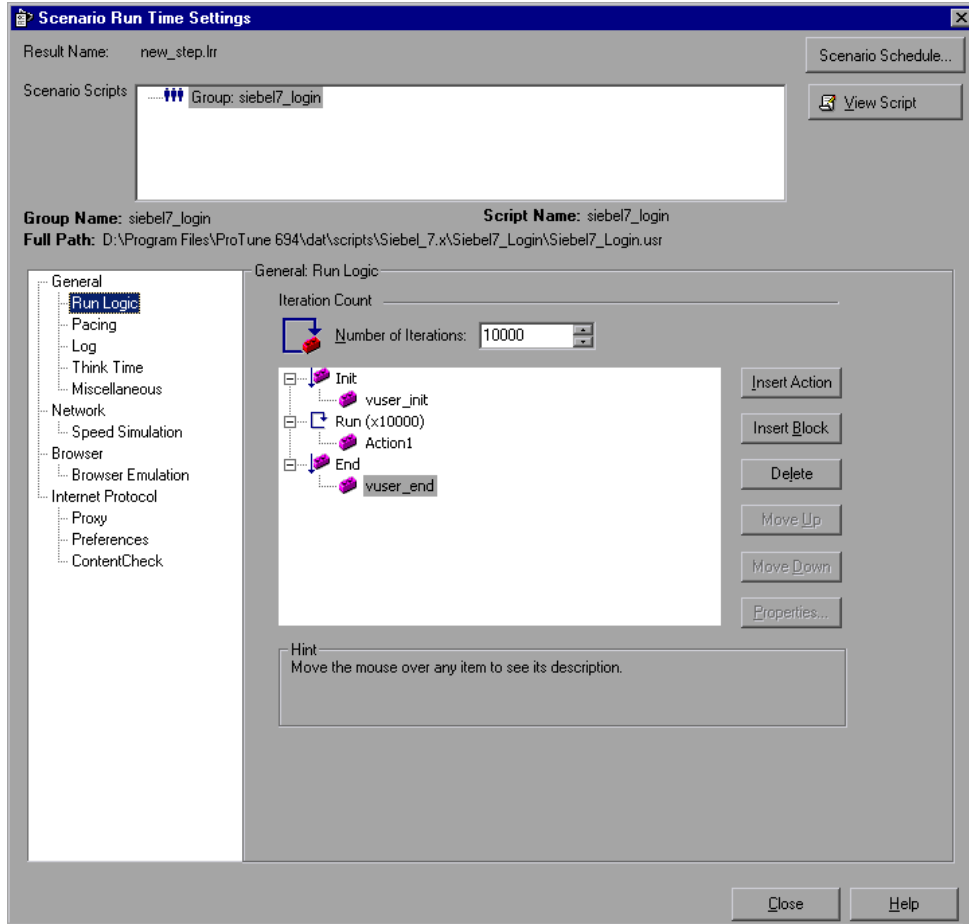
Understanding the Scenario Run Time Settings Dialog Box

The Scenario Run Time Settings dialog box enables you to view information about executed scenarios, as well as the run time settings for each script in a scenario.

Result Name: The name of the result file.

Scenario Scripts: Displays the result set for each executed scenario, as well as the Vuser groups and scripts that were run in the scenario.

Group Name: Displays the name of the group to which the selected script belongs.

Full Path: Displays the script's full directory path.

Script Name: Displays the name of the selected script.

Scenario Schedule: Displays goal-oriented or manual scenario scheduling information for the selected scenario.

View Script: Opens the Virtual User Generator, so that you can edit the script. For more information on editing scripts, see the *Creating Vuser Scripts* guide.

The options in the lower half of the screen are described in full in the *Creating Vuser Scripts* guide.

Analysis Graphs

Analysis graphs are divided into the following categories:

➤ Vuser Graphs
Provide information about Vuser states and other Vuser statistics. For more information, see Chapter 3, "Vuser Graphs."

➤ Error Graphs
Provide information about the errors that occurred during the scenario step. For more information, see Chapter 4, "Error Graphs."

➤ Transaction Graphs
Provide information about transaction performance and response time. For more information, see Chapter 5, "Transaction Graphs."

➤ Web Resource Graphs
Provide information about the throughput, hits per second, HTTP responses per second, number of retries per second, and downloaded pages per second for Web Vusers. For more information, see Chapter 6, "Web Resource Graphs."

➤ Web Page Breakdown Graphs
Provide information about the size and download time of each Web page component. For more information, see Chapter 7, "Web Page Breakdown Graphs."

➤ User-Defined Data Point Graphs
Provide information about the custom data points that were gathered by the online monitor. For more information, see Chapter 8, "User-Defined Data Point Graphs."

➤ System Resource Graphs
Provide statistics relating to the system resources that were monitored during the scenario using the online monitor. This category also includes graphs for SNMP monitoring. For more information, see Chapter 9, "System Resource Graphs."

➤ Network Monitor Graphs
Provide information about the network delays. For more information, see Chapter 10, "Network Monitor Graphs."

➤ Firewall Server Monitor Graphs
   Provide information about firewall server resource usage. For more
   information, see Chapter 11, "Firewall Server Monitor Graphs."

➤ Web Server Resource Graphs
   Provide information about the resource usage for the Apache,
   iPlanet/Netscape, iPlanet(SNMP), and MS IIS Web servers. For more
   information see Chapter 12, "Web Server Resource Graphs."

➤ Web Application Server Resource Graphs
   Provide information about the resource usage for various Web
   application servers. For more information see Chapter 13, "Web
   Application Server Resource Graphs."

➤ Database Server Resource Graphs
   Provide information about database resources. For more information, see
   Chapter 14, "Database Server Resource Graphs."

➤ Streaming Media Graphs
   Provide information about resource usage of streaming media. For more
   information, see Chapter 15, "Streaming Media Graphs."

➤ ERP/CRM Server Resource Graphs
   Provide information about ERP/CRM server resource usage. For more
   information, see Chapter 16, "ERP/CRM Server Resource Graphs."

➤ Java Performance Graphs
   Provide information about resource usage of Java-based applications. For
   more information, see Chapter 17, "Java Performance Graphs."

➤ Application Deployment Solutions Graphs
   Provide information about resource usage of the Citrix MetaFrame and
   1.8 servers. For more information, see Chapter 18, "Application
   Deployment Solutions Graphs."

➤ Middleware Performance Graphs
   Provide information about resource usage of the Tuxedo and IBM
   WebSphere MQ servers. For more information, see Chapter 19,
   "Middleware Performance Graphs."

Opening Analysis Graphs

By default, LoadRunner displays only the Summary Report in the graph tree view. You can add graphs to the graph tree view using the Open a New Graph dialog box.

To open a new graph:

1 Select **Graph** > **Add Graph**, or click **<New Graph>** in the graph tree view. The Open a New Graph dialog box opens.
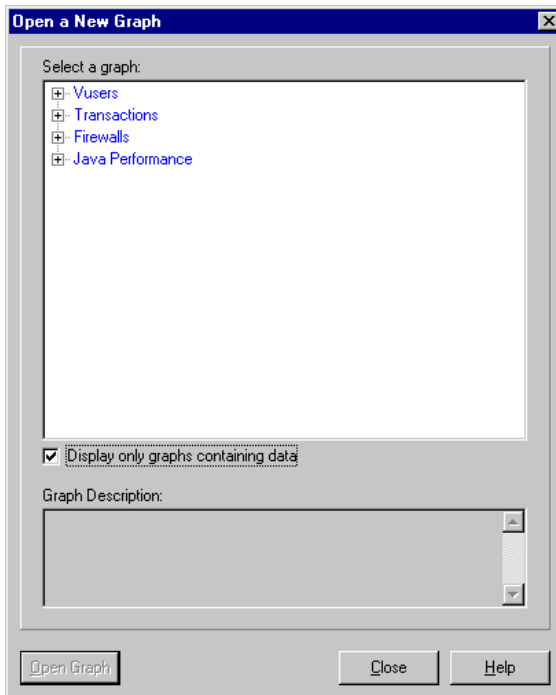
By default, only graphs which contain data are listed. To view the entire list of LoadRunner Analysis graphs, clear **Display only graphs containing data**.

2 Expand the graph tree and select a graph. You can view a description of the graph in the Graph Description box.

3 Click **Open Graph**. Analysis generates the selected graph and adds it to the graph tree view. The graph is displayed in the right pane of Analysis.

To display an existing graph in the right pane of Analysis, select the graph in the graph tree view.

Understanding the Open a New Graph Dialog Box

The Open a New Graph dialog box enables you to open a new graph and view its description.

Select a graph: Click the "+" to the left of each category to expand the tree view. Select a graph.

By default, only graphs which contain data are listed.

Note: You can only open one graph at a time.

Display only graphs containing data: By default, only graphs which contain data are listed. To view the entire list of LoadRunner Analysis graphs, clear this option.

Graph description: Displays the selected graph's description.

Open Graph: Opens the selected graph and displays it in the Graph Tree View.

2

Working with Analysis Graphs

Analysis contains several utilities that enable you to manage graph data to most effectively view the displayed data.

This chapter describes:

➤ Configuring the Graph Display

➤ Configuring the Data

➤ Analyzing Graph Results

➤ Printing Graphs

About Working with Analysis Graphs

Analysis provides a number of utilities that enable you to customize the graphs in your session so that you can view the data displayed in the most effective way possible.

You can configure display options and data settings, and manipulate the graphs for your use. For example, you can enlarge graph sections and add comments, filter, group, and sort graph data, and view trends and correlation.

Configuring the Graph Display

You can configure the graph display by performing any of the following:

➤ Enlarging a Section of a Graph

➤ Configuring Display Options

➤ Adding Comments and Arrows

➤ Viewing the Legend

➤ Configuring Measurement Options

➤ Configuring Columns

Enlarging a Section of a Graph

Graphs initially display data representing the entire duration of the scenario. You can enlarge any section of a graph to zoom in on a specific period of the scenario run. For example, if a scenario ran for ten minutes, you can enlarge and focus on the scenario events that occurred between the second and fifth minutes.

To zoom in on a section of the graph:

1 Click inside a graph.

2 Move the mouse pointer to the beginning of the section you want to enlarge, but not over a line of the graph.

3 Hold down the left mouse button and draw a box around the section you want to enlarge.

4 Release the left mouse button. The section is enlarged.

5 To restore the original view, choose **Clear Display Option** from the right-click menu.

Configuring Display Options

You can configure graph display options at the following levels:

Basic: Enables you to select the type of graph and the time settings.

Advanced: Enables you to modify the scale and format of each graph.

Configuring Basic Display Options

The basic display options let you choose the type of graph to display: line, point, bar, or pie graph. Not all options are available for all graphs.

In addition, you can indicate whether the graph should be displayed with a three-dimensional look, and specify the percent for the three-dimensional graphs. This percentage indicates the thickness of the bar, grid, or pie chart.
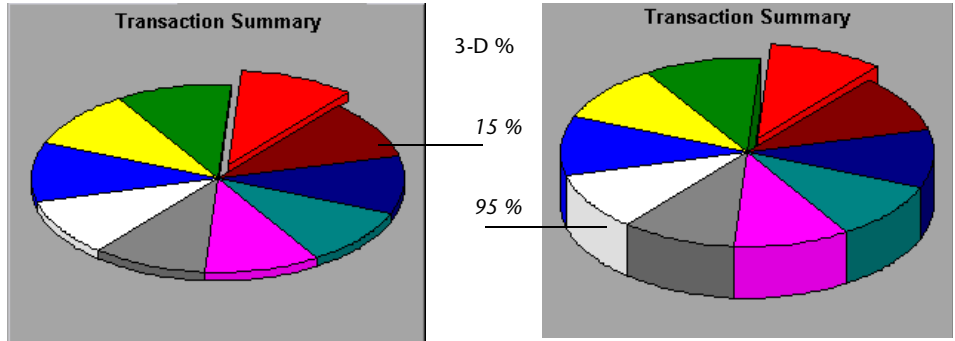
The basic display options also let you indicate how to plot the results that are time-based: relative to the beginning of the scenario (default), or absolute time, based on the system clock of the machine.

To configure graph display options:

1 Choose **View** > **Display Options** or click **Display Options**. The Display Options dialog box opens.
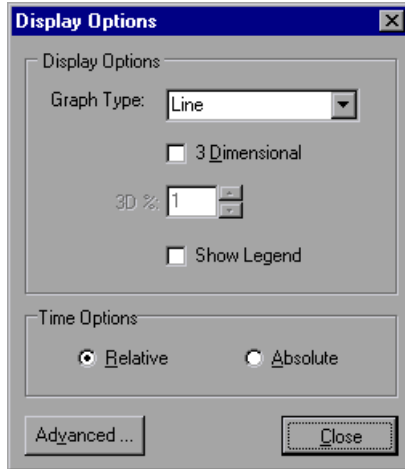
2 Set display and time options, as described in Understanding the Display Options Dialog Box.

3 To set advanced options, click **Advanced**. For more information on advanced options, see "Configuring Advanced Graph Display Options," on page 34.

4 Click **Close**.

Understanding the Display Options Dialog Box

The Display Options dialog box enables you to specify display and time options for the graph and to access the advanced display options.

Display Options

> **Graph Type:** Select the type of graph: bar, line, or point, or pie.

> **3 Dimensional:** Select this check box to enable a 3-dimensional display of the graph.

> **3D %:** Specify a percentage for the 3-dimensional aspect of lines in the graph.

> **Show Legend:** Select this check box to display a legend at the bottom of the graph.

Time Options: Select the way in which the graph shows the Elapsed Scenario Time on the x-axis:

> **Relative:** Relative to the beginning of the scenario

> **Absolute:** The absolute time of the machine's system clock

Advanced: Opens the Editing MainChart dialog box, enabling you to configure the look and feel of your graph as well as its title and the format of the data.

Configuring Advanced Graph Display Options

Advanced options let you configure the look and feel of your graph as well as its title and the format of the data.

To configure advanced graph display options:

🏠 **1** Choose **View** > **Display Options** or click **Display Options**. The Display Options dialog box opens.

2 Click **Advanced**. The Editing MainChart dialog box opens.
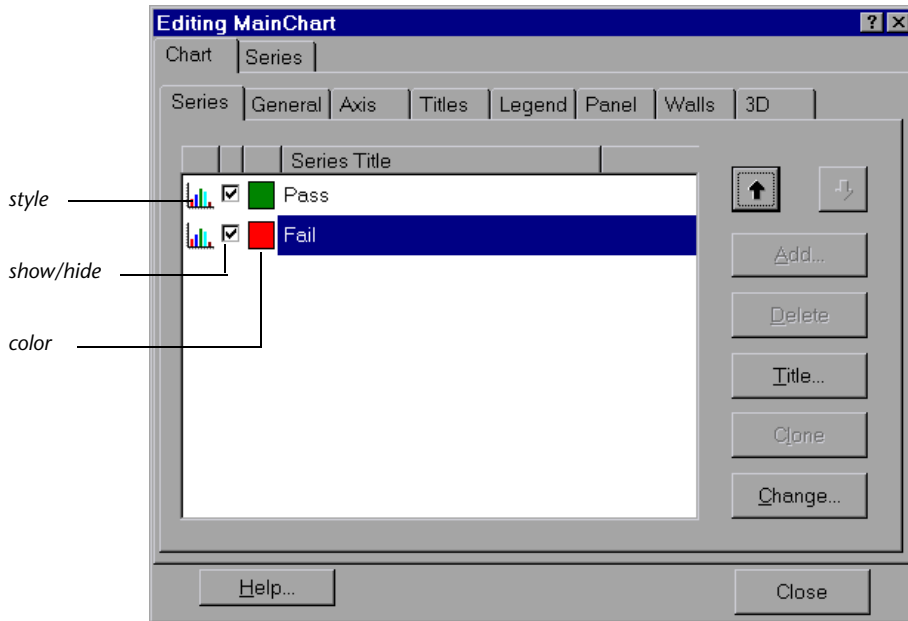
3 Set the desired advanced display options, as described in the following sections, and click **Close**.

You can customize the graph layout by setting the *Chart* and *Series* preferences. Select the appropriate tab and sub-tab to configure your graph.

Understanding the Editing MainChart Dialog Box—Chart Tab

The Chart tab of the Editing MainChart dialog box enables you to configure the look and feel of your entire graph. You set Chart preferences using the following tabs:

Series: Select the graph style (bar, line, etc.), the hide/show settings, line and fill color, and the title of the series.

General: Select options for print preview, export, margins, scrolling, and magnification.

Axis: Select which axes to show, as well as their scales, titles, ticks, and position.

Titles: Set the title of the graph, its font, background color, border, and alignment.

Legend: Set all legend related settings, such as position, fonts, and divider lines.

Panel: Shows the background panel layout of the graph. You can modify its color, set a gradient option, or specify a background image.

Walls: Set colors for the walls of 3-dimensional graphs.

3D: Select the 3-dimensional settings, offset, magnification, and rotation angle for the active graph.

Understanding the Editing MainChart Dialog Box—Series Tab

The Series tab of the Editing MainChart dialog box enables you to control the appearance of the individual points plotted in the graph. You set Series preferences using the following tabs:

Format: Set the border color, line color, pattern, and invert property for the lines or bars in your graph.

Point: Set the size, color and shape of the points that appear within your line graph.

General: Select the type of cursor, the format of the axis values, and show/hide settings for the horizontal and vertical axes.

Marks: Configure the format for each point in the graph.

Adding Comments and Arrows

You can tailor your graph by adding comments and arrows to clarify graphical data and specify significant points or areas.

Adding or Editing Comments

You add or edit comments using the Comment dialog box. The name of the dialog box changes to reflect whether you are adding or editing a comment.

To add or edit a comment:

1 To add a comment, perform either of the following:

➤ Right-click in the graph at the location where you want to add the comment, and choose **Comments** > **Add**.

➤ Click **Add Comments**. The cursor changes to a drag icon. Click in the graph where you want to position the comment.

2 To edit a comment, right-click in the graph, and choose **Comments** > **Edit.**

The Comment dialog box opens.
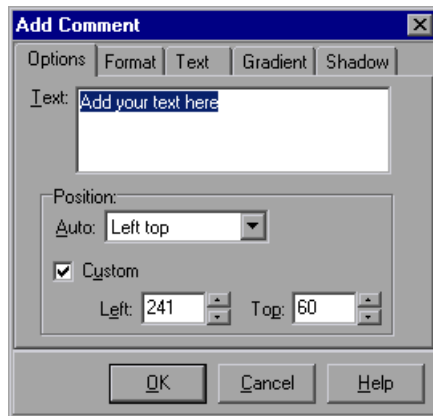
3 Type a comment in the **Text** box.

The comment text is displayed in the graph in the location specified in step 1.

4 To format a comment, select the **Format**, **Text**, **Gradient**, and **Shadow** tabs.

5 To delete a comment, select the comment and click **Delete** (displayed in Edit mode only).

6 Click **OK**.

Understanding the Comment Dialog Box

The Comment dialog box enables you to add and edit comments in the current graph.

Text: Type the text of your comment, or edit the text of an existing comment.

Position: Select an automatic or a custom position.

Auto: You can select a defined location for your comment:

    Left top: Positions the comment at the top left corner of the graph.

    Left bottom: Positions the comment at the lower left corner of the graph.

    Right top: Positions the comment at the top right corner of the graph.

    Right bottom: Positions the comment at the lower right corner of the graph.

Custom: Select this to position the comment in a custom location.

    Left: Set the position of the comment from the left of the graph.

    Top: Set the position of the comment from the top of the graph.

Using Arrows in a Graph

You can add and delete arrows in a graph.

To add an arrow to a graph:

1 Click **Draw Arrow**. The cursor changes to a hairline icon.

2 Click the mouse button within the graph to position the base of the arrow.

3 While holding the mouse button down, drag the mouse cursor to position the head of the arrow. Release the mouse button.

4 The arrow's position can be changed by selecting the arrow itself. Positional boxes appear at the base and head which can be dragged to alternate positions.

To delete an arrow from a graph:

1 Select the arrow by clicking on it. Positional boxes appear at the base and head of the arrow.

2 Press **Delete**.

Viewing the Legend

You can use the Legend tab to configure the color, scale, minimum, maximum, average, median, and standard deviation of each measurement appearing in the graph.

Color	Scale	Measurement	Minimum	Maximum	Average	Median	Std. Deviation
☑	1	air_head2_gif_Action1_159	0.176	0.657	0.247	0.19	0.155
☑	1	beige_gif_Action1_183	0.167	0.434	0.24	0.194	0.092
☑	1	book_gif_Action1_167	0.198	0.812	0.291	0.2	0.2
☑	1	bullet1_gif_Action1_234	0.11	0.245	0.164	0.17	0.04
☑	1	butt_ast_gif_Action1_207	0.13	0.238	0.177	0.176	0.027
☑	1	Click_here_Action1_107	0.187	0.827	0.288	0.201	0.22

Tabs: Legend | Graph Details | User Notes | Graph Data | Raw Data

The Legend tab shortcut (right-click) menu enables the following additional options:

➤ **Show**: Displays a measurement in the graph.

➤ **Hide**: Hides a measurement in the graph.

➤ **Show only selected:** Displays the highlighted measurement only.

➤ **Show all**: Displays all the available measurements in the graph.

➤ **Configure measurements**: Opens the Measurement Options dialog box that enables you to configure measurement options (for example, set color and measurement scale). For more information, see "Configuring Measurement Options," on page 40.

➤ **Show measurement description**: Displays a dialog box with the name, monitor type, and description of the selected measurement.

➤ **Animate selected line**: Displays the selected measurement as a flashing line.

➤ **Web page breakdown for <selected measurement>** (appears for measurements in the Average Transaction Response Time and Transaction Performance Summary graphs): Displays a Web Page Breakdown graph for the selected transaction measurement.

➤ **Auto Correlate**: Opens the Auto Correlate dialog box that enables you to correlate the selected measurement with other monitor measurements in the scenario. For more information on auto correlation, see "Auto Correlating Measurements," on page 60.

➤ **Sort by measurement column**: Sorts the measurements according to the selected column, in ascending or descending order.

➤ **Configure columns**: Opens the Legend Columns Options dialog box that enables you to configure the columns displayed in the Legend tab. For more information, see "Configuring Columns," on page 42.

➤ **Break down** (appears for measurements in the Web Page Breakdown graphs): Displays a graph with a breakdown of the selected page.

Configuring Measurement Options

You can configure measurement options using the Configuring Measurements dialog box.

To configure measurement options:

1 In the Legends tab, right-click and choose **Configure measurements**. The Measurement Options dialog box opens.
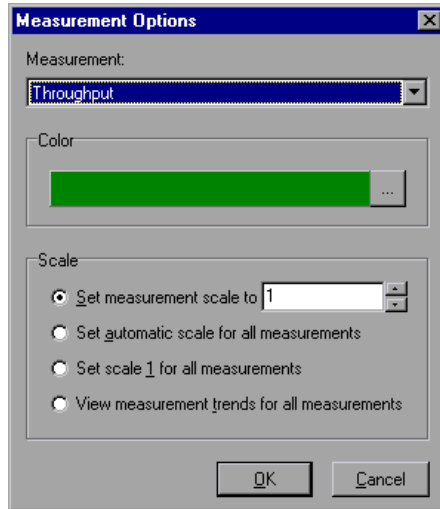
2 Select a measurement to configure.

3 Select color and scale options and click **OK**.

Understanding the Measurement Options Dialog Box

The Measurement Options dialog box enables you to set the color and scale of the measurement you selected.

Measurement: Select a measurement from the drop-down list.

Color: Select the color bar and choose a color for the selected measurement.

Scale

> **Set measurement scale to x:** Select the scale with which you want to view the selected measurement.
>
> **Set automatic scale for all measurements:** Uses an automatic scale optimized to display each measurement in the graph.
>
> **Set scale 1 for all measurements:** Sets the scale to one for all measurements in the graph.
>
> **View measurement trends for all measurements:** Standardizes the y-axis values in the graph, according to the following formula: New Y value = (Previous Y Value - Average of previous values) / STD of previous values. For more information on standardizing graph values, see "Viewing Measurement Trends," on page 59.

Configuring Columns

You can use the Legend Columns Options dialog box to select the columns to be displayed and to configure their position, width, and sort order.

To configure columns:

1 In the Legends tab, right-click and choose **Configure columns**. The Legend Columns Options dialog box opens.
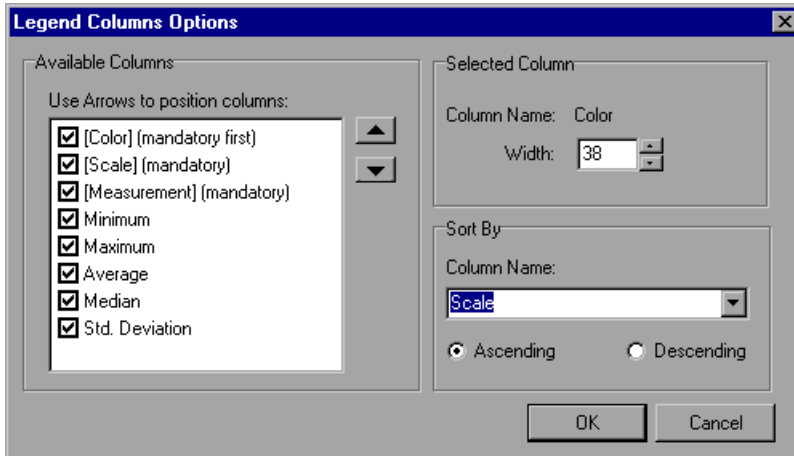
2 Configure the columns to be displayed, their position, width, and sort order, as described in "Understanding the Legend Columns Options Dialog Box," on page 42.

3 Click **OK**.

Understanding the Legend Columns Options Dialog Box

The Legend Columns Options dialog box enables you to select the columns you want to view, the width of each column, and the method by which to sort the columns.

Available Columns: Displays the columns available for the selected measurement. By default, all of the available columns are displayed .To hide a column, clear the check box to the left of the column name. To show it, select the check box. Note that several columns are mandatory and cannot be hidden, such as Measurement Name, Color, and Scale.

You can rearrange the order in which the columns appear (from left to right), by using the vertical arrows to position the columns in the desired order.

Selected Column

> **Column Name:** Displays the name of the selected column (read-only)
>
> **Width:** Displays the width of the column in pixels. Note that you can also modify a column's width by dragging its border in the Legend tab.

Sort By

> **Column Name:** Select the column by which to sort the measurement data.
>
> **Ascending:** Sort the measurement data in ascending order.
>
> **Descending:** Sort the measurement data in descending order.

Configuring the Data

You can customize the data displayed in your graphs with the following data operations:

➤ Filtering and Sorting Graph Data

➤ Viewing the Legend

➤ Viewing the Data as a Spreadsheet and as Raw Data

Filtering and Sorting Graph Data

You can filter and sort data that is displayed in a graph. You sort and filter graph data using the same dialog box, as described in the following sections.

Filtering Graph Data

You can filter graph data to show fewer transactions for a specific segment of the scenario. More specifically, you can display four transactions beginning from five minutes into the scenario and ending three minutes before the end of the scenario.

You can filter for a single graph, in all graphs in a scenario, or in the summary graph.

The available filter conditions differ for each type of graph. The filter conditions also depend on your scenario. For example, if you only had one group or one load generator machine in your scenario, the Group Name and Load Generator Name filter conditions do not apply.

Note that you can also filter merged graphs. The filter conditions for each graph are displayed on separate tabs.

Sorting Graph Data

You can sort graph data to show the data in more relevant ways. For example, Transaction graphs can be grouped by the Transaction End Status, and Vuser graphs can be grouped by Scenario Elapsed Time, Vuser End Status, Vuser Status, and VuserID.

You can sort by one or several groups—for example by Vuser ID and then Vuser status. The results are displayed in the order in which the groups are listed. You can change the grouping order by rearranging the list. The following graph shows the Transaction Summary grouped according to Vusers.
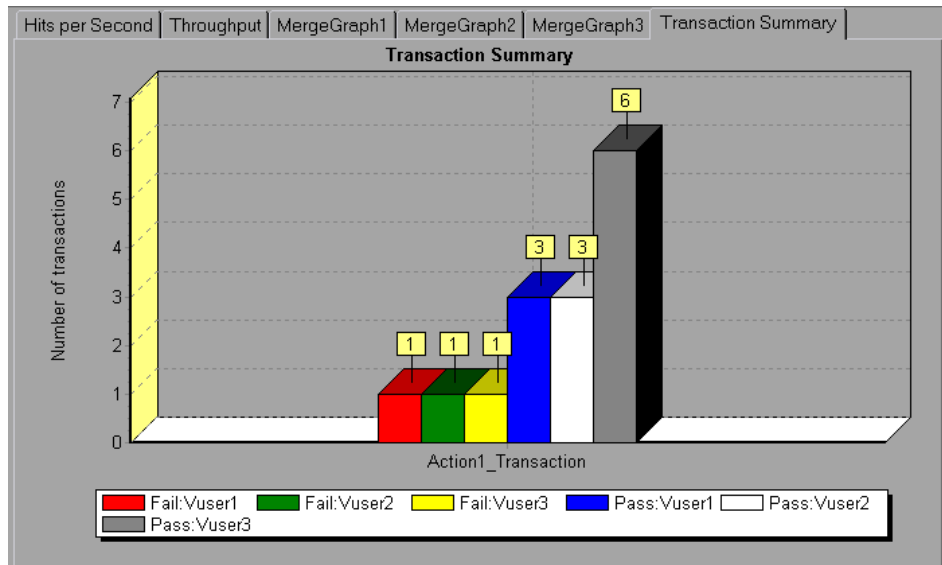
Applying Filter and Sort Criteria to Graphs

You apply filter and sort criteria to graphs using the following dialog boxes:

To Filter This Graph	Use This Dialog Box
A single graph	Graph Settings dialog box
All graphs in a scenario	Global Filter dialog box
Summary Report	Analysis Summary Filter dialog box

These dialog boxes are all similar, but the name and some of the options change depending on its use.

Note: You can only sort a single graph at a time. Sort settings are only included in the Graph Settings dialog box.

To set filter conditions:

1 Open the appropriate dialog box by performing the following:

➤ To set filter conditions for a single graph, select a graph to filter by clicking the graph tab or clicking the graph name in the tree view, and choose **View > Set Filter/Group By** or click **Set Filter**.

➤ To set filter conditions for all the graphs in a scenario (both those displayed and those that have not yet been opened), choose **File > Set Global Filter** or click **Set Global Filter**

➤ To set filter conditions for the Summary Report, choose **View > Summary Filter**.

A graph filter dialog box opens.
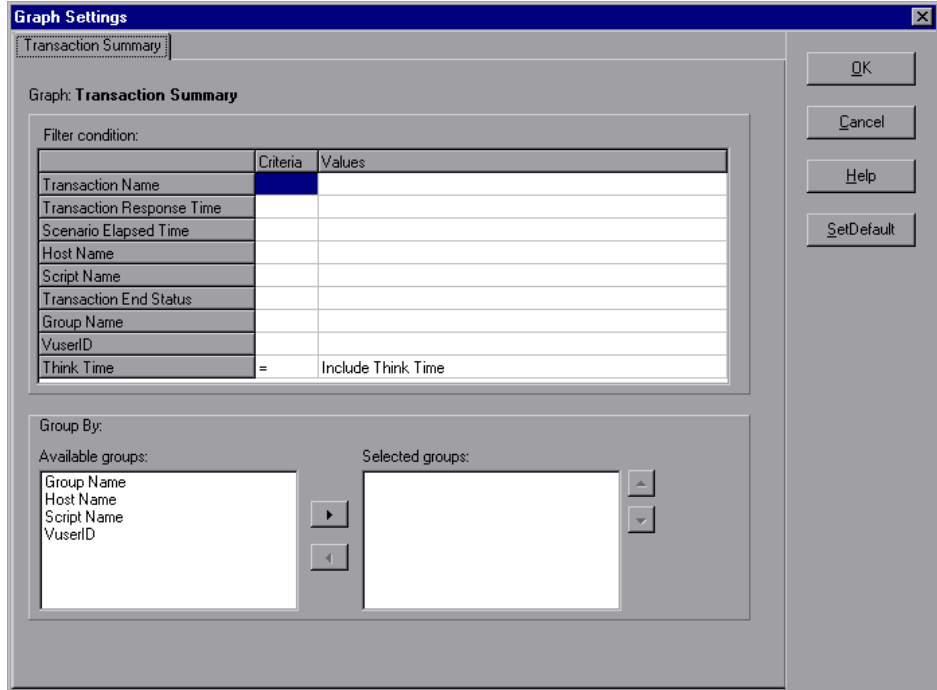
2 Select values for the **Criteria** and **Values** fields.

For some filter conditions, one of the following dialog boxes opens to enable you to additional filtering details.

➤ Set Dimension Information Dialog Box

➤ Vuser ID Dialog Box

➤ Scenario Elapsed Time Dialog Box

Set the conditions in the additional dialog box, as described for each dialog box.

3 To sort the graph display by group, select the Group By settings, as described in "Understanding the Filter Dialog Boxes," on page 47.

4 Click **OK**. The graph dialog box closes and the settings are applied.

Understanding the Filter Dialog Boxes

The filter dialog boxes (Graph Settings, Global Filter, and Analysis Summary Filter) enable you to filter the data that is displayed in the graph. Some of the following fields are not displayed in all of the filter boxes.

Filter Condition: Select criteria and values for each filter condition that you want to employ. The applicable filter conditions are displayed for each graph.

> **Criteria:** Select "=" (equals) or "<>" (does not equal).

> **Values:** Select a value from the **Values** list. The filter conditions are grouped into three value types (discrete, continuous, and time-based).

> A discrete value is a distinct integer (whole number) value such as Transaction Name or Vuser ID. Select the check box(es) of the value(s) that you want to include in your filter.

> Note: You can filter sub-transactions using the Transaction Father Tree Path condition. Select "<Transaction_name>" to filter the parent's sub-transactions; "NONE" to filter for parent transactions; or "Unknown" to filter for sub-transactions whose parents are unknown (usually caused by nesting errors during the session).

> A continuous value is a variable dimension that can take any value within the minimum and maximum range limits, such as Transaction Response Time. You set the dimension information for each measurement in the Set Dimension Information Dialog Box.

> A time-based value is a value that is based on time relative to the start of the scenario. Scenario Elapsed Time is the only condition that uses time-based values. You specify time-based values in the Scenario Elapsed Time Dialog Box.

Group By: Use these settings to sort the graph display by group.

> **Available groups:** Select the group by which you want to sort the results, and click the right arrow.

> **Selected groups:** Displays a list of all the selected groups by which the results will be sorted. To remove a value, select it and click the left arrow.

To change the order in which the results are grouped, select the group you want to move and click the up or down arrow until the groups are in the desired order.

Set Default: Displays the default criteria and values for each filter condition.

Clear All: Deletes all of the information you entered in the dialog box.

Set Dimension Information Dialog Box

The Set Dimension Information dialog box enables you to set the dimension information for each measurement (transaction, number of released Vusers, resource) in the result set. You specify the minimum and maximum values for each measurement you want in the analysis. By default, the full range of values for each measurement is displayed.
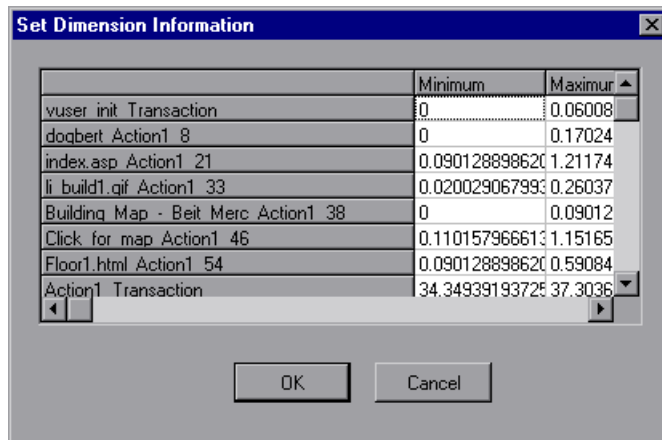
Minimum: Specify a minimum value for the measurement.

Maximum: Specify a maximum value for the measurement.

Note: If you are specifying the start and end time for a transaction (in minutes:seconds format), the time is relative to the beginning of the scenario.

The Set Dimension Information dialog box opens to enable the entering of additional filter information in the following cases:

➤ For the Transaction Response Time filter condition.

➤ For Rendezvous graphs, while setting the *Number of Released Vusers* condition. Specify a minimum and maximum number of released Vusers.

➤ For all graphs that measure resources (Web Server, Database Server, etc.), when you set the *Resource Value* condition. The dialog box opens displaying a full range of values for each resource. Specify a minimum and maximum value for the resource. Specify a minimum and maximum transaction response time for each transaction.

Vuser ID Dialog Box

The Vuser ID dialog box opens to enable the entering of additional filter information for the Vuser ID filter condition.
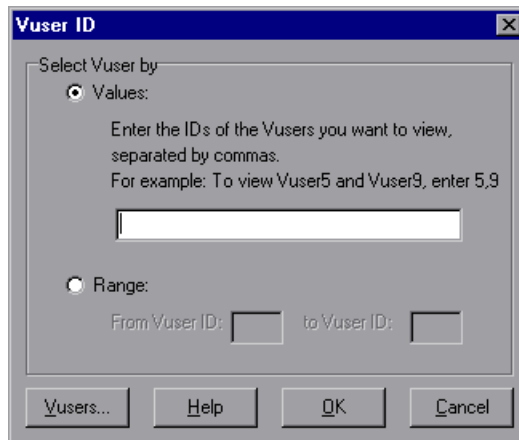
Select Vuser by

Value: Enter the Vuser IDs of the Vusers you want the graph(s) to display, separated by commas.

Range: Specify the beginning and end of the desired range of Vusers you want the graph(s) to display.

Vusers: Displays the existing Vuser IDs from which you can choose.

Scenario Elapsed Time Dialog Box

The Scenario Elapsed Time dialog box opens to enable you to specify the start and end times for the graph's x-axis.
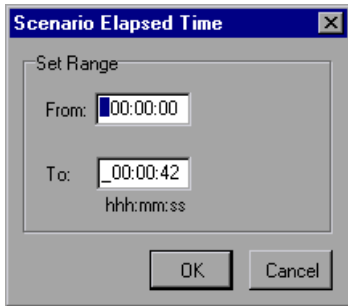
Specify the start and end time for the graph in hours:minutes:seconds format. The time is relative to the start of the scenario.

Set Range

From: Specify a start value for the desired range.

To: Specify an end value for the desired range.

Note: The time is relative to the start of the scenario.

Analyzing Graph Results

You can analyze graph data using the following methods:

➤ Determining a Point's Coordinates

➤ Drilling Down in a Graph

➤ Changing the Granularity of the Data

➤ Viewing the Data as a Spreadsheet and as Raw Data

➤ Viewing Measurement Trends

➤ Auto Correlating Measurements

➤ Using the WAN Emulation Overlay

Determining a Point's Coordinates

You can determine the coordinates and values at any point in a graph. Place the cursor over the point you want to evaluate and Analysis displays the axis values and other grouping information.
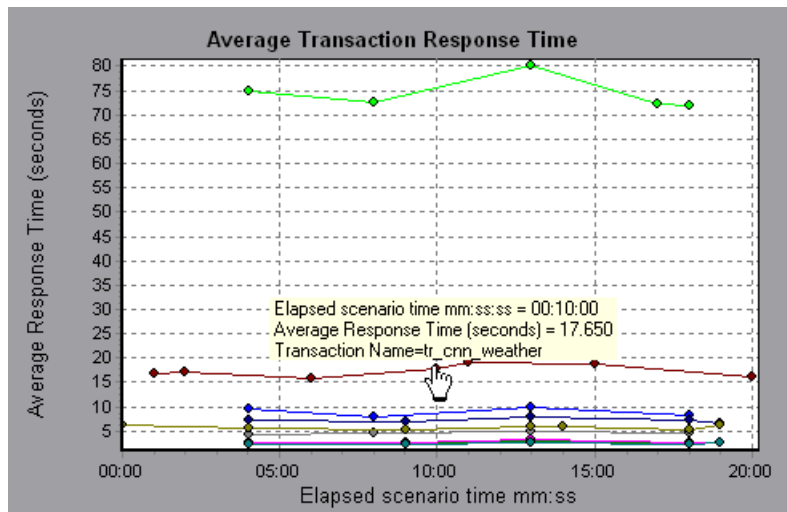
Drilling Down in a Graph

Drill down enables you to focus on a specific measurement within your graph and display it according to a desired grouping. The available groupings depend on the graph. For example, the Average Transaction Response Time graph shows one line per transaction. To determine the response time for each Vuser, you drill down on one transaction and sort it according to Vuser ID. The graph displays a separate line for each Vuser's transaction response time.

Note: The drill down feature is not available for the Web Page Breakdown graph.

The following graph shows a line for each of five transactions.
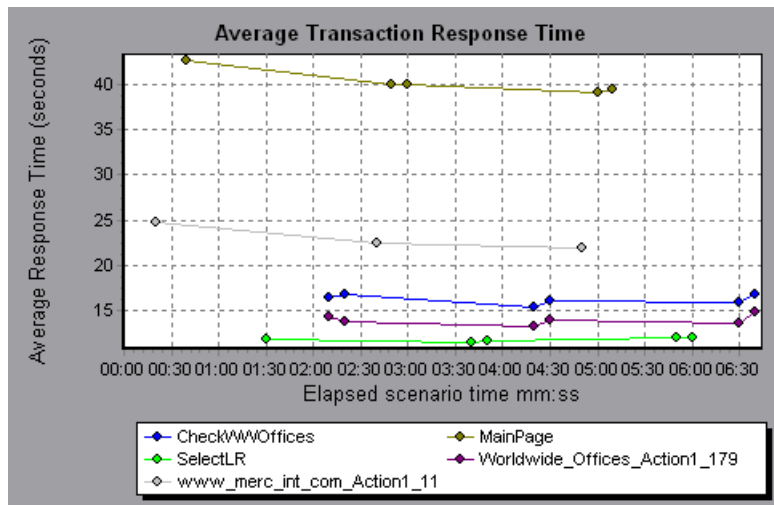
When you drill down on the MainPage transaction, grouped by Vuser ID, the graph displays the response time only for the MainPage transaction, one line per Vuser.
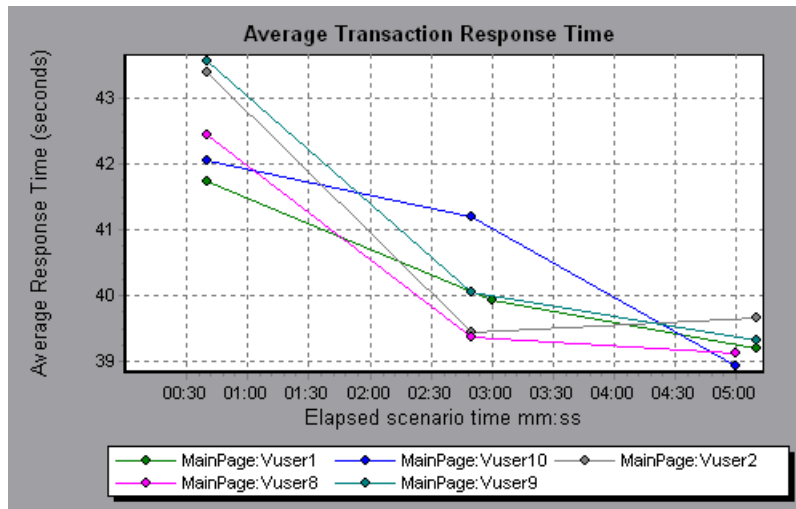
You can see from the graph that the response time was longer for some Vusers than for others.

To determine the response time for each host, you drill down on one transaction and sort it according to host. The graph displays a separate line for the transaction response time on each host.

Performing a Drill Down in a Graph

To drill down in a graph:

1 Right-click on a line, bar, or segment within the graph, and select **Drill Down**. The Drill Down Options dialog box opens, listing all of the measurements in the graph.
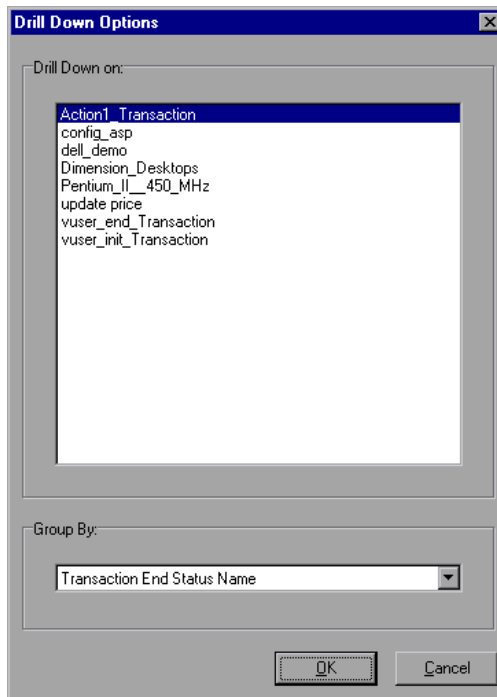
2 Select a measurement for drill down.

3 From the **Group By** box, select a group by which to sort.

4 Click **OK**. Analysis drills down and displays the new graph.

To undo the last drill down settings, choose **Undo Set Filter/Group By** from the right-click menu.

To perform additional drill downs, repeat steps 1 to 4.

To clear all filter and drill down settings, choose **Clear Filter/Group By** from the right-click menu.

Changing the Granularity of the Data

You can make the graphs easier to read and analyze by changing the granularity (scale) of the x-axis. The maximum granularity is half of the graph's time range. To ensure readability and clarity, Analysis automatically adjusts the minimum granularity of graphs with ranges of 500 seconds or more.

In the following example, the Hits per Second graph is displayed using different granularities. The y-axis represents the number of hits per second within the granularity interval. For a granularity of 1, the y-axis shows the number of hits per second for each one second period of the scenario. For a granularity of 5, the y-axis shows the number of hits per second for every five-second period of the scenario.

Granularity=1

Granularity=5
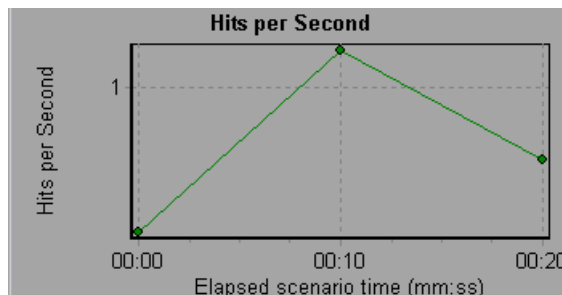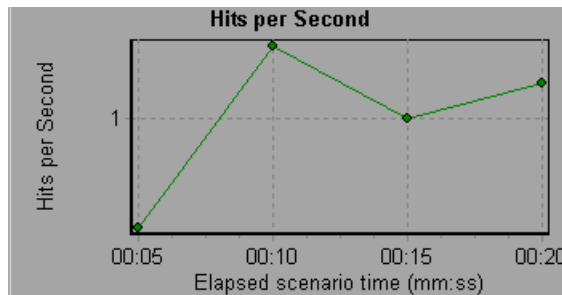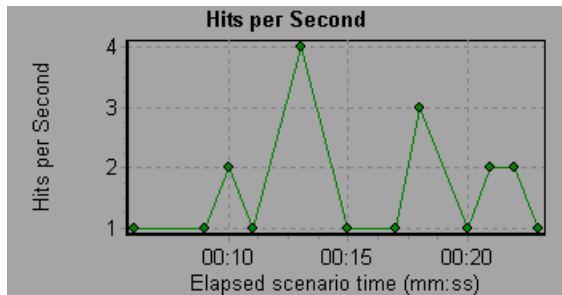
Granularity=10

In the above graphs, the same scenario results are displayed in a granularity of 1, 5, and 10. The lower the granularity, the more detailed the results. For example, using a low granularity as in the upper graph, you see the intervals in which no hits occurred. It is useful to use a higher granularity to study the overall Vuser behavior throughout the scenario.

By viewing the same graph with a higher granularity, you can easily see that overall, there was an average of approximately 1 hit per second.

To change the granularity of a graph:

 1 Click inside a graph.

 2 Select **View** > **Set Granularity**, or click **Set Granularity** . The Granularity dialog box opens.
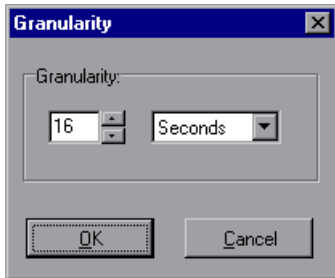
 3 Enter the granularity of the x axis and select a time measurement. The maximum granularity is half of the graph's time range.

To ensure readability and clarity, LoadRunner automatically adjusts the minimum granularity of graphs with ranges of 500 seconds or more.

 4 Click **OK**.

Viewing the Data as a Spreadsheet and as Raw Data

Analysis allows you to view graph data in the following ways:

Spreadsheet View: View the graph values displayed in the Graph Data tab.

Raw Data View: View the actual raw data collected during the scenario, displayed in the Raw Data tab.

Spreadsheet View

You can view the graph in spreadsheet format using the **Graph Data** tab.

Transaction Name	Minimum	Average	Maximum
air_head2_gif_Action1_159	0.12	0.304	3.105
beige_gif_Action1_183	0.111	0.27	1.462
book_gif_Action1_167	0.19	0.373	3.314
bullet1_gif_Action1_234	0.11	0.184	0.701
butt_ast_gif_Action1_207	0.13	0.184	0.311
Click_here_Action1_107	0.16	0.33	2.124

The first column displays the values of the x-axis. The following columns show the y-axis values for each transaction.

If there are multiple values for the y-axis, as in the Transaction Performance Summary graph (minimum, average, and maximum), all of the plotted values are displayed. If you filter out a transaction, it will not appear in the view.

The Spreadsheet shortcut menu (right-click) has the following additional features:

Copy All: Copies the spreadsheet to the clipboard. You can paste it to a spreadsheet.

Save As: Saves the spreadsheet data to an Excel file. Once you have the data in Excel, you can generate your own customized graphs.

Viewing Raw Data

You can view the actual raw data collected during test execution for the active graph. The Raw Data view is not available for all graphs.

Viewing the raw data can be especially useful in the following cases:

➤ To determine specific details about a peak—for example, which Vuser was running the transaction that caused the peak value(s).

➤ To perform a complete export of unprocessed data for your own spreadsheet application.

To display a graph's Raw Data view:

1 Choose **View** > **View Raw Data** or click **Raw Data**. The Raw Data dialog box opens.
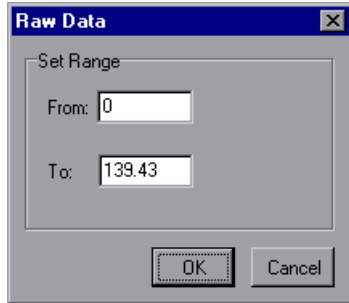
2 Specify a time range, and click **OK**.

3 To view the raw data, select the **Raw Data** tab.

Understanding the Raw Data Tab

The Raw Data tab displays the raw data appearing in the active graph.

The Raw Data shortcut menu (right-click) has the following additional features:

Copy All: Allows you to copy the raw data.

Save As: Allows you to save the raw data.

Understanding the Graph Details Tab

The Graph Details tab displays a description of the selected graph, the graph's title, the result file, and the filters that have been applied to the graph.

The information on this tab can be displayed in Microsoft Word reports. For more information, see "Creating Microsoft Word Reports," on page 360.

Understanding the User Notes Tab

The User Notes tab displays user notes on the graph. You can enter text into this tab, which will be saved with the session.

The information on this tab can be displayed in Microsoft Word reports. For more information, see "Creating Microsoft Word Reports," on page 360.

Viewing Measurement Trends

You can view the pattern of a line graph more effectively by standardizing the graph's y-axis values. Standardizing a graph causes the graph's y-axis values to converge around zero. This cancels the measurements' actual values and allows you to focus on the behavior pattern of the graph during the course of the scenario.

Analysis standardizes the y-axis values in a graph according to the following formula:

New Y value = (Previous Y Value - Average of previous values) / STD of previous values

To view a line graph as a standardized graph:

1 Select **View** > **View Measurement Trends**, or right-click the graph and choose **View Measurement Trends**. Alternatively, you can select **View > Configure Measurements** and check the View measurement trends for all measurements box.

Note: The standardization feature can be applied to all line graphs except the Web Page Breakdown graph.

---

2 View the standardized values for the line graph you selected. Note that the values in the Minimum, Average, Maximum, and Std. Deviation legend columns are real values.

To undo the standardization of a graph, repeat step 1.

Note: If you standardize two line graphs, the two y-axes merge into one y-axis.

Auto Correlating Measurements

You can detect similar trends among measurements by correlating a measurement in one graph with measurements in other graphs. Correlation cancels the measurements' actual values and allows you to focus on the behavior pattern of the measurements during a specified time range of the scenario.

In the following example, the t106Zoek:245.lrr measurement in the Average Transaction Response Time graph is correlated with the measurements in the Windows Resources, Microsoft IIS, and SQL Server graphs. The five measurements most closely correlated with t106Zoek:245.lrr are displayed in the graph below.
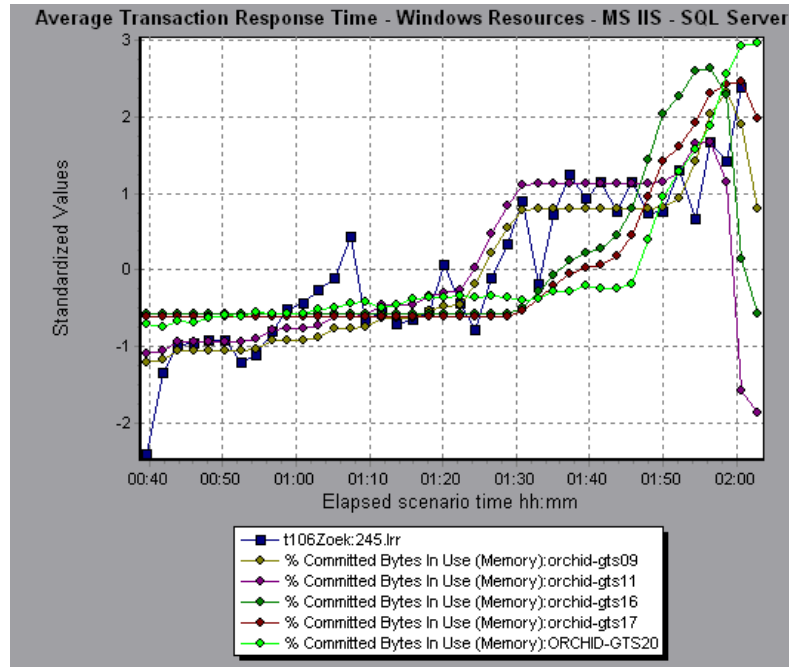
Note: This feature can be applied to all line graphs except the Web Page Breakdown graph.

For more information on auto correlation, see Chapter 25, "Interpreting Analysis Graphs."

Correlating Graph Measurements Automatically

To correlate graph measurements automatically:

 1 From a graph or legend, right-click the measurement you want to correlate
 and choose **Auto Correlate**. The Auto Correlate dialog box opens with the
 selected measurement displayed in the graph.
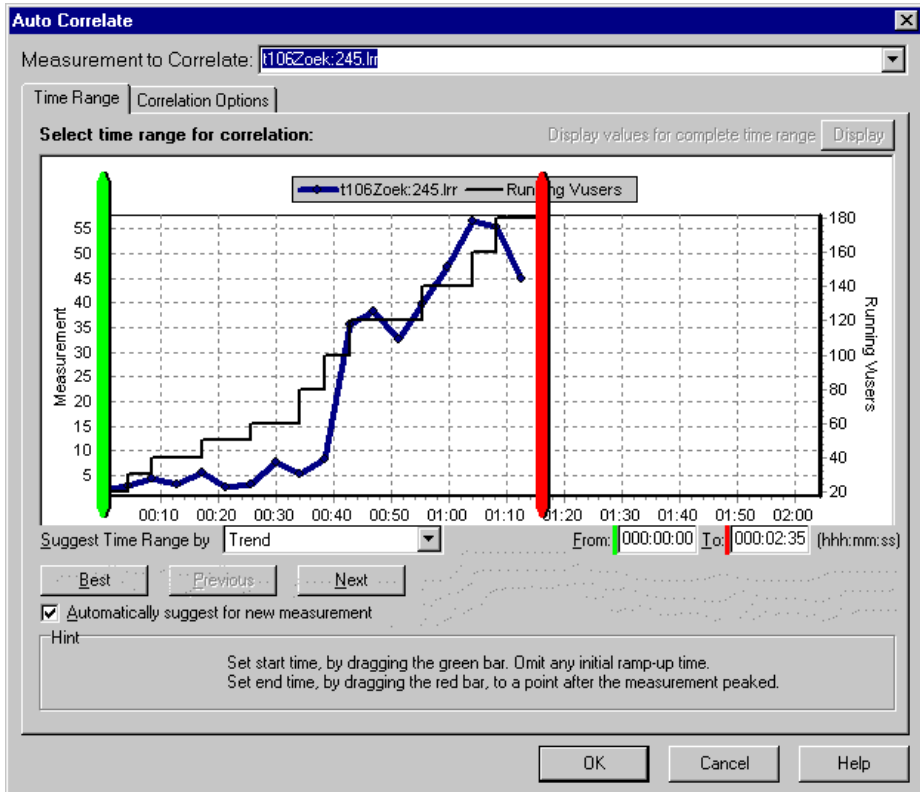
The following settings configure the Auto Correlate tool to automatically
demarcate the most significant time period for the measurement in the
scenario. For more information, see "Understanding the Auto Correlate
Dialog Box—Time Range Tab," on page 64.

2 Select a suggested time range method and time range.

3 If you applied a time filter to your graph, you can correlate values for the complete scenario time range by clicking the **Display** button in the upper right-hand corner of the dialog box.

Note: The granularity of the correlated measurements graph may differ from that of the original graph, depending on the scenario time range defined.

---

4 To specify the graphs you want to correlate with a selected measurement and the type of graph output to be displayed, perform the following:

➤ Select the **Correlation Options** tab.

➤ Select the graphs to correlate, the data interval, and output options, as described in "Understanding the Auto Correlate Dialog Box—Correlation Options Tab," on page 65.
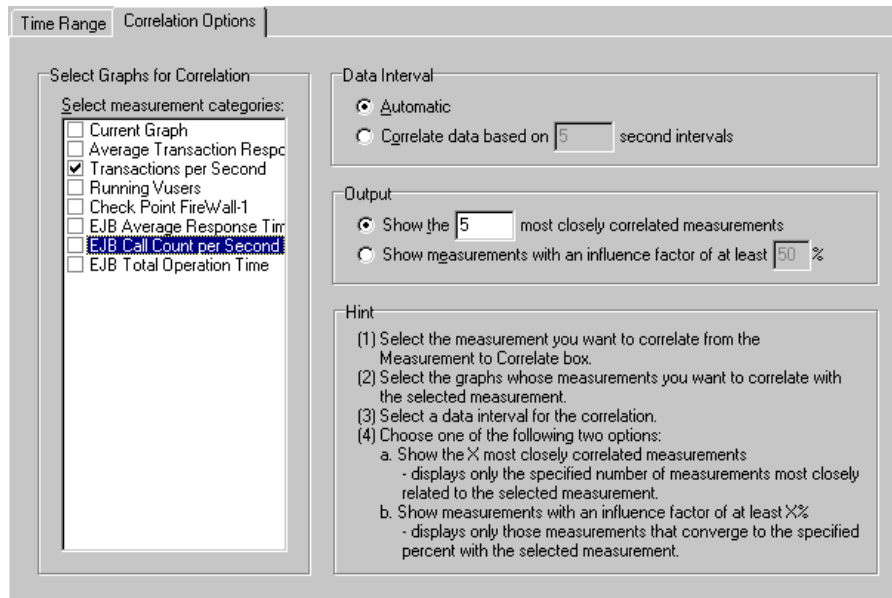
5 On the **Time Range** tab, click **OK**. Analysis generates the correlated graph you specified. Note the two new columns—*Correlation Match* and *Correlation*—that appear in the Legend tab below the graph.

To specify another measurement to correlate, select the measurement from the Measurement to Correlate box at the top of the Auto Correlate dialog box.

The minimum time range should be more than 5% of the total time range of the measurement. Trends which are smaller than 5% of the whole measurement will be contained in other larger segments.

Sometimes, very strong changes in a measurement can hide smaller changes. In cases like these, only the strong change is suggested, and the **Next** button will be disabled.

Understanding the Auto Correlate Dialog Box—Time Range Tab

The Time Range tab of the Auto Correlate dialog box enables you to specify a scenario time range for the correlated measurement graph.

Measurement to Correlate: Select the measurement you want to correlate.

Display values for complete time range: Click **Display** to correlate values for the complete scenario time range. This option is available only if you applied a time filter to your graph.

Suggest Time Range By: Analysis automatically demarcates the most significant time period for the measurement in the scenario.

    Trend: Demarcates an extended time segment which contains the most significant changes.

    Feature: Demarcates a smaller dimension segment which forms the trend.

Best: Choose the time segment most dissimilar to its adjacent segments.

Next: Suggest the next time segment for auto correlation. Each suggestion is successively less dissimilar.

Previous: Return to the previous suggestion of a time segment.

Automatically suggest for new measurement: Generates new suggestions each time that the Measurement to Correlate item changes.

From: Specify a start value (in hhh:mm:ss format) for the desired scenario time range.

To: Specify an end value (in hhh:mm:ss format) for the desired scenario time range.

You can also use the green and red vertical drag bars to specify the start and end values for the scenario time range.

Note: The granularity of the correlated measurements graph may differ from that of the original graph, depending on the scenario time range defined.

Understanding the Auto Correlate Dialog Box—Correlation Options Tab

You use the Correlation Options tab of the Auto Correlate dialog box to set the graphs to correlate, the data interval, and the output options.

Select Graphs for Correlation: Select the graphs whose measurements you want to correlate with your selected measurement.

Data Interval: Calculate the interval between correlation measurement polls.

> **Automatic:** Uses an automatic value, determined by the time range.

> **Correlate data based on X second intervals**: Enter a fixed value.

Output: Choose the level of output displayed.

> **Show the X most closely correlated measurements:** Displays only the specified number of measurements most closely related to the selected measurement. The default setting is 5.

> **Show measurements with an influence factor of at least X%:** Displays only those measurements that converge to the specified percent with the selected measurement. The default setting is 50%.

Using the WAN Emulation Overlay

During scenario execution, you can use WAN effects such as latency, packet loss, link faults, and dynamic routing to characterize many aspects of the WAN cloud. Using the WAN emulation overlay during Analysis, you can display the time period(s) in a scenario that the WAN emulator was active. By comparing measurements taken during WAN emulation to measurements taken with the WAN emulator feature disabled, you can see the impact of WAN settings on your network performance.

To view the WAN emulation overlay:

1 Click inside a graph.

2 Select **View** > **Overlay with WAN Emulation**, or right-click the graph and choose **Overlay with WAN Emulation**. A line appears on the selected graph representing time periods during which WAN emulation was enabled.

The Effect of WAN Emulation on a Scenario

The WAN emulator delays packets, loses packets, fragments data, and emulates other network phenomena according to the parameters you set. The effect of WAN emulation on a scenario can be seen on the Transaction and Web Resource graphs. When WAN emulation is enabled, the time taken to perform a transaction is higher, and the amount of throughput on the server is reduced.

In addition, a scenario running with WAN emulation takes longer to complete than a scenario running with the WAN emulation function disabled. This is due to the delays caused by the packet latency, packet loss, and link disconnection settings.

In the following example, the Average Transaction Response Time Graph is displayed with the WAN Emulation overlay. WAN emulation was enabled between the 1st and 3rd minute of the scenario. During this time, the average transaction response times increased sharply. When WAN emulation was stopped, average response times dropped.
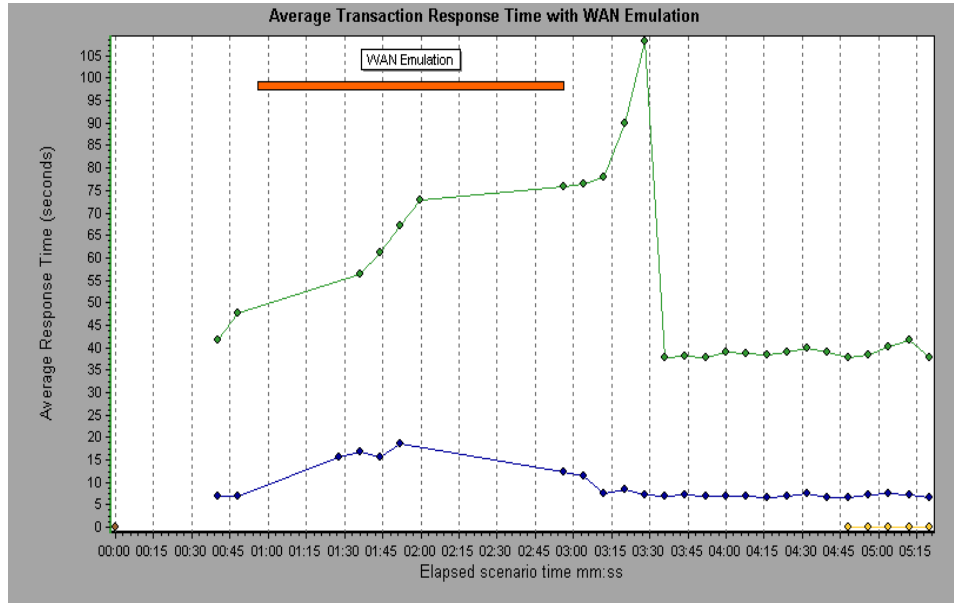
In the same scenario, throughput on the server fell during the period of WAN emulation. After WAN emulation was stopped, server throughput increased. You can also compare this graph to the Average Transaction Response Time Graph to see how the throughput affects transaction performance.
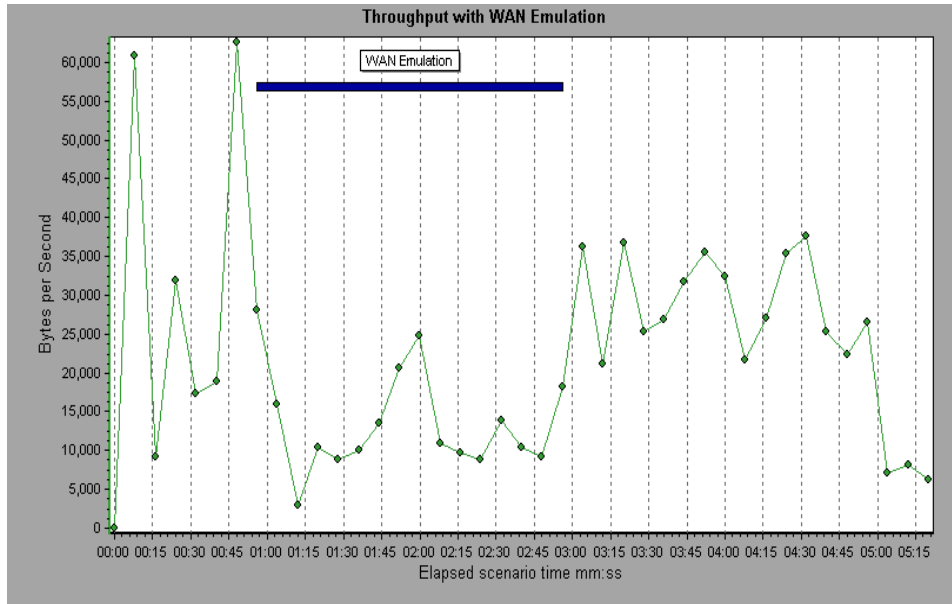
Printing Graphs

You can print all or selected displayed graphs.

To print graphs:

 1 Choose **File** > **Print**. The Print Graph(s) dialog box opens.
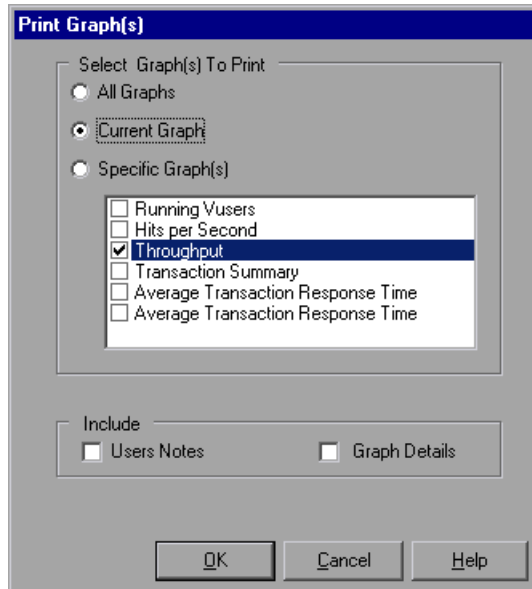
 2 Select the graphs to print.

 3 To include notes from the User Notes tab, select **User Notes**.

 4 To include information from the Graph Details tab, select **Graph Details**.

 5 Click **OK**.

Understanding the Print Graph(s) Dialog Box

You use the The Print Graph(s) dialog box to define print settings and print the selected graph(s).

Select Graph(s) to Print

All Graphs: Prints all graphs in the current session.

Current Graph: Prints the graph currently displayed in the right pane.

Specific Graph(s): Select the graphs to print.

Include

Users Notes: Prints the notes in the User Notes tab.

Graph Details: Prints the information in the Graph Details tab.

3

Vuser Graphs

After running a scenario, you can check the behavior of the Vusers that participated in the scenario using the following Vuser graphs:

➤ Running Vusers Graph

➤ Vuser Summary Graph

➤ Rendezvous Graph

About Vuser Graphs

During scenario execution, Vusers generate data as they perform transactions. The Vuser graphs let you determine the overall behavior of Vusers during the scenario. They display the Vuser states, the number of Vusers that completed the script, and rendezvous statistics. Use these graphs in conjunction with Transaction graphs to determine the effect of the number of Vusers on transaction response time.

Running Vusers Graph

The Running Vusers graph displays the number of Vusers that executed Vuser scripts and their status during each second of the test. This graph is useful for determining the Vuser load on your server at any given moment. By default, this graph only shows the Vusers with a *Run* status. To view another Vuser status, set the filter conditions to the desired status. For more information, see Chapter 2, "Working with Analysis Graphs."

The x-axis represents the elapsed time from the beginning of the scenario run. The y-axis represents the number of Vusers in the scenario.
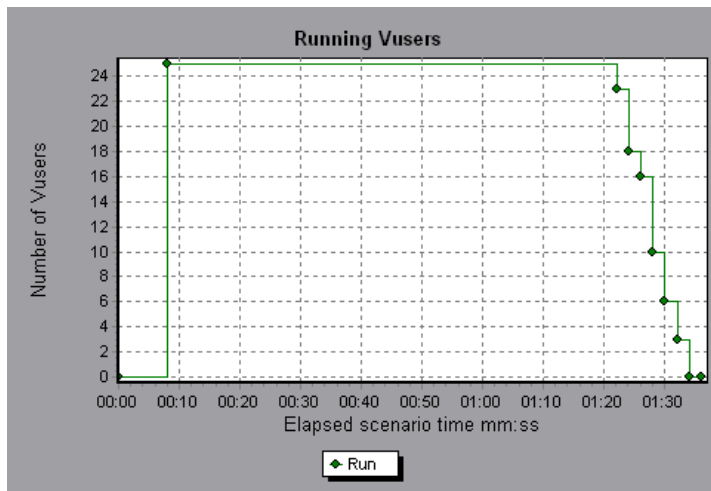
Vuser Summary Graph

The Vuser Summary graph displays a summary of Vuser performance. It lets you view the number of Vusers that successfully completed the scenario run relative to those that did not.

This graph may only be viewed as a pie.
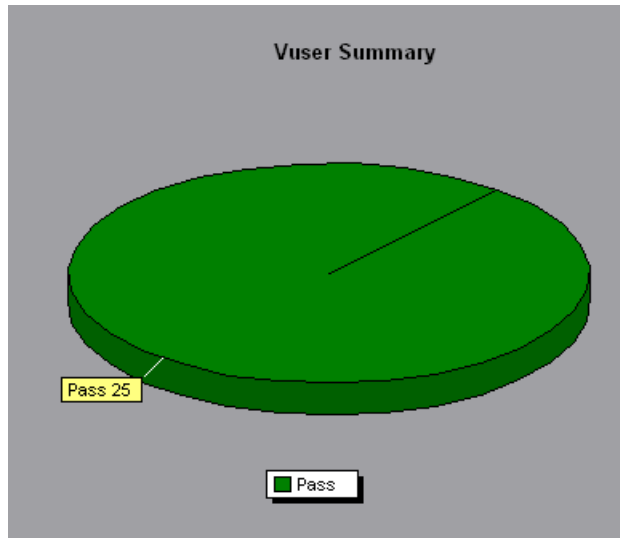
Rendezvous Graph

The Rendezvous graph indicates when Vusers were released from rendezvous points, and how many Vusers were released at each point.

This graph helps you understand transaction performance times. If you compare the Rendezvous graph to the Average Transaction Response Time graph, you can see how the load peak created by a rendezvous influences transaction times.

On the Rendezvous graph, the x-axis indicates the time that elapsed since the beginning of the scenario. The y-axis indicates the number of Vusers that were released from the rendezvous. If you set a rendezvous for 60 Vusers, and the graph indicates that only 25 were released, you can see that the rendezvous ended when the timeout expired because all of the Vusers did not arrive.
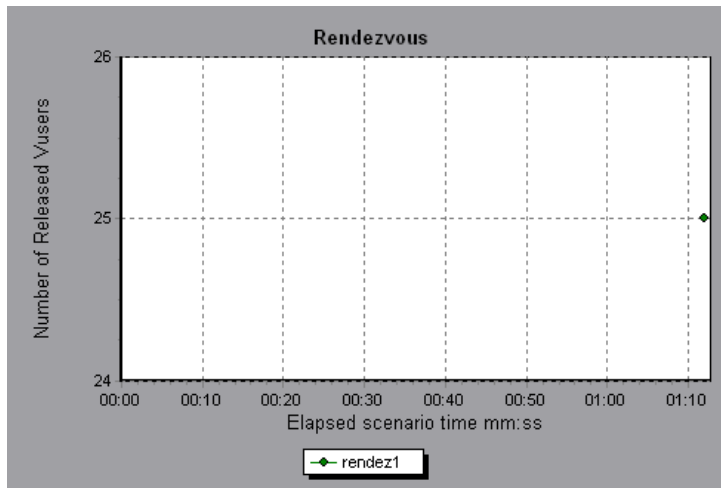
4

Error Graphs

After a scenario run, you can use the error graphs to analyze the errors that occurred during the load test.

This chapter describes:

➤ Error Statistics Graph

➤ Errors per Second Graph

About Error Graphs

During scenario execution, Vusers may not complete all transactions successfully. The Error graphs let you view information about the transactions that failed, stopped, or ended in errors. Using the Error graphs, you can view a summary of errors that occurred during the scenario and the average number of errors that occurred per second.

Error Statistics Graph

The Error Statistics graph displays the number of errors that accrued during scenario execution, grouped by error code.

In the graph below, out of a total of 178 errors that occurred during the scenario run, the second error code displayed in the legend occurred twelve times, comprising 6.74% the errors.

This graph may only be viewed as a pie.

Color	Scale	Measurement	Graph's Minimum	Graph's Average	Graph's Maximum	Graph's Median
⬛	1	Error -26627:-26627 Ac...	13	13	13	13
🟦	1	Error -26627:-26627 Ac...	12	12	12	12
🟪	1	Error -26627:-26627 Ac...	12	12	12	12
⬛	1	Error -26627:-26627 Ac...	3	3	3	3
🟩	1	Error -26627:-26627 Ac...	1	1	1	1
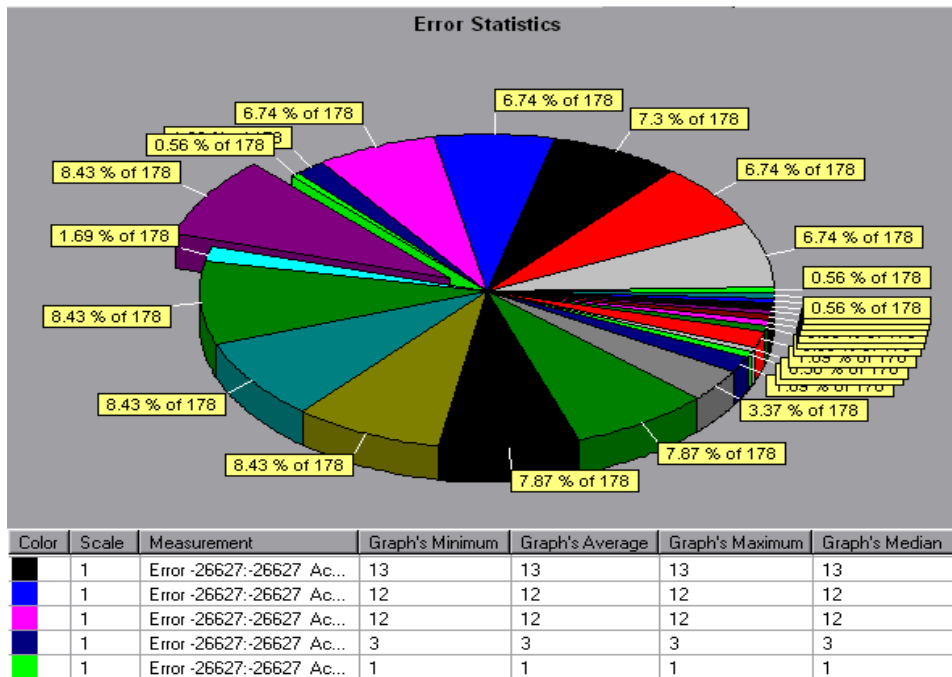
Errors per Second Graph

The Errors per Second graph displays the average number of errors that occurred during each second of the scenario run, grouped by error code.

The x-axis represents the elapsed time from the beginning of the scenario run. The y-axis represents the number of errors.

5
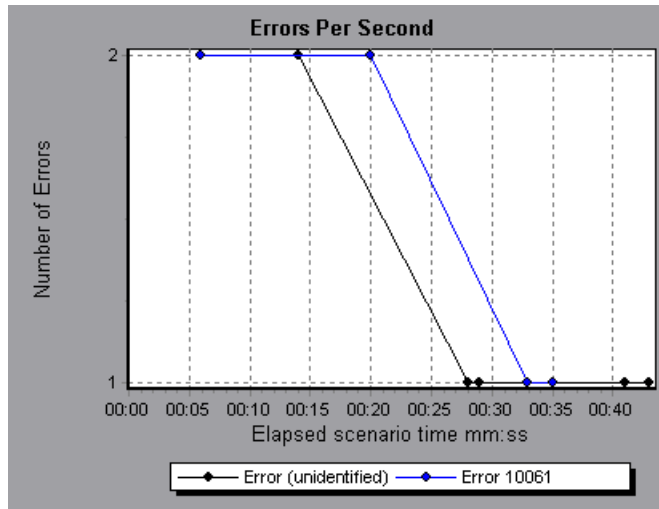
Transaction Graphs

After running a scenario, you can analyze the transactions that were executed during the test using one or more of the following graphs:

➤ Average Transaction Response Time Graph

➤ Transactions per Second Graph

➤ Total Transactions per Second

➤ Transaction Summary Graph

➤ Transaction Performance Summary Graph

➤ Transaction Response Time (Under Load) Graph

➤ Transaction Response Time (Percentile) Graph

➤ Transaction Response Time (Distribution) Graph

About Transaction Graphs

During scenario execution, Vusers generate data as they perform transactions. Analysis enables you to generate graphs that show the transaction performance and status throughout script execution.

You can use additional Analysis tools such as merging and crossing results to understand your transaction performance graphs. You can also sort the graph information by transactions. For more information about working with Analysis, see Chapter 2, "Working with Analysis Graphs."

Average Transaction Response Time Graph

The Average Transaction Response Time graph displays the average time taken to perform transactions during each second of the scenario run.

The x-axis represents the elapsed time from the beginning of the scenario run. The y-axis represents the average time (in seconds) taken to perform each transaction.
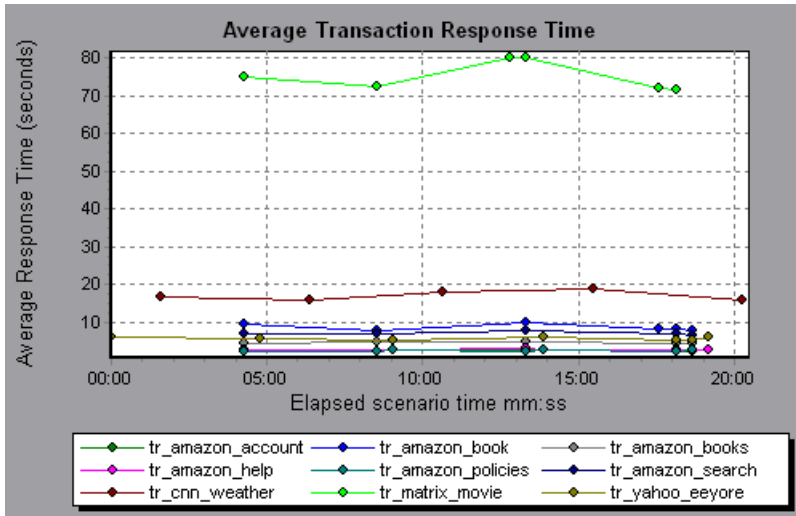
This graph is displayed differently for each granularity. The lower the granularity, the more detailed the results. However, it may be useful to view the results with a higher granularity to study the overall Vuser behavior throughout the scenario. For example, using a low granularity, you may see intervals when no transactions were performed. However, by viewing the same graph with a higher granularity, you will see the graph for the overall transaction response time. For more information on setting the granularity, see Chapter 2, "Working with Analysis Graphs."

Note: By default, only transactions that passed are displayed.

You can view a breakdown of a transaction in the Average Transaction Response Time graph by selecting **View > Show Transaction Breakdown Tree**, or right-clicking the transaction and selecting **Show Transaction Breakdown Tree**. In the Transaction Breakdown Tree, right-click the transaction you want to break down, and select **Break Down <transaction name>**. The Average Transaction Response Time graph displays data for the sub-transactions.

To view a breakdown of the Web page(s) included in a transaction or sub-transaction, right-click it and select **Web page breakdown for <transaction name>**. For more information on the Web Page Breakdown graphs, see Chapter 7, "Web Page Breakdown Graphs."

You can compare the Average Transaction Response Time graph to the Running Vusers graph to see how the number of running Vusers affects the transaction performance time.

For example, if the Average Transaction Response Time graph shows that performance time gradually improved, you can compare it to the Running Vusers graph to see whether the performance time improved due to a decrease in the Vuser load.

If you have defined acceptable minimum and maximum transaction performance times, you can use this graph to determine whether the performance of the server is within the acceptable range.

Understanding the Transaction Breakdown Tree View

The Transaction Breakdown Tree view displays a tree view of the transactions and sub-transactions in the current session.

To expand the view and drill down to the transactions and sub-transactions, click the "+" sign.

To collapse the view, click the "-" sign.

To view a breakdown of a transaction, right-click it and select **Break Down <transaction name>**. The Average Transaction Response Time or Transaction Performance Summary graph displays data for the sub-transactions. To view data only for the selected transaction/sub-transaction, right-click it and choose **Show Only <transaction name>**.

To view a breakdown of the Web page(s) included in a transaction or sub-transaction, right-click the transaction/sub-transaction, and select **Web page breakdown for <page name>**.

Note: The tree view pane is resizable.

Understanding the Web Page Breakdown Tree View

The Web Page Breakdown Tree view displays a tree view of the transactions, sub-transactions, and Web pages for which you can view Web Page Breakdown graphs. For more information about Web Page Breakdown graphs, see Chapter 7, "Web Page Breakdown Graphs."

To expand the view and drill down to the transactions, sub-transactions, Web pages, and page components, click the "+" sign.

To collapse the view, click the "-" sign.

Note that the tree view pane is resizable.

To view a breakdown of a transaction or sub-transaction, right-click it and select **Web Page Breakdown for <transaction name>**. The Web Page Breakdown graph opens in the right pane. To view a breakdown of a specific Web page, right-click the page and select **Break Down <page name>**.

Note: You can open a browser displaying the problematic page by right-clicking the page in the Web Page Breakdown tree and selecting **View page in browser**.

You can also open a Web Page Breakdown graph by choosing **Graph > Add Graph > Web Page Breakdown,** and selecting one of the Web Page Breakdown graphs.

Transactions per Second Graph

The Transactions per Second graph displays, for each transaction, the number of times it passed, failed, and stopped during each second of a scenario run. This graph helps you determine the actual transaction load on your system at any given moment. You can compare this graph to the Average Transaction Response Time graph in order to analyze the effect of the number of transactions on the performance time.

The x-axis represents the elapsed time from the beginning of the scenario run. The y-axis represents the number of transactions performed during the scenario run.
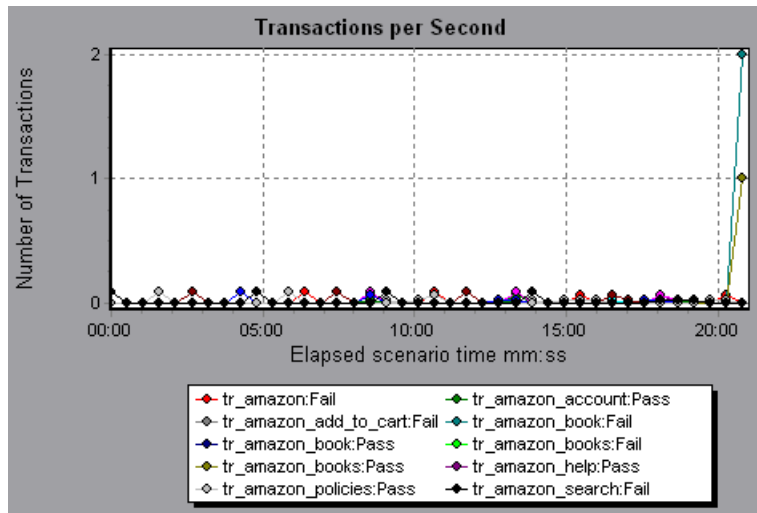
Total Transactions per Second

The Total Transactions per Second graph displays the total number of transactions that passed, the total number of transactions that failed, and the total number of transactions that were stopped, during each second of a scenario run.

The x-axis represents the elapsed time (in seconds) since the start of the scenario run. The y-axis represents the total number of transactions performed during the scenario run.
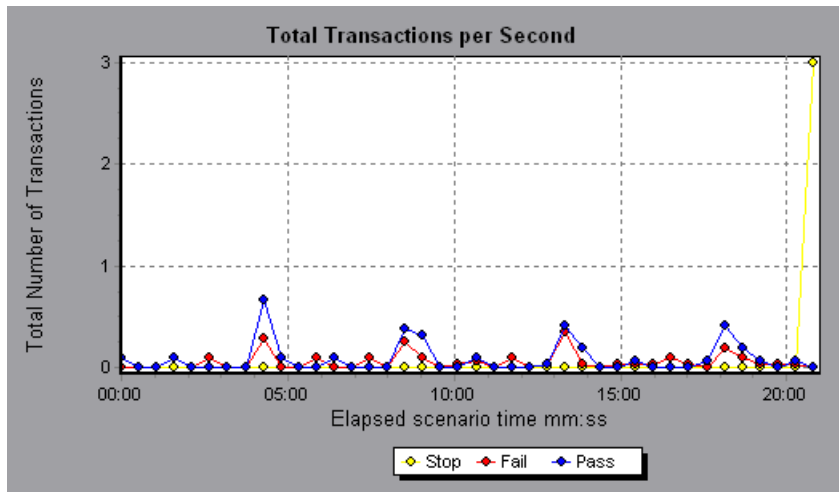
Transaction Summary Graph

The Transaction Summary graph summarizes the number of transactions in the scenario that failed, passed, stopped, and ended in error.

The x-axis specifies the name of the transaction. The y-axis shows the number of transactions performed during the scenario run.
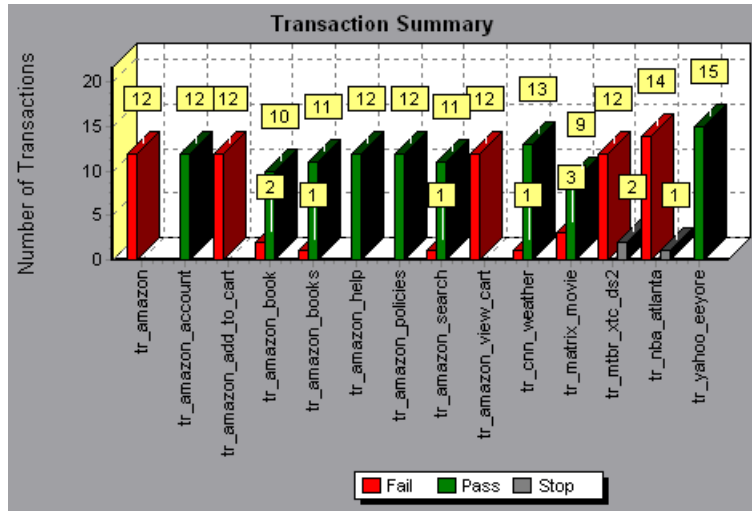
Transaction Performance Summary Graph

The Transaction Performance Summary graph displays the minimum, maximum and average performance time for all the transactions in the scenario.

The x-axis specifies the name of the transaction. The y-axis shows the time—rounded off to the nearest second—taken to perform each transaction.
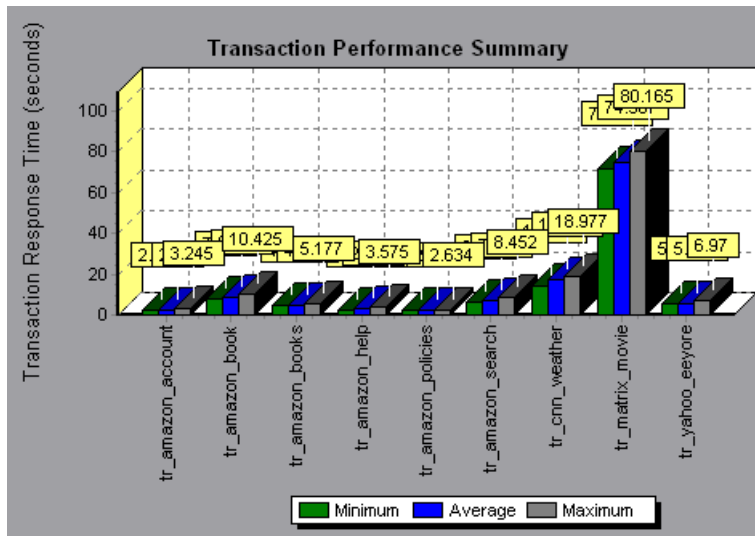
You can view a breakdown of a transaction in the Transaction Performance Summary graph by selecting **View** > **Show Transaction Breakdown Tree**, or right-clicking the transaction and selecting **Show Transaction Breakdown Tree**. In the Transaction Breakdown Tree, right-click the transaction you want to break down, and select **Break Down <transaction name>**. The Transaction Performance Summary graph displays data for the sub-transactions.

To view a breakdown of the Web page(s) included in a transaction or sub-transaction, right-click it and select **Web page breakdown for <transaction name>**. For more information on the Web Page Breakdown graphs, see Chapter 7, "Web Page Breakdown Graphs."

Transaction Response Time (Under Load) Graph

The Transaction Response Time (Under Load) graph is a combination of the Running Vusers and Average Transaction Response Time graphs and indicates transaction times relative to the number of Vusers running at any given point during the scenario. This graph helps you view the general impact of Vuser load on performance time and is most useful when analyzing a scenario with a gradual load. For information about creating a gradual load for a scenario, see the *LoadRunner Controller User's Guide*.

The x-axis indicates the number of running Vusers, and the y-axis indicates the average transaction time in seconds.
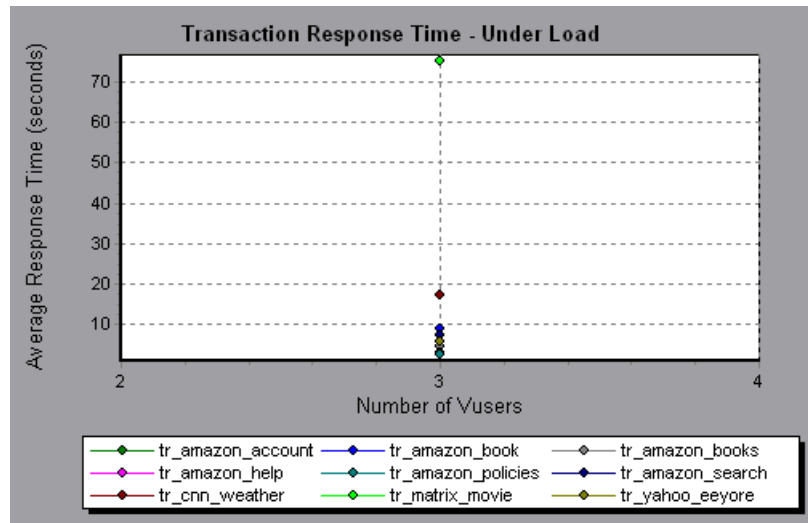
Transaction Response Time (Percentile) Graph

The Transaction Response Time (Percentile) graph analyzes the percentage of transactions that were performed within a given time range. This graph helps you determine the percentage of transactions that met the performance criteria defined for your system. In many instances, you need to determine the percent of transactions with an acceptable response time. The maximum response time may be exceptionally long, but if most transactions have acceptable response times, the overall system is suitable for your needs.

The x-axis represents the percentage of the total number of transactions measured during the scenario run. The y-axis represents the time taken to perform the transactions.

Note: Analysis approximates the transaction response time for each available percentage of transactions. The y-axis values, therefore, may not be exact.

In the following graph, fewer than 20 percent of the *tr_matrix_movie* transactions had a response time less than 70 seconds.
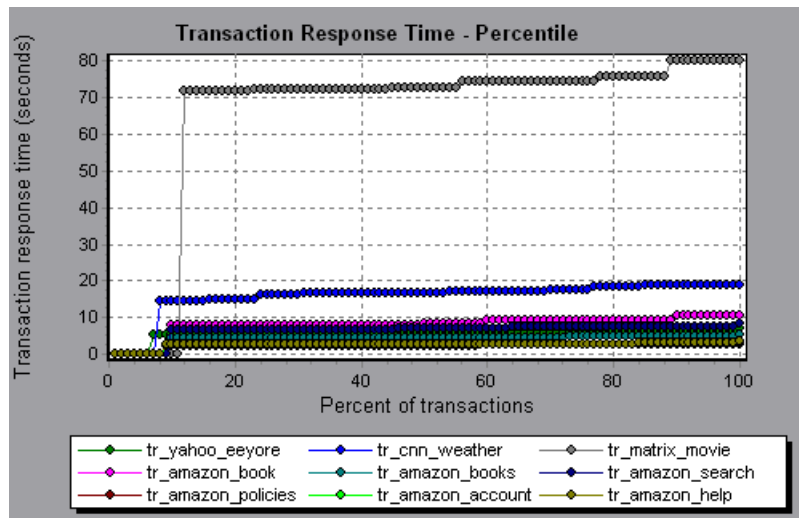
It is recommended to compare the Percentile graph to a graph indicating average response time such as the Average Transaction Response Time graph. A high response time for several transactions may raise the overall average. However, if the transactions with a high response time occurred less than five percent of the time, that factor may be insignificant.

Transaction Response Time (Distribution) Graph

The Transaction Response Time (Distribution) graph displays the distribution of the time taken to perform transactions in a scenario. If you compare it to the Transaction Performance Summary graph, you can see how the average performance was calculated.

The x-axis represents the transaction response time (rounded down to the nearest second). The y-axis represents the number of transactions executed during the scenario.

In the following graph, most of the transactions had a response time of less than 20 seconds.
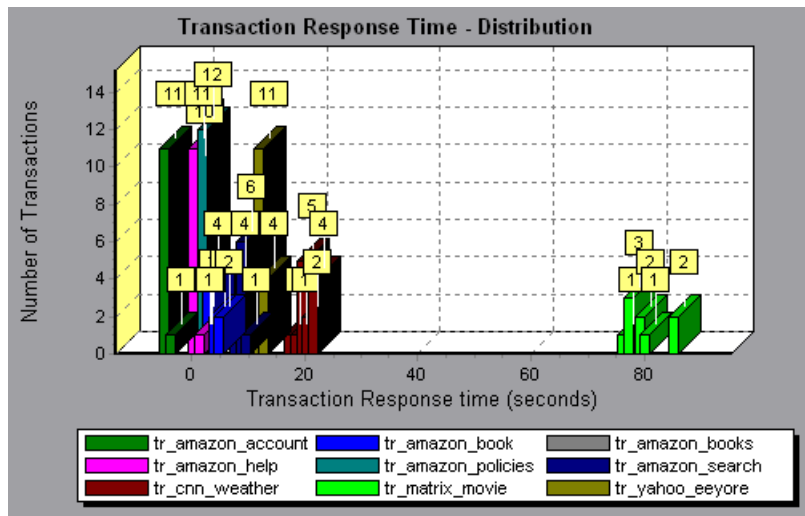
Note: This graph can only be displayed as a bar graph.

If you have defined acceptable minimum and maximum transaction performance times, you can use this graph to determine whether the performance of the server is within the acceptable range.

6

Web Resource Graphs

After a scenario run, you use the Web Resource graphs to analyze Web server performance.

This chapter describes:

➤ Hits per Second Graph

➤ Hits Summary Graph

➤ Throughput Graph

➤ Throughput Summary Graph

➤ HTTP Status Code Summary Graph

➤ HTTP Responses per Second Graph

➤ Pages Downloaded per Second Graph

➤ Retries per Second Graph

➤ Retries Summary Graph

➤ Connections Graph

➤ Connections per Second Graph

➤ SSL Connections per Second Graph

About Web Resource Graphs

Web Resource graphs provide you with information about the performance of your Web server. You use the Web Resource graphs to analyze the throughput on the Web server, the number of hits per second, the number of HTTP responses per second, the HTTP status codes returned from the Web server, the number of downloaded pages per second, the number of server retries per second, a summary of the server retries during the scenario, the number of open TCP/IP connections, the number of new TCP/IP connections opened and the number of connections that are shut down, and the number of new and reused SSL connections opened.

Hits per Second Graph

The Hits per Second graph shows the number of HTTP requests made by Vusers to the Web server during each second of the scenario run. This graph helps you evaluate the amount of load Vusers generate, in terms of the number of hits. You can compare this graph to the Average Transaction Response Time graph to see how the number of hits affects transaction performance.
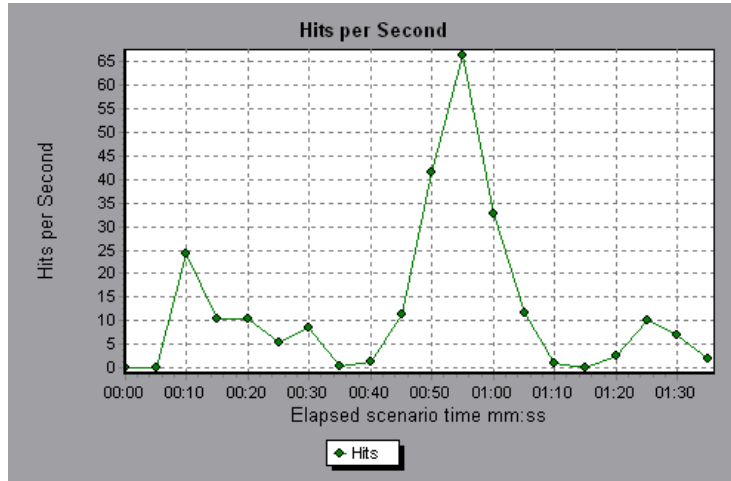
The x-axis represents the elapsed time since the start of the scenario run. The y-axis represents the number of hits on the server. For example, the graph above shows that the most hits per second took place during the fifty-fifth second of the scenario.

Note: You cannot change the granularity of the x-axis to a value that is less than the Web granularity you defined in the General tab of the Options dialog box.

Hits Summary Graph

The Hits Summary graph shows a pie chart of the number of HTTP requests made by Vusers to the Web server during the scenario run. The graph first appears with a 100% pie segment showing the total number of hits. However, you can further divide the graph into groups using the **Set Filter/Group By** utility:
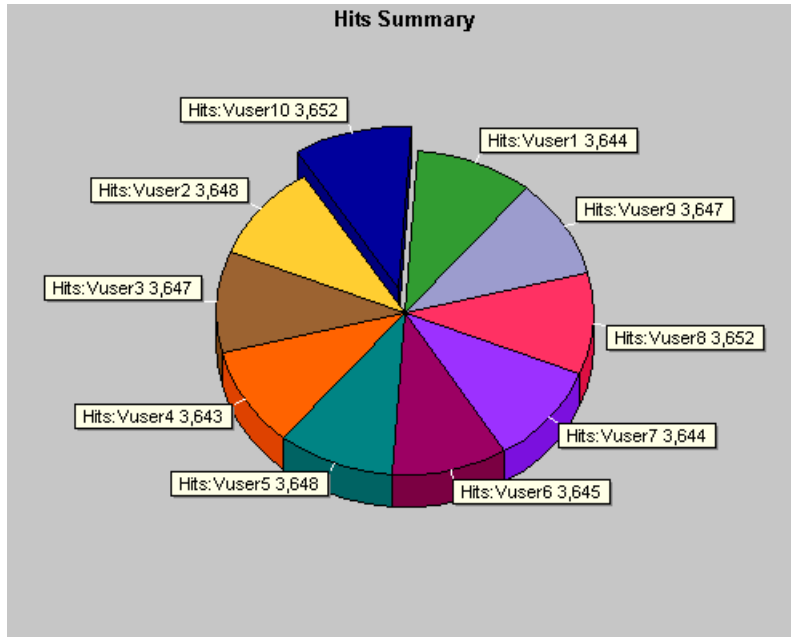
The graph above is the transformation of a Hits Summary graph after being grouped by VuserID. It shows the number of hits made by each Vuser.

Throughput Graph

The Throughput graph shows the amount of throughput on the server during each second of the scenario run. Throughput is measured in bytes and represents the amount of data that the Vusers received from the server at any given second. This graph helps you evaluate the amount of load Vusers generate, in terms of server throughput. You can compare this graph to the Average Transaction Response Time graph to see how the throughput affects transaction performance.

The x-axis represents the elapsed time since the start of the scenario run. The y-axis represents the throughput of the server, in bytes.

The following graph shows the highest throughput to be 193,242 bytes during the fifty-fifth second of the scenario.
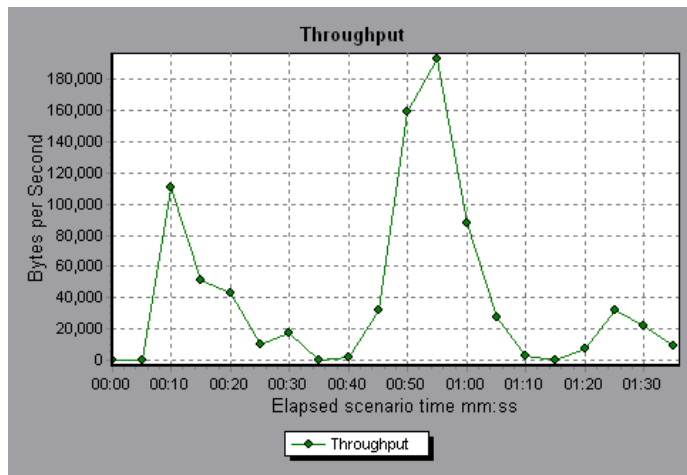
Note: You cannot change the granularity of the x-axis to a value that is less than the Web granularity you defined in the General tab of the Options dialog box.

Throughput Summary Graph

The Throughput Summary graph shows a pie chart of the amount of throughput on a Web server during the scenario run. The graph first appears with a 100% pie segment showing the total amount of throughput. However, you can further divide the graph into groups using the **Set Filter/Group By** utility:

Throughput Summary

Throughput:Vuser10 7,857,328

Throughput:Vuser2 7,884,017

Throughput:Vuser1 7,848,589

Throughput:Vuser9 7,848,62

Throughput:Vuser3 7,816,104

Throughput:Vuser8 7,84

Throughput:Vuser4 7,828,299

Throughput:Vuser7 7,860,165

Throughput:Vuser5 7,825,150

Throughput:Vuser6 7,853,670
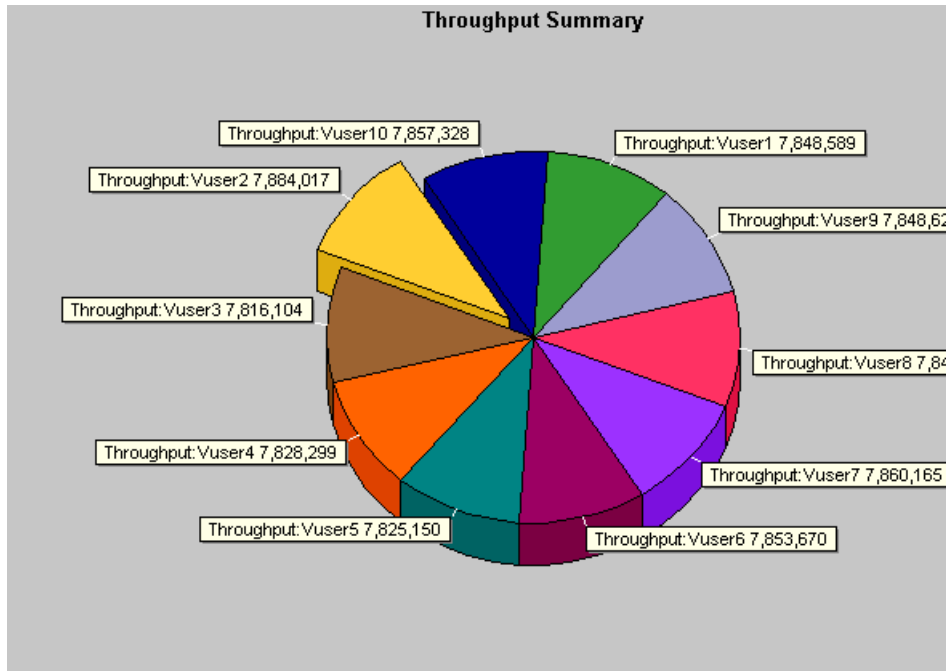
The graph above is the transformation of a Throughput Summary graph after being grouped by VuserID. It shows the amount of throughput generated by each Vuser.

HTTP Status Code Summary Graph

The HTTP Status Code Summary Graph shows the number of HTTP status codes (which indicate the status of HTTP requests, for example, "the request was successful," "the page was not found") returned from the Web server during the scenario run, grouped by status code. Use this graph together with the HTTP Responses per Second Graph to locate those scripts which generated error codes.

This graph can only be viewed as a pie.

The following graph shows that only the HTTP status codes **200** and **302** were generated. Status code **200** was generated 1,100 times, and status code **302** was generated 125 times.
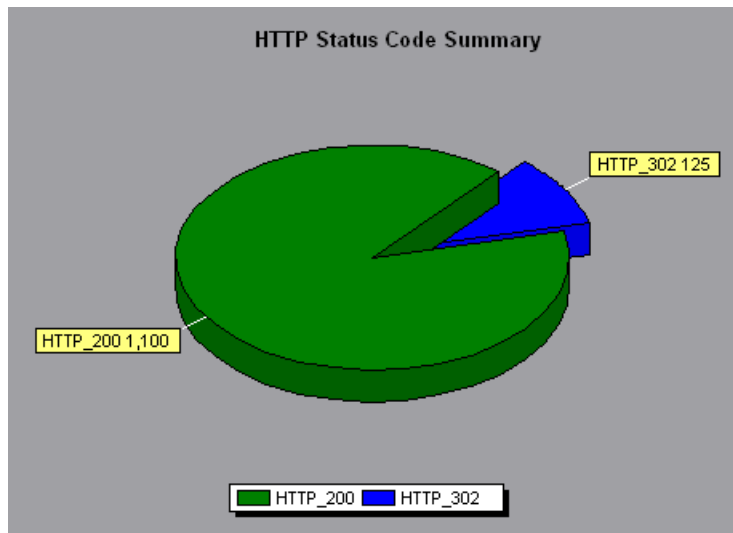
HTTP Responses per Second Graph

The HTTP Responses per Second graph shows the number of HTTP status codes (which indicate the status of HTTP requests, for example, "the request was successful," "the page was not found") returned from the Web server during each second of the scenario run, grouped by status code. You can group the results shown in this graph by script (using the "Group By" function) to locate scripts which generated error codes. For more information on the "Group By" function, see Chapter 2, "Working with Analysis Graphs."

The x-axis represents the time that has elapsed since the start of the scenario run. The y-axis represents the number of HTTP responses per second.

The following graph shows that the greatest number of **200** status codes, 60, was generated in the fifty-fifth second of the scenario run. The greatest number of **302** codes, 8.5, was generated in the fiftieth second of the scenario run.
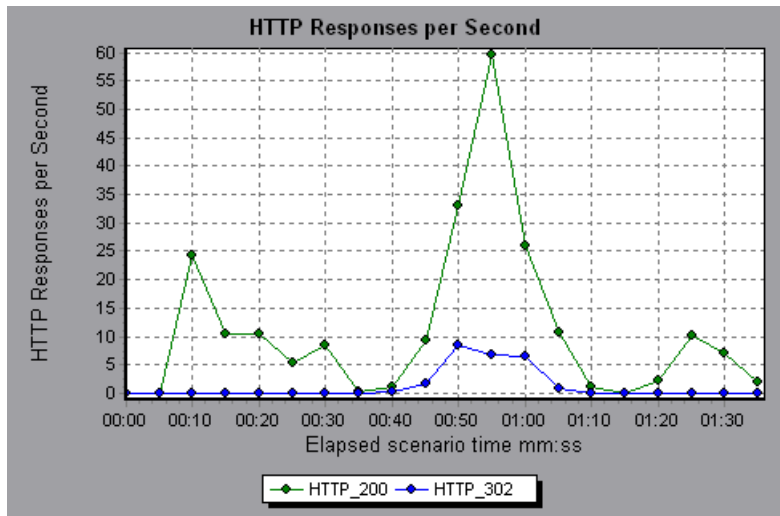
The following table displays a list of HTTP status codes:

Code	Description
200	OK
201	Created
202	Accepted
203	Non-Authoritative Information
204	No Content
205	Reset Content
206	Partial Content
300	Multiple Choices
301	Moved Permanently
302	Found
303	See Other
304	Not Modified
305	Use Proxy
307	Temporary Redirect
400	Bad Request
401	Unauthorized
402	Payment Required
403	Forbidden
404	Not Found
405	Method Not Allowed
406	Not Acceptable
407	Proxy Authentication Required
408	Request Timeout
409	Conflict

Code	Description
410	Gone
411	Length Required
412	Precondition Failed
413	Request Entity Too Large
414	Request - URI Too Large
415	Unsupported Media Type
416	Requested range not satisfiable
417	Expectation Failed
500	Internal Server Error
501	Not Implemented
502	Bad Gateway
503	Service Unavailable
504	Gateway Timeout
505	HTTP Version not supported

For more information on the above status codes and their descriptions, see http://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html#sec10.

Pages Downloaded per Second Graph

The Pages Downloaded per Second graph shows the number of Web pages (y-axis) downloaded from the server during each second of the scenario run (x-axis). This graph helps you evaluate the amount of load Vusers generate, in terms of the number of pages downloaded.

The following graph shows that the greatest number of pages downloaded per second, about 7, occurred in the fiftieth second of the scenario run.
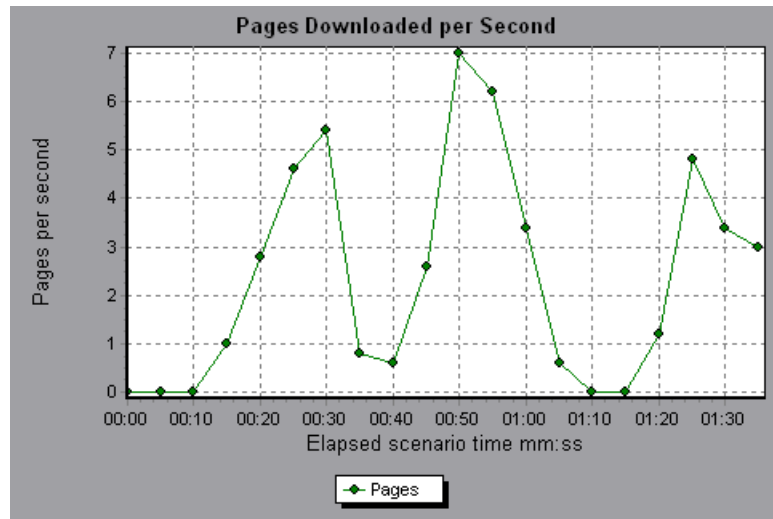
Like throughput, downloaded pages per second is a representation of the amount of data that the Vusers received from the server at any given second. However, the Throughput graph takes into account each resource and its size (for example, the size of each .gif file, the size of each Web page). The Pages Downloaded per Second graph takes into account only the number of pages.

Note: To view the Pages Downloaded per Second graph, you must select **Pages per second (HMTL Mode only)** from the run time settings Preferences tab before running your scenario.

In the following example, the Throughput graph is merged with the Pages Downloaded per Second graph. It is apparent from the graph that throughput is not completely proportional to the number of pages downloaded per second. For example, between 10 and 25 seconds into the scenario run, the number of pages downloaded per second increased while the throughput decreased.
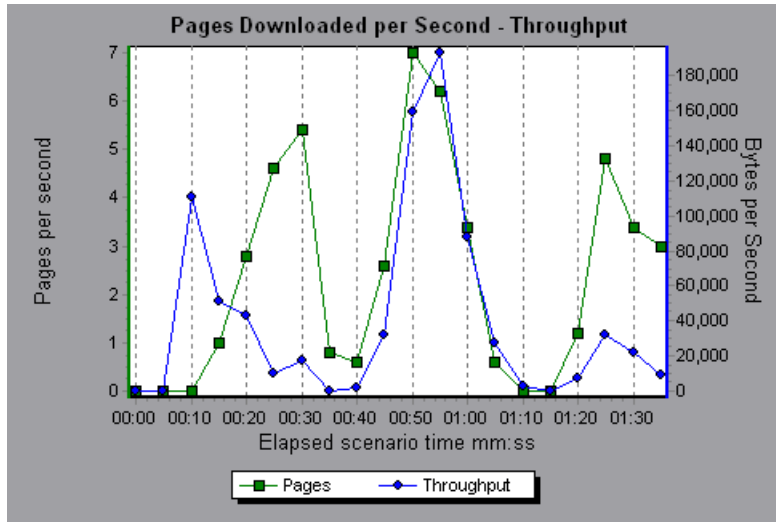
Retries per Second Graph

The Retries per Second graph displays the number of attempted server connections during each second of the scenario run. A server connection is retried when the initial connection was unauthorized, when proxy authentication is required, when the initial connection was closed by the server, when the initial connection to the server could not be made, or when the server was initially unable to resolve the load generator's IP address.

The x-axis displays the time that has elapsed since the start of the scenario run. The y-axis displays the number of server retries per second.

The following graph shows that during the first second of the scenario, the number of retries was 0.4, whereas in the fifth second of the scenario, the number of retries per second rose to 0.8.

103
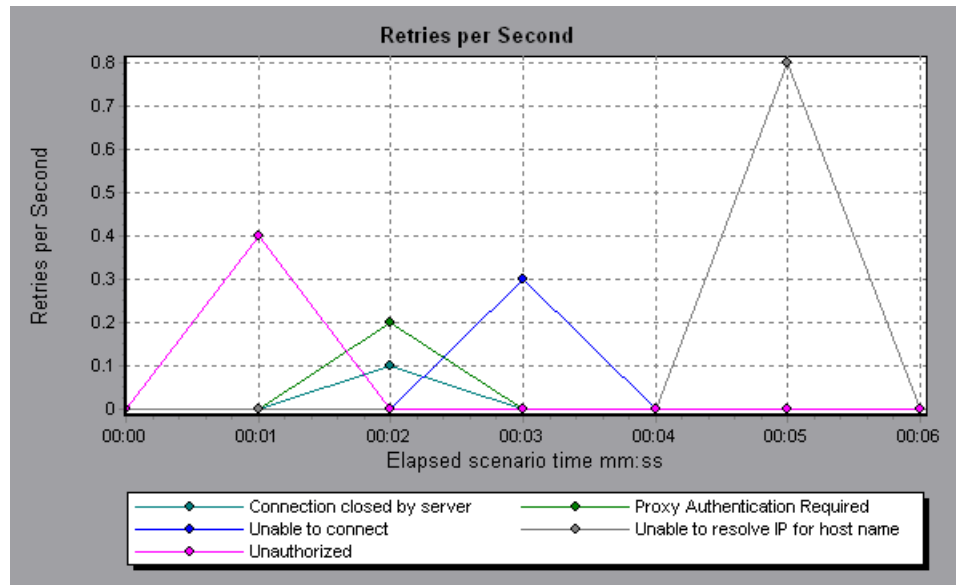
Retries Summary Graph

The Retries Summary Graph shows the number of attempted server connections during the scenario run, grouped by the cause of the retry. Use this graph together with the Retries per Second Graph to determine at what point during the scenario the server retries were attempted.

This graph may only be viewed as a pie.

The following graph shows that the server's inability to resolve the load generator's IP address was the leading cause of server retries during the scenario run.
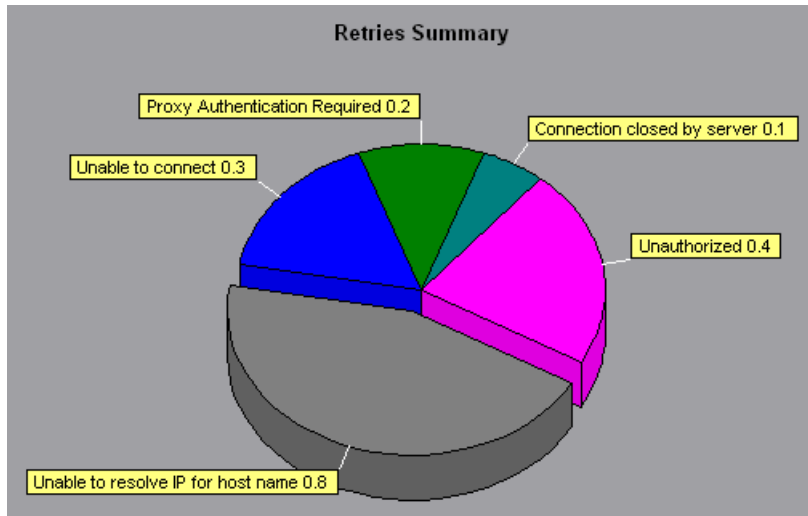
Connections Graph

The **Connections** graph shows the number of open TCP/IP connections (y-axis) at each point in time of the scenario (x-axis). Note that one HTML page may cause the browser to open several connections, when links on the page go to different Web addresses. Two connections are opened for each Web server.

This graph is useful in indicating when additional connections are needed. For example, if the number of connections reaches a plateau, and the transaction response time increases sharply, adding connections would probably cause a dramatic improvement in performance (reduction in the transaction response time).
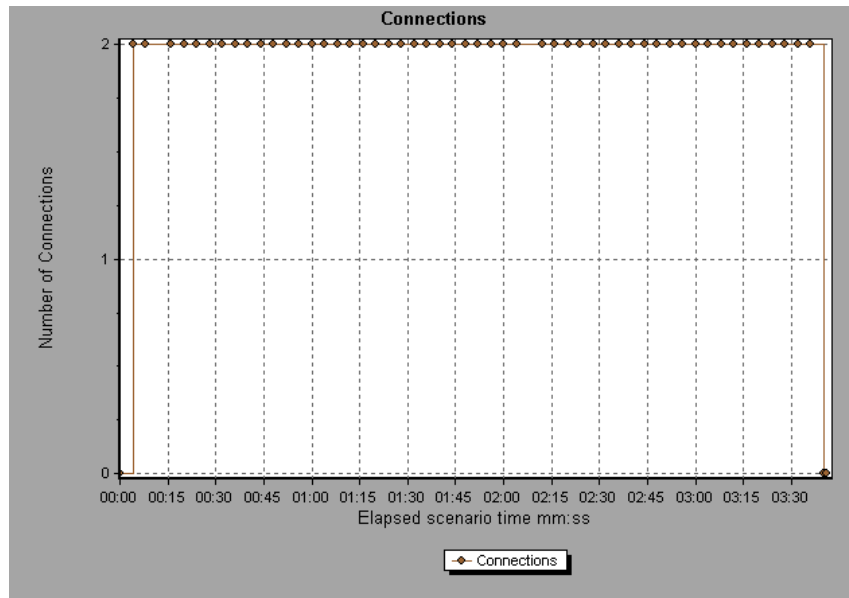
Connections per Second Graph

The **Connections Per Second** graph shows the number of new TCP/IP connections (y-axis) opened and the number of connections that are shut down for each second of the scenario (x-axis).

The number of new connections should be a small fraction of the number of hits per second, because new TCP/IP connections are very expensive in terms of server, router and network resource consumption. Ideally, many HTTP requests should use the same connection, instead of opening a new connection for each request.
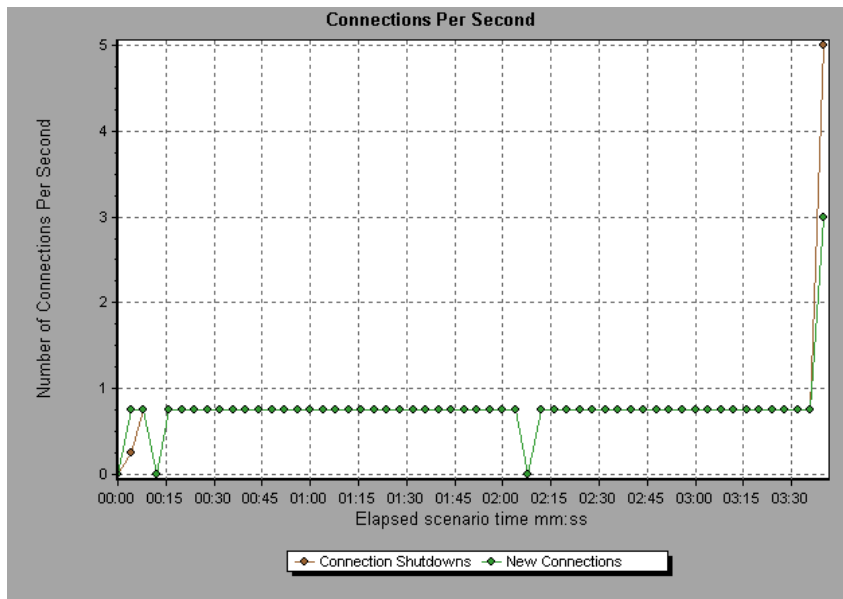
SSL Connections per Second Graph

The **SSL Connections per Second** graph shows the number of new and reused SSL Connections (y-axis) opened in each second of the scenario (x-axis). An SSL connection is opened by the browser after a TCP/IP connection has been opened to a secure server.
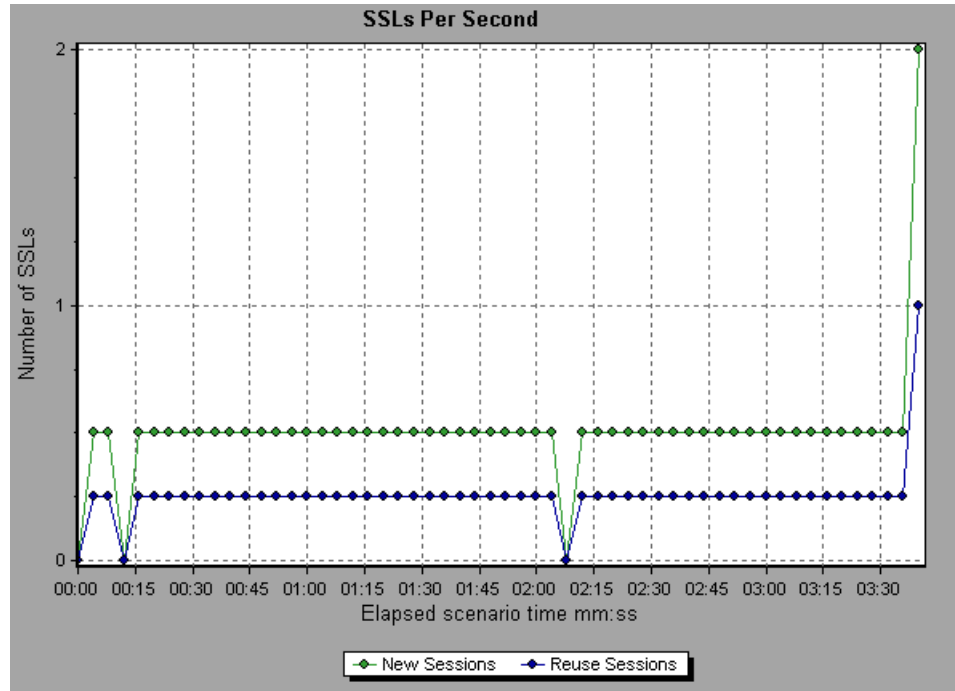
Because creating a new SSL connection entails heavy resource consumption, you should try to open as few new SSL connections as possible; once you've established an SSL connection, you should reuse it. There should be no more than one new SSL connection per Vuser. If you set your run-time settings to simulate a new Vuser at each iteration (via the Browser Emulation tab in the Run-Time Settings menu), you should have no more than one new SSL connection per Vuser per iteration. Ideally, you should have very few new TCP/IP and SSL connections each second.

7

Web Page Breakdown Graphs

The Web Page Breakdown graphs enable you to assess whether transaction response times were affected by page content. Using the Web Page Breakdown graphs, you can analyze problematic elements—for example, images that download slowly, or broken links—of a Web site.

This chapter describes:

➤ Activating the Web Page Breakdown Graphs

➤ Page Component Breakdown Graph

➤ Page Component Breakdown (Over Time) Graph

➤ Page Download Time Breakdown Graph

➤ Page Download Time Breakdown (Over Time) Graph

➤ Time to First Buffer Breakdown Graph

➤ Time to First Buffer Breakdown (Over Time) Graph

➤ Downloaded Component Size Graph

About Web Page Breakdown Graphs

Web Page Breakdown graphs provide you with performance information for each monitored Web page in your script. You can view the download time of each page in the script and its components, and identify at what point during download time problems occurred. In addition, you can view the relative download time and size of each page and its components. Analysis displays both average download time and download time over time data.

You correlate the data in the Web Page Breakdown graphs with data in the Transaction Performance Summary and Average Transaction Response Time graphs in order to analyze why and where problems are occurring, and whether the problems are network- or server-related.

Note: Because server time is being measured from the client, network time may influence this measurement if there is a change in network performance from the time the initial HTTP request is sent until the time the first buffer is sent. The server time displayed, therefore, is estimated server time and may be slightly inaccurate.

You begin analyzing these graphs with the Web Page Breakdown graph, which displays the average download time (in seconds) for each monitored Web page during each second of the run. The x-axis represents the elapsed time from the beginning of the run. The y-axis represents the average download time (in seconds) for each Web page.

In order for Analysis to generate Web Page Breakdown graphs, you must enable the component breakdown feature before recording your script.

To enable the component breakdown feature:

1 From the Controller menu, choose **Tools** > **Options**.

2 Select the **Web Page Breakdown** tab. Select the **Enable Web Page Breakdown** check box.

Note: It is recommended that, in VuGen, you select **HTML-based script** in the Recording tab of the Recording Options dialog box.

For more information on recording Web Vuser scripts, see the *Creating Vuser Scripts* guide.

Activating the Web Page Breakdown Graphs

The Web Page Breakdown graphs are most commonly used to analyze a problem detected in the Transaction Performance Summary or Average Transaction Response Time graphs. For example, the Average Transaction Response Time graph below demonstrates that the average transaction response time for the trans1 transaction was high.
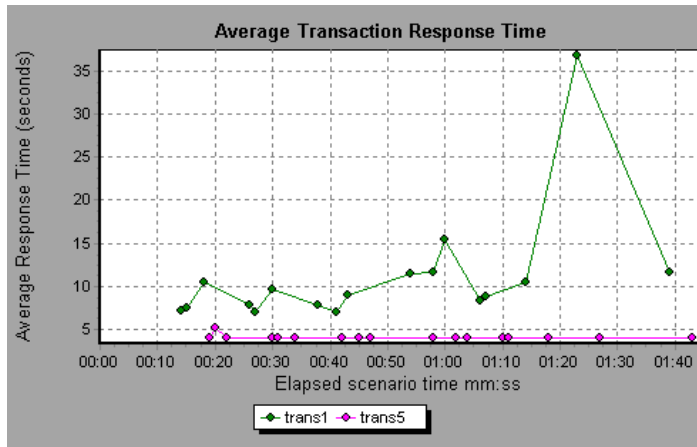
Using the Web Page Breakdown graphs, you can pinpoint the cause of the delay in response time for the trans1 transaction.

To view a breakdown of a transaction:

1 Right-click trans1 and select **Web Page Breakdown for trans1**. The Web Page Breakdown graph and the Web Page Breakdown tree appear. An icon appears next to the page name indicating the page content. See "Web Page Breakdown Content Icons" on page 114.

2 In the Web Page Breakdown tree, right-click the problematic page you want to break down, and select **Break Down <component name>**. Alternatively, select a page in the Select Page to Break Down box. The Web Page Breakdown graph for that page appears.

Note: You can open a browser displaying the problematic page by right-clicking the page in the Web Page Breakdown tree and selecting **View page in browser**.

---

3 Select one of the following available options:

➤ **Download Time Breakdown:** Displays a table with a breakdown of the selected page's download time. The size of each page component (including the component's header) is displayed. See the Page Download Time Breakdown Graph for more information about this display.

➤ **Component Breakdown (Over Time):** Displays the Page Component Breakdown (Over Time) Graph for the selected Web page.

➤ **Download Time Breakdown (Over Time):** Displays the Page Download Time Breakdown (Over Time) Graph for the selected Web page.

➤ **Time to First Buffer Breakdown (Over Time):** Displays the Time to First Buffer Breakdown (Over Time) Graph for the selected Web page.

To display the graphs in full view, click the **Open graph in full view** button. Note that you can also access these graphs, as well as additional Web Page Breakdown graphs, from the Open a New Graph dialog box.

Web Page Breakdown Content Icons

The following icons appear in the Web Page breakdown tree. They indicate the HTTP content of the page.

Transaction: Specifies that the ensuing content is part of the transaction.

Page content: Specifies that the ensuing content, which may include text, images etc, is all part of one logical page.

Text content: Textual information. Plain text is intended to be displayed as-is. Includes HTML text and stylesheets.

Multipart content: Data consisting of multiple entities of independent data types.

Message content: An encapsulated message. Common subtypes are news, or external-body which for specifies large bodies by reference to an external data source

Application content: Some other kind of data, typically either uninterpreted binary data or information to be processed by an application. An example subtype is Postscript data.

Image content: Image data. Two common subtypes are the jpeg and gif format

Video content: A time-varying picture image. A common subtype is the mpeg format.

Resource content: Other resources not listed above. Also, content that is defined as "not available" are likewise included.

Page Component Breakdown Graph

The Page Component Breakdown graph displays the average download time (in seconds) for each Web page and its components.

To isolate problematic components, it may be helpful to sort the legend according to the average number of seconds taken to download a component. To sort the legend by average, click the **Graph's Average** column.

The following graph demonstrates that the main cnn.com URL took 28.64% of the total download time, compared to 35.67% for the www.cnn.com/WEATHER component.
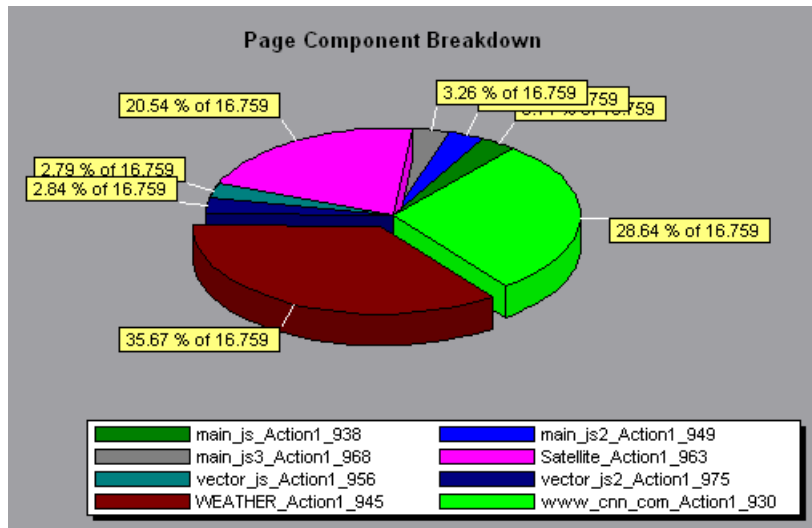
The graph can only be viewed as a pie.

To ascertain which components caused the delay in download time, you can break down the problematic URL by double-clicking it in the Web Page Breakdown tree. In the following example, the cnn.com/WEATHER component is broken down.
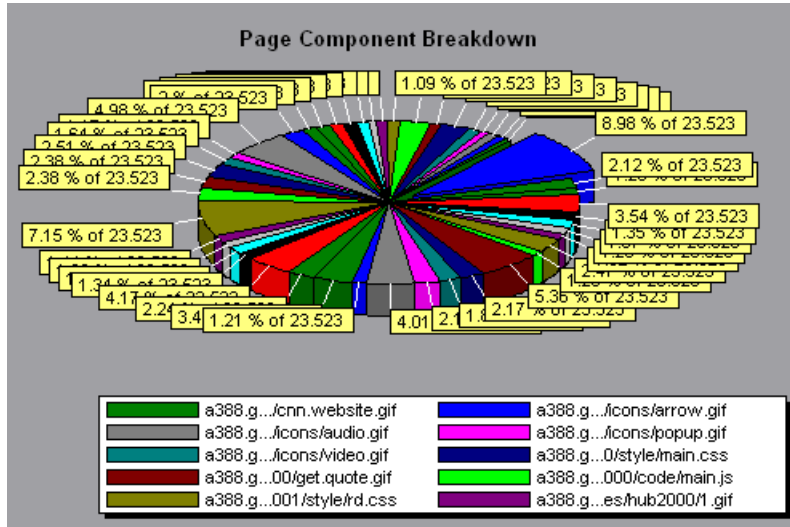
The above graph shows that the main cnn.com/WEATHER component took the longest time to download (8.98% of the total download time). To isolate other problematic components of the URL, it may be helpful to sort the legend according to the average number of seconds taken to download a component. To sort the legend by average, click the **Graph's Average** column heading.

Page Component Breakdown (Over Time) Graph

The Page Component Breakdown (Over Time) graph displays the average response time (in seconds) for each Web page and its components during each second of the scenario run.

The x-axis represents the elapsed time from the beginning of the scenario run. The y-axis represents the average response time (in seconds) for each component.

For example, the following graph demonstrates that the response time for Satellite_Action1_963 was significantly greater, throughout the scenario, than the response time for main_js_Action1_938.
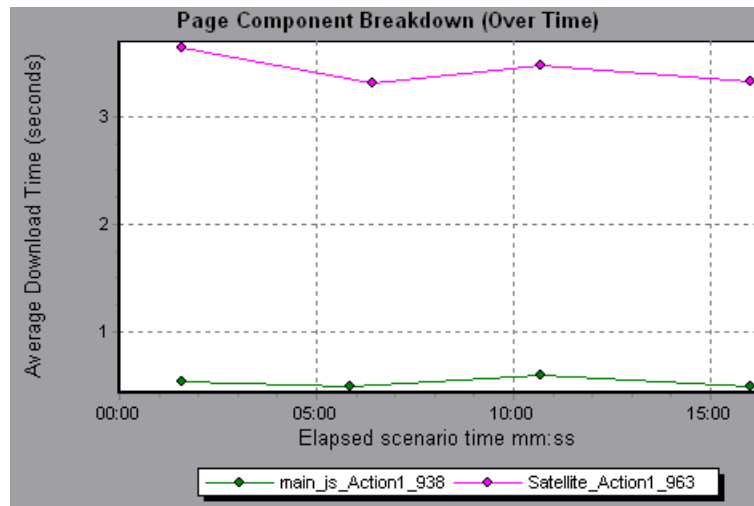
To ascertain which components were responsible for the delay in response time, you can break down the problematic component by double-clicking it in the Web Page Breakdown tree.
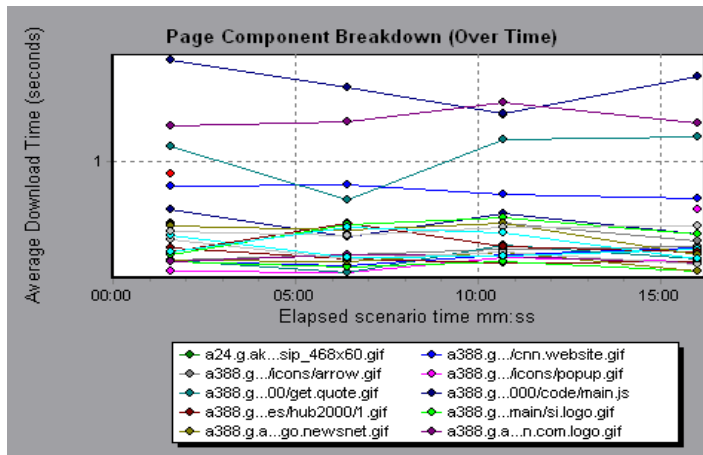
Using the above graph, you can track which components of the main component were most problematic, and at which point(s) during the scenario the problem(s) occurred. To isolate the most problematic components, it may be helpful to sort the legend tab according to the average number of seconds taken to download a component. To sort the legend by average, double-click the **Average** column heading.

To identify a component in the graph, you can select it. The corresponding line in the legend tab is selected.

Page Download Time Breakdown Graph

The Page Download Time Breakdown graph displays a breakdown of each page component's download time, enabling you to determine whether slow response times are being caused by network or server errors during Web page download.
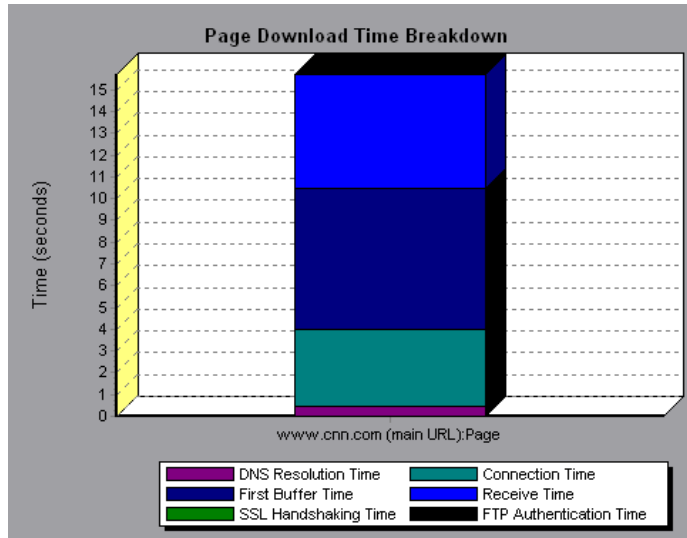
The Page Download Time Breakdown graph breaks down each component by DNS resolution time, connection time, time to first buffer, SSL handshaking time, receive time, FTP authentication time, client time, and error time.

These breakdowns are described below:

Name	Description
DNS Resolution	Displays the amount of time needed to resolve the DNS name to an IP address, using the closest DNS server. The DNS Lookup measurement is a good indicator of problems in DNS resolution, or problems with the DNS server.
Connection	Displays the amount of time needed to establish an initial connection with the Web server hosting the specified URL. The connection measurement is a good indicator of problems along the network. It also indicates whether the server is responsive to requests.
First Buffer	Displays the amount of time that passes from the initial HTTP request (usually GET) until the first buffer is successfully received back from the Web server. The first buffer measurement is a good indicator of Web server delay as well as network latency. **Note:** Since the buffer size may be up to 8K, the first buffer might also be the time it takes to completely download the element.
SSL Handshaking	Displays the amount of time taken to establish an SSL connection (includes the client hello, server hello, client public key transfer, server certificate transfer, and other—partially optional—stages). After this point, all the communication between the client and server is encrypted. The SSL Handshaking measurement is only applicable for HTTPS communications.
Receive	Displays the amount of time that passes until the last byte arrives from the server and the downloading is complete. The Receive measurement is a good indicator of network quality (look at the time/size ratio to calculate receive rate).

Name	Description
FTP Authentication	Displays the time taken to authenticate the client. With FTP, a server must authenticate a client before it starts processing the client's commands. The FTP Authentication measurement is only applicable for FTP protocol communications.
Client Time	Displays the average amount of time that passes while a request is delayed on the client machine due to browser think time or other client-related delays.
Error Time	Displays the average amount of time that passes from the moment an HTTP request is sent until the moment an error message (HTTP errors only) is returned.

Note: Each measurement displayed on the page level is the sum of that measurement recorded for each page component. For example, the Connection Time for www.cnn.com is the sum of the Connection Time for each of the page's components.

The above graph demonstrates that receive time, connection time, and first buffer time accounted for a large portion of the time taken to download the main cnn.com URL.

If you break the cnn.com URL down further, you can isolate the components with the longest download time, and analyze the network or server problems that contributed to the delay in response time.
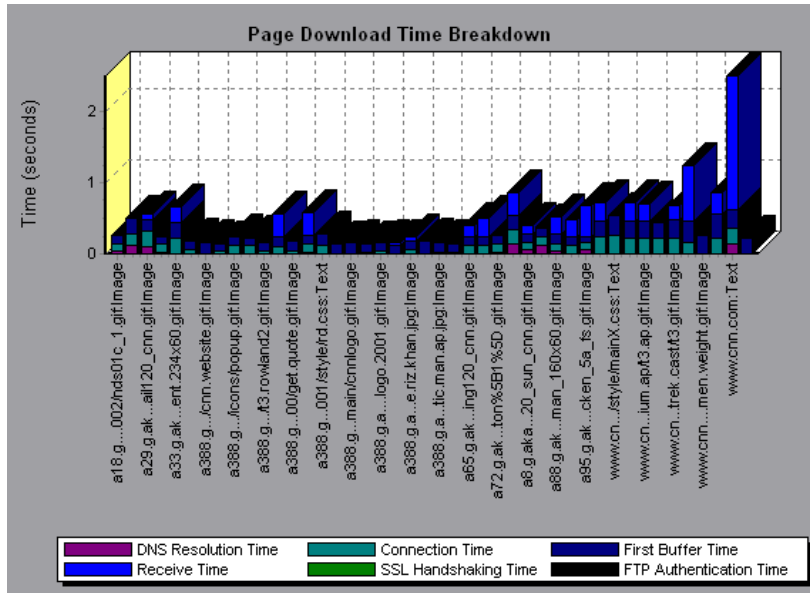
Breaking down the cnn.com URL demonstrates that for the component with the longest download time (the www.cnn.com component), the receive time accounted for a large portion of the download time.

Page Download Time Breakdown (Over Time) Graph

The Page Download Time Breakdown (Over Time) graph displays a breakdown of each page component's download time during each second of the scenario run.

The x-axis represents the elapsed time from the beginning of the scenario run. The y-axis represents the time (in seconds) taken for each step in the download process.

This graph enables you to determine at what point during scenario execution network or server problems occurred.
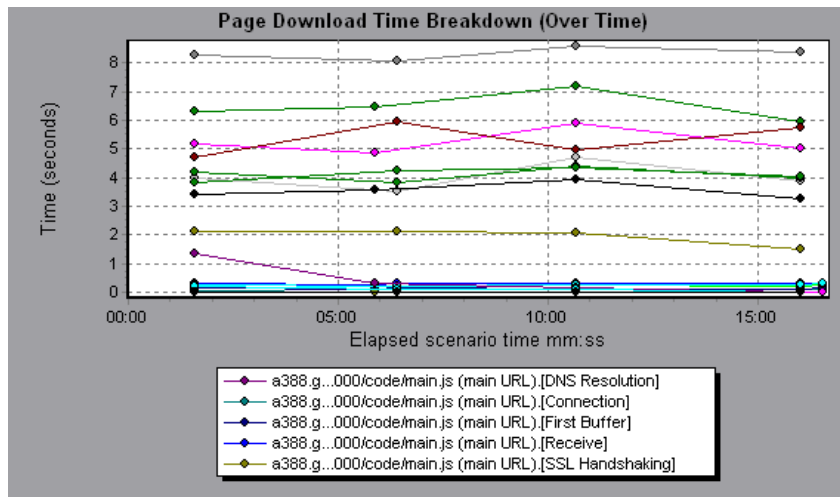
Note: Each measurement displayed on the page level is the sum of that measurement recorded for each page component. For example, the Connection Time for www.cnn.com is the sum of the Connection Time for each of the page's components.

To isolate the most problematic components, you can sort the legend tab according to the average number of seconds taken to download a component. To sort the legend by average, double-click the **Average** column heading.

To identify a component in the graph, select it. The corresponding line in the legend tab is selected.

In the example in the previous section, it is apparent that cnn.com was the most problematic component. If you examine the cnn.com component, the Page Download Time Breakdown (Over Time) graph demonstrates that first buffer and receive time remained high throughout the scenario, and that DNS Resolution time decreased during the scenario.
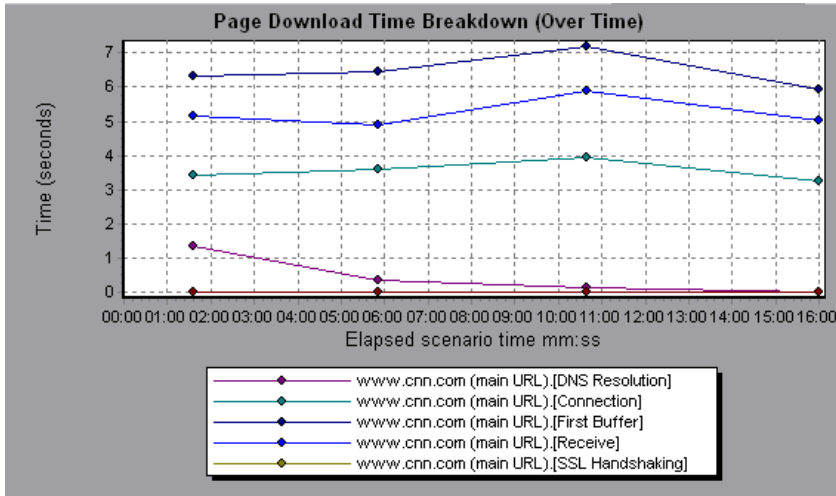
Note: When the Page Download Time Breakdown (Over Time) graph is selected from the Web Page Breakdown graph, it appears as an area graph.

Time to First Buffer Breakdown Graph

The Time to First Buffer Breakdown graph displays each Web page component's relative server/network time (in seconds) for the period of time until the first buffer is successfully received back from the Web server. If the download time for a component is high, you can use this graph to determine whether the problem is server- or network-related

The x-axis specifies the name of the component. The y-axis shows the average network/server time (in seconds) for each component.
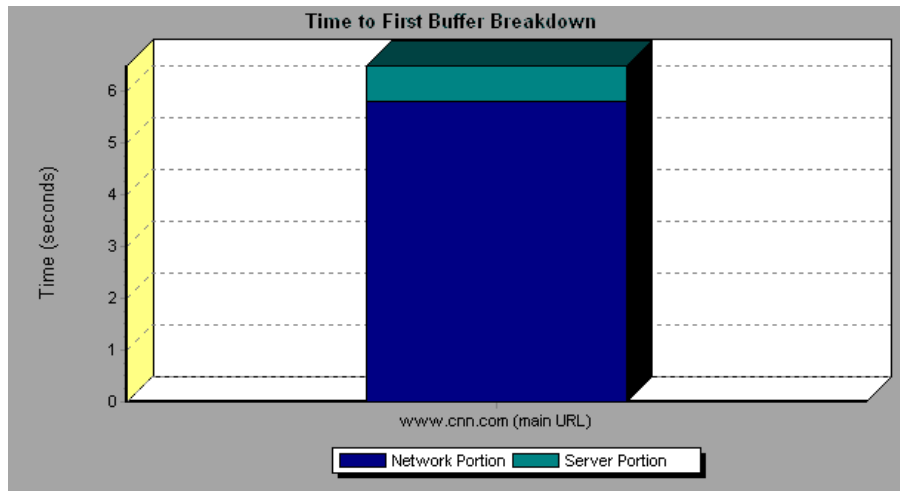
Note: Each measurement displayed on the page level is the sum of that measurement recorded for each page component. For example, the network time for www.cnn.com is the sum of the network time for each of the page's components.

Network time is defined as the average amount of time that passes from the moment the first HTTP request is sent until receipt of ACK.

Server time is defined as the average amount of time that passes from the receipt of ACK of the initial HTTP request (usually GET) until the first buffer is successfully received back from the Web server.

In the above graph, it is apparent that network time is greater than server time.

Note: Because server time is being measured from the client, network time may influence this measurement if there is a change in network performance from the time the initial HTTP request is sent until the time the first buffer is sent. The server time displayed, therefore, is estimated server time and may be slightly inaccurate.

The graph can only be viewed as a bar graph.

You can break the main cnn.com URL down further to view the time to first buffer breakdown for each of its components.
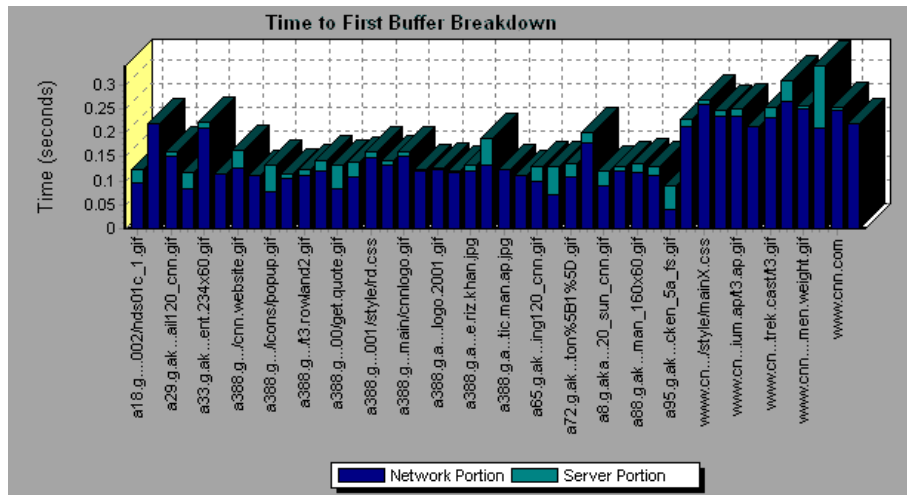
It is apparent that for the main cnn.com component (the first component on the right), the time to first buffer breakdown is almost all network time.

Time to First Buffer Breakdown (Over Time) Graph

The Time to First Buffer Breakdown (Over Time) graph displays each Web page component's server and network time (in seconds) during each second of the scenario run, for the period of time until the first buffer is successfully received back from the Web server. You can use this graph to determine when during the scenario run a server- or network-related problem occurred.

The x-axis represents the elapsed time from the beginning of the scenario run. The y-axis represents the average network or server time (in seconds) for each component.
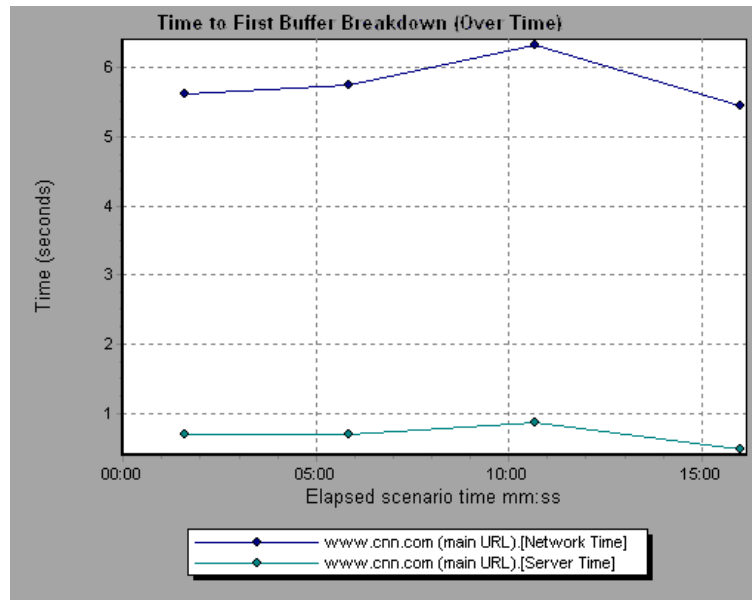
Network time is defined as the average amount of time that passes from the moment the first HTTP request is sent until receipt of ACK.

Server time is defined as the average amount of time that passes from the receipt of ACK of the initial HTTP request (usually GET) until the first buffer is successfully received back from the Web server.

Because server time is being measured from the client, network time may influence this measurement if there is a change in network performance

from the time the initial HTTP request is sent until the time the first buffer is sent. The server time displayed, therefore, is estimated server time and may be slightly inaccurate.

Note: Each measurement displayed on the page level is the sum of that measurement recorded for each page component. For example, the network time for www.cnn.com is the sum of the network time for each of the page's components.

You can break the main cnn.com URL down further to view the time to first buffer breakdown for each of its components.
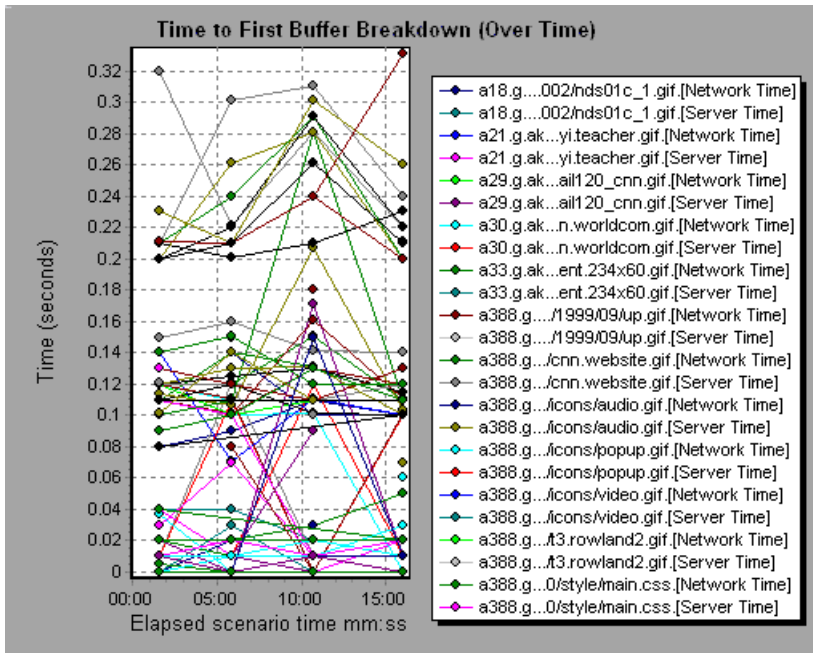
Note: When the Time to First Buffer Breakdown (Over Time) graph is selected from the Web Page Breakdown graph, it appears as an area graph.

Downloaded Component Size Graph

The Downloaded Component Size graph displays the size of each Web page component. For example, the following graph shows that the www.cnn.com/WEATHER component is 39.05% of the total size, whereas the main cnn.com component is 34.56% of the total size.
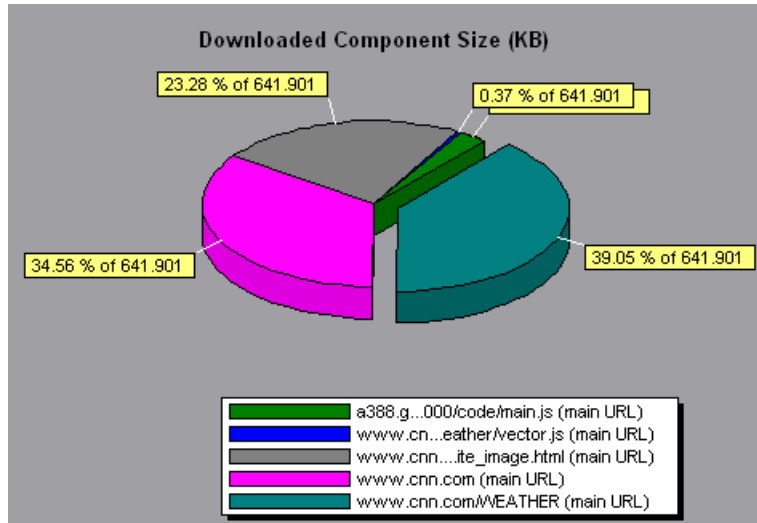
Note: The Web page size is a sum of the sizes of each of its components.

You can break the main cnn.com URL down further to view the size of each
of its components.
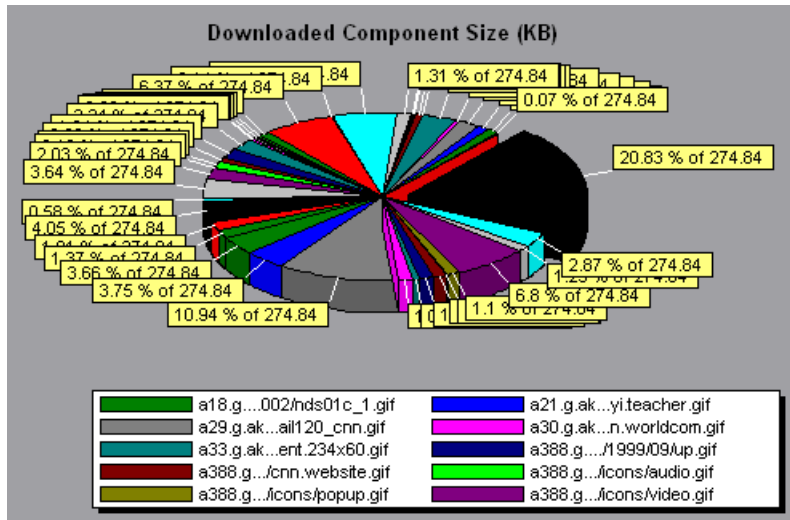
In the above example, the cnn.com component's size (20.83% of the total
size) may have contributed to the delay in its downloading. To reduce
download time, it may help to reduce the size of this component.

Note: The Downloaded Component Size graph can only be viewed as a pie
graph.

8

User-Defined Data Point Graphs

After a scenario run, you can use the User-Defined Data Point graphs to display the values of user-defined data points in the Vuser script.

This chapter describes the:

➤ Data Points (Sum) Graph

➤ Data Points (Average) Graph

About User-Defined Data Point Graphs

The User-Defined Data Point graphs display the values of user-defined data points. You define a data point in your Vuser script by inserting an **lr_user_data_point** function at the appropriate place (**user_data_point** for GUI Vusers and **lr.user_data_point** for Java Vusers).

```
Action1()
{
    lr_think_time(1);
    lr_user_data_point ("data_point_1",1);
    lr_user_data_point ("data_point_2",2);
    return 0;
}
```

For Vuser protocols that support the graphical script representations such as Web and Oracle NCA, you insert a data point as a User Defined step. Data point information is gathered each time the script executes the function or step. For more information about data points, see the online *LoadRunner Function Reference*.

Data points, like other LoadRunner data, are aggregated every few seconds, resulting in less data points shown on the graph than actually recorded. For more information see "Changing the Granularity of the Data" on page 55.

Data Points (Sum) Graph

The Data Points (Sum) graph shows the sum of the values for user-defined data points throughout the scenario run.

The x-axis represents the number of seconds that elapsed since the start time of the run. The y-axis displays the sum of the recorded data point values.

This graph typically indicates the total amount of measurements which all Virtual Users are able to generate. For example, suppose only a certain set of circumstances allow a Vuser to call a server. Each time it does, a data point is recorded. In this case, the Sum graph displays the total number of times that Vusers call the function.

In the following example, the call to the server is recorded as the data point *user_data_point_val_1*. It is shown as a function of the elapsed scenario time.
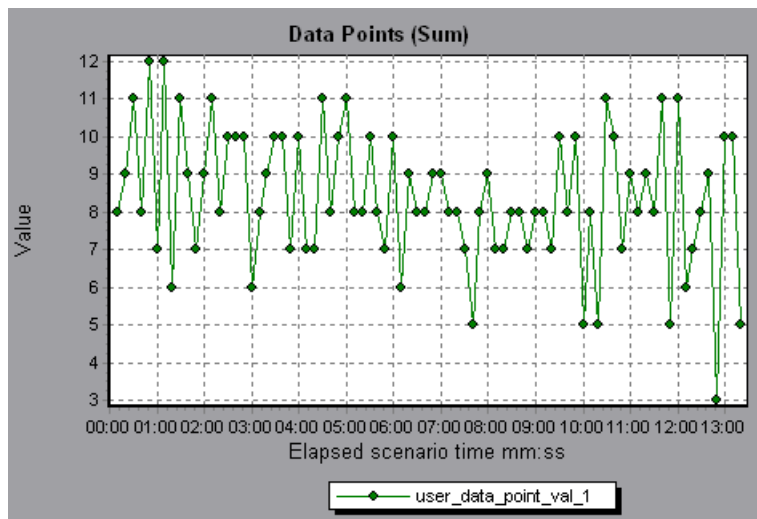
Data Points (Average) Graph

The Data Points (Average) graph shows the average values that were recorded for user-defined data points during the scenario run.

The x-axis represents the number of seconds that elapsed since the start time of the run. The y-axis displays the average values of the recorded data point statements.

This graph is typically used in cases where the actual value of the measurement is required. Suppose that each Vuser monitors CPU utilization on its machine and records it as a data point. In this case, the actual recorded value of CPU utilization is required. The Average graph displays the average value recorded throughout the scenario.

In the following example, the CPU utilization is recorded as the data point *user_data_point_val_1.* It is shown as a function of the elapsed scenario time:

9
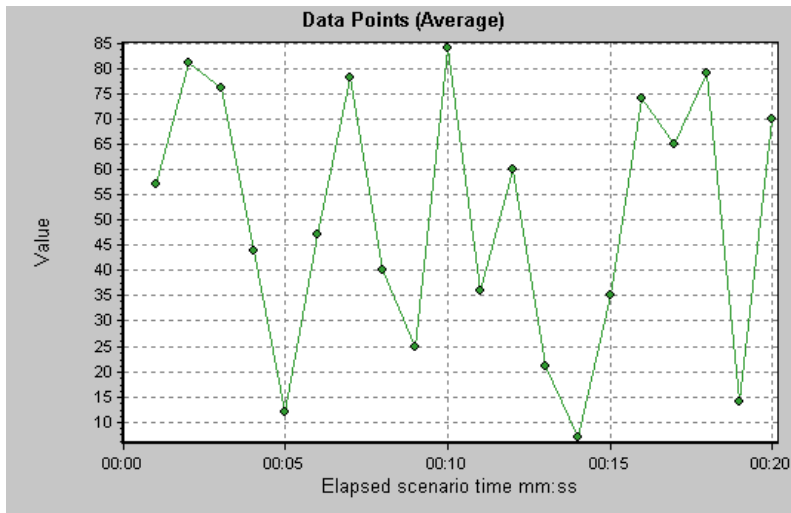
System Resource Graphs

After running a scenario, you can check the various system resources that were monitored during the scenario using one or more of the following System Resource graphs:

➤ Windows Resources Graph

➤ UNIX Resources Graph

➤ SNMP Resources Graph

➤ Antara FlameThrower Resources Graph

➤ SiteScope Graph

About System Resource Graphs

System Resource graphs display the system resource usage measured by the online monitors during the scenario run. These graphs require that you specify the resources you want to measure *before* running the scenario. For more information, see the section on online monitors in the *LoadRunner Controller User's Guide*.

Windows Resources Graph

The Windows Resources graph shows the NT and Windows 2000 resources measured during the scenario. The NT and Windows 2000 measurements correspond to the built-in counters available from the Windows Performance Monitor.

This graph helps you determine the impact of Vuser load on the various system resources. The x-axis represents the elapsed time. The y-axis represents the resource usage.

Note: To obtain data for this graph, you need to select the desired measurements for the online monitor (from the Controller) before running the scenario.
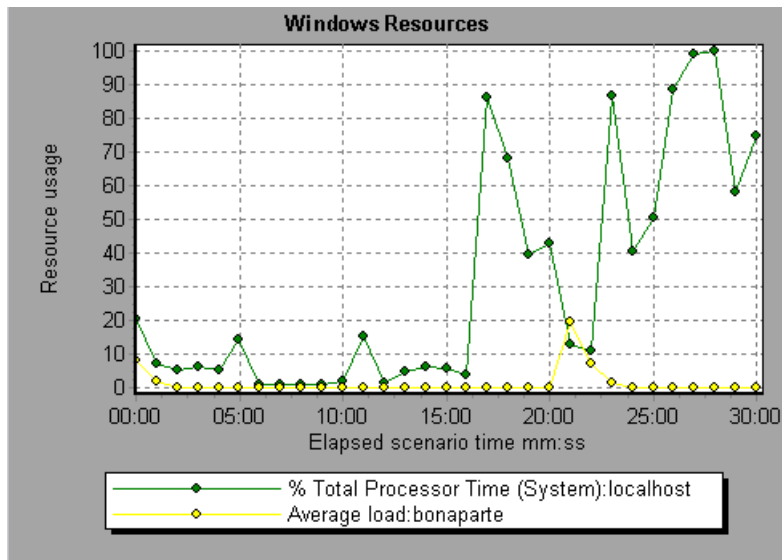
The following default measurements are available for Windows Resources:

Object	Measurement	Description
System	**% Total Processor Time**	The average percentage of time that all the processors on the system are busy executing non-idle threads. On a multi-processor system, if all processors are always busy, this is 100%, if all processors are 50% busy this is 50% and if 1/4th of the processors are 100% busy this is 25%. It can be viewed as the fraction of the time spent doing useful work. Each processor is assigned an Idle thread in the Idle process which consumes those unproductive processor cycles not used by any other threads.
Processor	**% Processor Time (Windows 2000)**	The percentage of time that the processor is executing a non-idle thread. This counter was designed as a primary indicator of processor activity. It is calculated by measuring the time that the processor spends executing the thread of the idle process in each sample interval, and subtracting that value from 100%. (Each processor has an idle thread which consumes cycles when no other threads are ready to run.) It can be viewed as the percentage of the sample interval spent doing useful work. This counter displays the average percentage of busy time observed during the sample interval. It is calculated by monitoring the time the service was inactive, and then subtracting that value from 100%.
System	**File Data Operations/sec**	The rate at which the computer issues read and write operations to file system devices. This does not include File Control Operations.

Object	Measurement	Description
System	**Processor Queue Length**	The instantaneous length of the processor queue in units of threads. This counter is always 0 unless you are also monitoring a thread counter. All processors use a single queue in which threads wait for processor cycles. This length does not include the threads that are currently executing. A sustained processor queue length greater than two generally indicates processor congestion. This is an instantaneous count, not an average over the time interval.
Memory	**Page Faults/sec**	This is a count of the page faults in the processor. A page fault occurs when a process refers to a virtual memory page that is not in its Working Set in the main memory. A page fault will not cause the page to be fetched from disk if that page is on the standby list (and hence already in main memory), or if it is in use by another process with which the page is shared.
PhysicalDisk	**% Disk Time**	The percentage of elapsed time that the selected disk drive is busy servicing read or write requests.
Memory	**Pool Nonpaged Bytes**	The number of bytes in the nonpaged pool, a system memory area where space is acquired by operating system components as they accomplish their appointed tasks. Nonpaged pool pages cannot be paged out to the paging file. They remain in main memory as long as they are allocated.

Object	Measurement	Description
Memory	**Pages/sec**	The number of pages read from the disk or written to the disk to resolve memory references to pages that were not in memory at the time of the reference. This is the sum of Pages Input/sec and Pages Output/sec. This counter includes paging traffic on behalf of the system cache to access file data for applications. This value also includes the pages to/from non-cached mapped memory files. This is the primary counter to observe if you are concerned about excessive memory pressure (that is, thrashing), and the excessive paging that may result.
System	**Total Interrupts/sec**	The rate at which the computer is receiving and servicing hardware interrupts. The devices that can generate interrupts are the system timer, the mouse, data communication lines, network interface cards, and other peripheral devices. This counter provides an indication of how busy these devices are on a computer-wide basis. See also Processor:Interrupts/sec.
Objects	**Threads**	The number of threads in the computer at the time of data collection. Notice that this is an instantaneous count, not an average over the time interval. A thread is the basic executable entity that can execute instructions in a processor.
Process	**Private Bytes**	The current number of bytes that the process has allocated that cannot be shared with other processes.

UNIX Resources Graph

The UNIX Resources graph shows the UNIX resources measured during the scenario. The UNIX measurements include those available by the *rstatd* daemon: average load, collision rate, context switch rate, CPU utilization, incoming packets error rate, incoming packets rate, interrupt rate, outgoing packets error rate, outgoing packets rate, page-in rate, page-out rate, paging rate, swap-in rate, swap-out rate, system mode CPU utilization, and user mode CPU utilization.

This graph helps you determine the impact of Vuser load on the various system resources. The x-axis represents the elapsed time. The y-axis represents the resource usage.

Note: To obtain data for this graph, you need to select the desired measurements for the online monitor (from the Controller) before running the scenario.
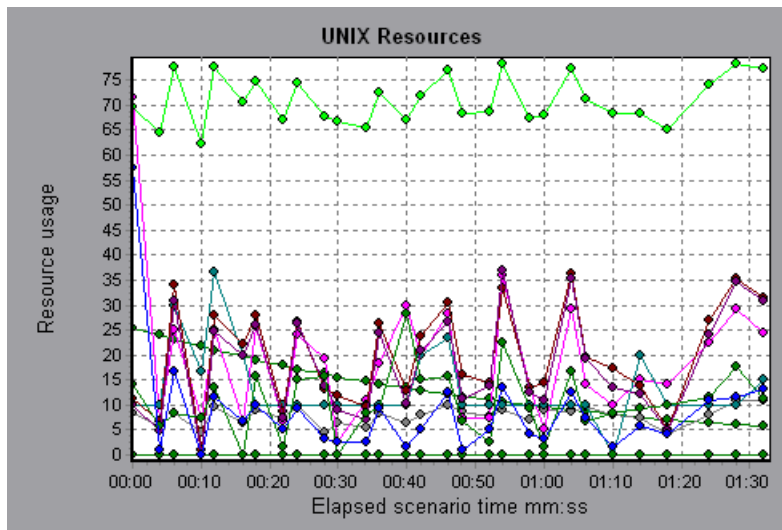
The following default measurements are available for UNIX machines:

Measurement	Description
Average load	Average number of processes simultaneously in 'Ready' state during the last minute
Collision rate	Collisions per second detected on the Ethernet
Context switches rate	Number of switches between processes or threads, per second
CPU utilization	Percent of time that the CPU is utilized
Disk rate	Rate of disk transfers
Incoming packets error rate	Errors per second while receiving Ethernet packets
Incoming packets rate	Incoming Ethernet packets per second
Interrupt rate	Number of device interrupts per second
Outgoing packets errors rate	Errors per second while sending Ethernet packets
Outgoing packets rate	Outgoing Ethernet packets per second
Page-in rate	Number of pages read to physical memory, per second
Page-out rate	Number of pages written to pagefile(s) and removed from physical memory, per second
Paging rate	Number of pages read to physical memory or written to pagefile(s), per second
Swap-in rate	Number of processes being swapped
Swap-out rate	Number of processes being swapped
System mode CPU utilization	Percent of time that the CPU is utilized in system mode
User mode CPU utilization	Percent of time that the CPU is utilized in user mode

SNMP Resources Graph

The SNMP Resources graph shows statistics for machines running an SNMP agent, using the Simple Network Management Protocol (SNMP).

Note: To obtain data for this graph, you need to enable the SNMP monitor (from the Controller) and select the default measurements you want to display, before running the scenario.

The following graph displays SNMP measurements for a machine called *bonaparte*:
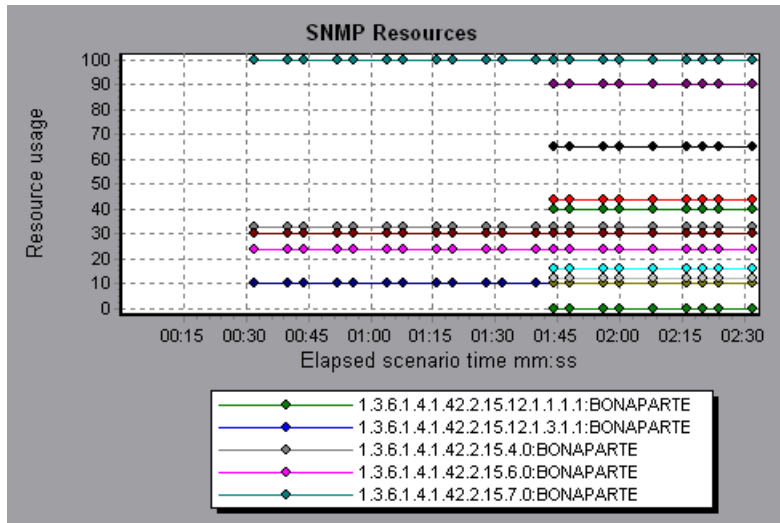
Antara FlameThrower Resources Graph

The Antara FlameThrower graph displays statistics about the resource usage on the Antara FlameThrower server during the scenario run.

The x-axis represents the time that has elapsed since the start of the scenario run. The y-axis represents the resource usage.

Note: To obtain data for this graph, you need to enable the iPlanet/Netscape online monitor (from the Controller) and select the default measurements you want to display, before running the scenario.

The following default measurements are available for the Antara FlameThrower server:

Layer Performance Counters

Measurement	Description
TxBytes	The total number of Layer 2 data bytes transmitted.
TxByteRate(/sec)	The number of Layer 2 data bytes transmitted per second.
TxFrames	The total number of packets transmitted.
TxFrameRate(/sec)	The number of packets transmitted per second.
RxBytes	The total number of Layer 2 data bytes received.
RxByteRate(/sec)	The number of Layer 2 data bytes received per second.
RxFrames	The total number of packets received.
RxFrameRate(/sec)	The number of packets received per second.

TCP Performance Counters

Measurement	Description
ActiveTCPConns	Total number of currently active TCP connections.
SuccTCPConns	Total number of SYN ACK packets received.
SuccTCPConn Rate(/sec)	Number of SYN ACK packets received per second.
TCPConnLatency(mil isec)	Interval between transmitting a SYN packet and receiving a SYN ACK reply packet in msec.
MinTCPConn Latency(milisec)	Minimum TCPConnectionLatency in msec.
MaxTCPConn Latency(milisec)	Maximum TCPConnectionLatency in msec.
TCPSndConnClose	Total number of FIN or FIN ACK packets transmitted (Client).
TCPRcvConnClose	Total number of FIN or FIN ACK packets received (Client).
TCPSndResets	Total number of RST packets transmitted.
TCPRcvResets	Total number of RST packets received.
SYNSent	Total number of SYN packets transmitted.
SYNSentRate(/sec)	Number of SYN packets transmitted per second.
SYNAckSent	Total number of SYN ACK packets transmitted.
SYNAckRate(/sec)	Number of SYN ACK packets transmitted per second.

HTTP Performance Counters

Measurement	Description
HTTPRequests	Total number of HTTP Request command packets transmitted.
HTTPRequestRate (/sec)	Number of HTTP Request packets transmitted per second.
AvgHTTPData Latency(milisecs)	The average HTTP Data Latency over the past second in msec.
HTTPData Latency(milisecs)	Interval between transmitting a Request packet and receiving a response in msec.
DataThroughput (bytes/sec)	The number of data bytes received from the HTTP server per second.
MinHTTPData Latency(milisecs)	Minimum HTTPDataLatency in msec.
MaxHTTPData Latency(milisecs)	Maximum HTTPDataLatency in msec.
MinData Throughput (bytes/sec)	Minimum HTTPDataThroughput in seconds.
MaxData Throughput (bytes/sec)	Maximum HTTPDataThroughput in seconds.
SuccHTTPRequests	Total number of successful HTTP Request Replies (200 OK) received.
SuccHTTPRequest Rate(/sec)	Number of successful HTTP Request Replies (200 OK) received per second.
UnSuccHTTP Requests	Number of unsuccessful HTTP Requests.

SSL/HTTPS Performance Counters

Measurement	Description
SSLConnections	Number of ClientHello messages sent by the Client.
SSLConnection Rate(/sec)	Number of ClientHello messages sent per second.
SuccSSL Connections	Number of successful SSL Connections. A successful connection is one in which the Client receives the Server's finished handshake message without any errors.
SuccSSLConnectionR ate(/sec)	Number of successful SSL connections established per second.
SSLAlertErrors	Number of SSL alert messages received by the client (e.g. bad_record_mac, decryption_failed, handshake_failure, etc.)
SuccSSLResumed Sessions	Number of SSL Sessions that were successfully resumed.
FailedSSLResumed Sessions	Number of SSL Sessions that were unable to be resumed.

Sticky SLB Performance Counters

Measurement	Description
Cookie AuthenticationFail	The number of Cookie's that were not authenticated by the Server.
SuccCookie Authentication	The number of Cookie's authenticated by the server.
SSLClientHellos	The number of Client Hello packets sent to the server.
SSLServerHellos	The number of Server Hello packets sent to back to the client.

Measurement	Description
SSLSessionsFailed	The number of Session ID's that were not authenticated by the server.
SSLSessions Resumed	The number of Session ID's authenticated by the server.
succSSLClientHellos	The number of Client Hello replies received by the client or packets received by the server.
succSSLServerHellos	The number of Server Hello's received by the client.

FTP Performance Counters

Measurement	Description
TPUsers	Total number of Ftp User command packets transmitted.
FTPUserRate(/sec)	Number of Ftp User command packets transmitted per second.
FTPUserLatency (milisecs)	Interval between transmitting a Ftp User command packet and receiving a response in msec.
MinFTPUserLatency(milisecs)	Minimum FTPUsersLatency in msec.
MaxFTPUserLatency(milisecs)	Maximum FTPUsersLatency in msec.
SuccFTPUsers	Total number of successful Ftp User command replies received.
SuccFTPUserRate (/sec)	Number of successful Ftp User command replies received per second.
FTPPasses	Total number of FTP PASS packets transmitted.
FTPPassRate(/sec)	Number of FTP PASS packets transmitted per second.
FTPPassLatency (milisecs)	Interval between transmitting a Ftp PASS packet and receiving a response in msec.
MinFTPPassLatency (milisecs)	Minimum FTPPassLatency in msec.

Measurement	Description
MaxFTPPassLatency(milisecs)	Maximum FTPPassLatency in msec.
SuccFTPPasses	Total number of successful FTP PASS replies received.
SuccFTPPassRate (/sec)	Number of successful FTP PASS replies received per second.
FTPControl Connections	Total number of SYN packets transmitted by the FTP client.
FTPControl ConnectionRate (/sec)	Number of SYN packets transmitted by the FTP client per second.
SuccFTPControl Connections	Total number of SYN ACK packets received by the FTP client.
SuccFTPControl ConnectionRate (/sec)	Number of SYN ACK packets received by the FTP Client per second.
FTPData Connections	Number of SYN ACK packets received by the FTP client per second.
FTPDataConnectionRate(/sec)	Number of SYN ACK packets transmitted by the FTP Client or received by the FTP Server per second.
SuccFTPData Connections	Total number of SYN ACK packets transmitted by the FTP Client or received by the FTP Server.
SuccFTPData ConnectionRate (/sec)	Number of SYN ACK packets received by the FTP server per second.
FtpAuthFailed	Total number of error replies received by the FTP client.
FTPGets	Total number of client Get requests.
FTPPuts	Total number of client Put requests.
SuccFTPGets	Total number of successful Get requests (data has been successfully transferred from server to client).
SuccFTPPuts	Total number of successful Put requests (data has been successfully transferred from client to server).

SMTP Performance Counters

Measurement	Description
SMTPHelos	Total number of HELO packets transmitted.
SMTPHeloRate(/sec)	Number of HELO packets transmitted per second.
SMTPHeloLatency (milisecs)	Interval between transmitting a HELO packet and receiving a response in msec.
MinSMTPHelo Latency(milisecs)	Minimum SMTPHeloLatency in msec.
MaxSMTPHelo Latency(milisecs)	Maximum SMTPHeloLatency in msec.
SuccSMTPHelos	Total number of successful HELO replies received.
SuccSMTPHelo Rate(/sec)	Number of successful HELO replies received per second.
SMTPMailFroms	Total number of Mail From packets transmitted.
SMTPMailFromRate(/sec)	Number of Mail From packets transmitted per second.
SMTPMailFrom Latency(milisecs)	Interval between transmitting a Mail From packet and receiving a response in msec.
MinSMTPMailFromLa tency(milisecs)	Minimum SMTPMailFromLatency in msec.
MaxSMTPMailFromL atency(milisecs)	Maximum SMTPMailFromLatency in msec.
SuccSMTPMail Froms	Total number of successful Mail From replies received.
SuccSMTPMailFromR ate(/sec)	Number of successful Mail From replies received per second.
SMTPRcptTos	Total number of RcptTo packets transmitted.
SMTPRcptToRate (/sec)	Number of RcptTo packets transmitted per second.

Measurement	Description
SMTPRcptTo Latency(milisecs)	Interval between transmitting a RcptTo packet and receiving a response in msec.
MinSMTPRcptTo Latency(milisecs)	Minimum SMTPRcptToLatency in msec.
MaxSMTPRcptTo Latency(milisecs)	Maximum SMTPRcptToLatency in msec.
SuccSMTPRcptTos	Total number of successful RcptTo replies received.
SuccSMTPRcptTo Rate(/sec)	Number of successful RcptTo replies received per second.
SMTPDatas	Total number of Data packets transmitted.
SMTPDataRate(/sec)	Number of Data packets transmitted per second.
SMTPDataLatency (milisecs)	Interval between transmitting a Data packet and receiving a response in msec.
MinSMTPData Latency(milisecs)	Minimum SMTPDataLatency in msec.
MaxSMTPData Latency(milisecs)	Maximum SMTPDataLatency in msec.
SuccSMTPDatas	Total number of successful Data replies received.
SuccSMTPDataRate (/sec)	Number of successful Data replies received per second.

POP3 Performance Counters

Measurement	Description
POP3Users	Total number of Pop3 User command packets transmitted.
POP3UserRate(/sec)	Number of Pop3 User command packets transmitted per second.
POP3UserLatency (milisecs)	Interval between transmitting a Pop3 User command packet and receiving a response in msec.

Measurement	Description
MinPOP3User Latency(milisecs)	Minimum POP3UserLatency in msec.
MaxPOP3User Latency(milisecs)	Maximum POP3UserLatency in msec.
SuccPOP3Users	Total number of successful Pop3 User replies received.
SuccPOP3UserRate (/sec)	Number of successful Pop3 User replies received per second.
POP3Passes	Total number of Pop3 Pass command packets transmitted.
POP3PassRate(/sec)	Number of Pop3 Pass command packets transmitted per second.
POP3PassLatency (milisecs)	Interval between transmitting a Pop3 Pass packet and receiving a response in msec.
MinPOP3Pass Latency(milisecs)	Minimum POP3PassLatency in msec.
MaxPOP3Pass Latency(milisecs)	Maximum POP3PassLatency in msec.
SuccPOP3Passes	Total number of successful Pop3 Pass replies received.
SuccPOP3PassRate (/sec)	Number of successful Pop3 Pass replies received per second.
POP3Stats	Total number of Pop3 Stat command packets sent.
POP3StatRate(/sec)	Number of Pop3 Stat command packets transmitted per second.
POP3StatLatency (milisecs)	Interval between transmitting a Pop3 Stat packet and receiving a response in msec.
MinPOP3Stat Latency(milisecs)	Minimum POP3StartLatency in msec.
MaxPOP3Stat Latency(milisecs)	Maximum POP3StartLatency in msec.
SuccPOP3Stats	Total number of successful Pop3 Stat replies received.

Measurement	Description
SuccPOP3StatRate (/sec)	Number of successful Pop3 Stat replies received per second.
POP3Lists	Total number of Pop3 List command packets transmitted.
POP3ListRate(/sec)	Number of Pop3 List command packets transmitted per second.
POP3ListLatency (milisecs)	Interval between transmitting a Pop3 List packet and receiving a response in msec.
MinPOP3List Latency(milisecs)	Minimum POP3ListLatency in msec.
MaxPOP3List Latency(milisecs)	Maximum POP3ListLatency in msec.
SuccPOP3Lists	Total number of successful Pop3Lists received.
SuccPOP3ListRate (/sec)	Number of successful Pop3Lists received per second.
POP3Retrs	Total number of Pop3 Retr packets transmitted.
POP3RetrRate(/sec)	Number of Pop3 Retr packets transmitted per second.
POP3RetrLatency (milisecs)	Interval between transmitting a Pop3 Retr packet and receiving a response in msec.
MinPOP3Retr Latency(milisecs)	Minimum POP3RetrLatency in msec.
MaxPOP3Retr Latency(milisecs)	Maximum POP3RetrLatency in msec.
SuccPOP3Retrs	Total number of successful Pop3Retrs received.
SuccPOP3RetrRate (/sec)	Number of successful Pop3Retrs received per second.

DNS Performance Counters

Measurement	Description
SuccPrimaryDNS Request	Total number of Successful DNS requests made to the Primary DNS server.
SuccSecondaryDNSR equest	Total number of Successful DNS requests made to the Secondary DNS server.
SuccDNSData RequestRate(/sec)	Number of Successful DNS Request packets transmitted per second.
PrimaryDNSFailure	Total number of DNS requests failures received from the Primary DNS server.
PrimaryDNSRequest	Total number of DNS requests made to the Primary DNS server.
SecondaryDNS Failure	Total number of DNS requests failures received from the Secondary DNS server.
SecondaryDNS Request	Total number of DNS requests made to the Secondary DNS server.
MinDNSData Latency	Minimum DNS Data Latency in msec.
MaxDNSData Latency	Maximum DNS Data Latency in msec.
CurDNSData Latency	Interval between sending a DNS request packet and receiving a response in msec.
DNSDataRequest Rate(/sec)	Number of DNS Request packets transmitted per second.
NoOf ReTransmission	Total number of DNS Request packets re
NoOfAnswers	Total number of Answers to the DNS Request packets.

153

Attacks Performance Counters

Measurement	Description
Attacks	Total number of attack packets transmitted (All Attacks)
AttackRate(/sec)	Number of attack packets transmitted per second (ARP, Land, Ping, SYN, and Smurf)
Havoc Flood	Number of Havoc packets generated (Stacheldraht only)
Icmp Flood	Number of ICMP attack packets generated (TFN, TFN2K, & Stacheldraht)
Mix Flood	Number of Mix packets generated (TFN2K only)
Mstream Flood	Number of Mstream packets generated (Stacheldraht only)
Null Flood	Number of Null packets generated (Stacheldraht only)
Smurf Flood	Number of Smurf packets generated (TFN, TFN2K, & Stacheldraht)
Syn Flood	Number of SYN packets generated (TFN, TFN2K, & Stacheldraht)
Targa Flood	Number of Targa packets generated (TFN2K only)
Udp Flood	Number of UDP packets generated (All DDoS Attacks only)

SiteScope Graph

The SiteScope graph displays statistics about the resource usage on the SiteScope machine during the scenario run.

Note: To obtain data for this graph, you need to enable the SiteScope online monitor (from the Controller) and select the default measurements you want to display, before running the scenario.

The following graph displays the percentage of memory resources used, the number of pages read per second, and the CPU utilization:

10
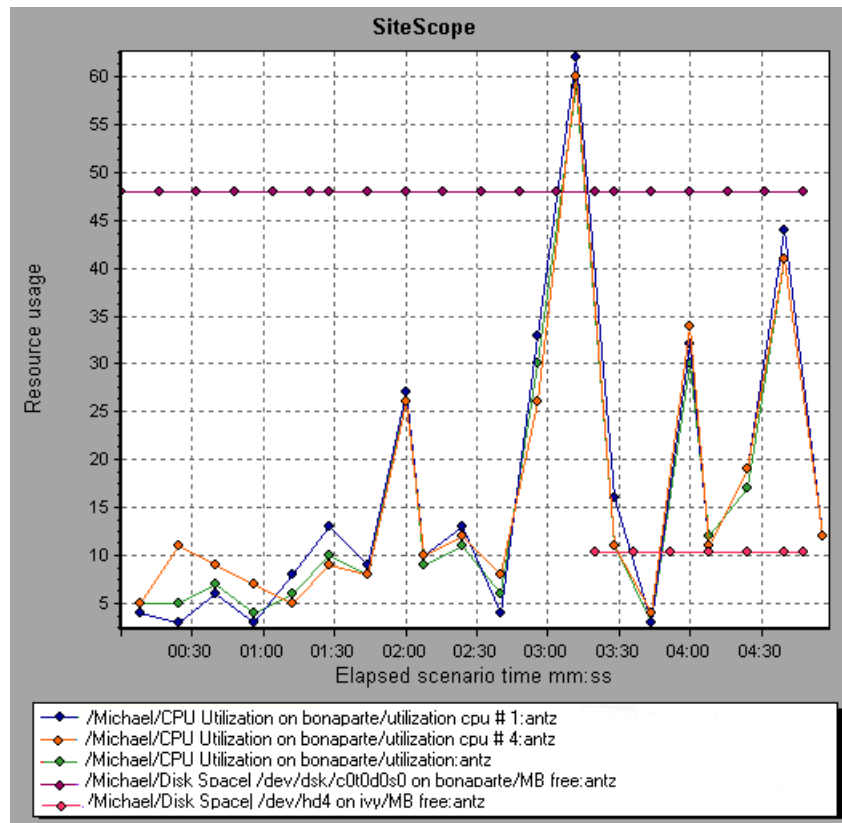
Network Monitor Graphs

You can use Network graphs to determine whether your network is causing a delay in the scenario. You can also determine the problematic network segment.

This chapter describes:

➤ Understanding Network Monitoring

➤ Network Delay Time Graph

➤ Network Sub-Path Time Graph

➤ Network Segment Delay Graph

➤ Verifying the Network as a Bottleneck

About Network Monitoring

Network configuration is a primary factor in the performance of applications and Web systems. A poorly designed network can slow client activity to unacceptable levels.

In an application, there are many network segments. A single network segment with poor performance can affect the entire application.

Network graphs let you locate the network-related problem so that it can be fixed.

Understanding Network Monitoring

The following diagram shows a typical network. In order to go from the server machine to the Vuser machine, data must travel over several segments.
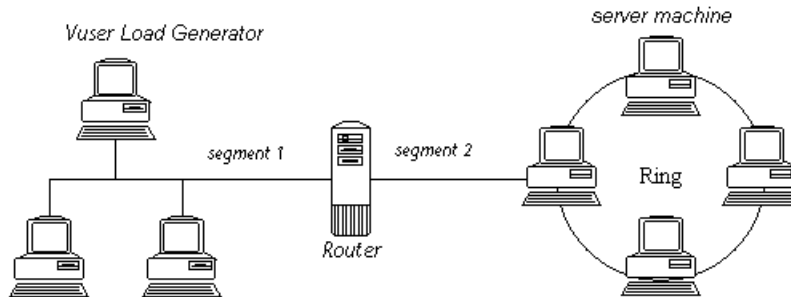
To measure network performance, the Network monitor sends packets of data across the network. When a packet returns, the monitor calculates the time it takes for the packet to go to the requested node and return. The Network Sub-Path Time graph displays the delay from the source machine to each node along the path. The Network Segment Delay graph displays the delay for each segment of the path. The Network Delay Time graph displays the delay for the complete path between the source and destination machines.

Using the Network Monitor graphs, you can determine whether the network is causing a bottleneck. If the problem is network-related, you can locate the problematic segment so that it can be fixed.

In order for Analysis to generate Network monitor graphs, you must activate the Network monitor before executing the scenario. In the Network monitor settings, you specify the path you want to monitor. For information about setting up the Network monitor, see the *LoadRunner Controller User's Guide*.

Network Delay Time Graph

The Network Delay Time graph shows the delays for the complete path between the source and destination machines (for example, the database server and Vuser load generator). The graph maps the delay as a function of the elapsed scenario time.

Each path defined in the Controller is represented by a separate line with a different color in the graph.

The following graph shows the network delay as a function of the elapsed scenario time. It shows a delay of 16 milliseconds in the 8th minute of the scenario.
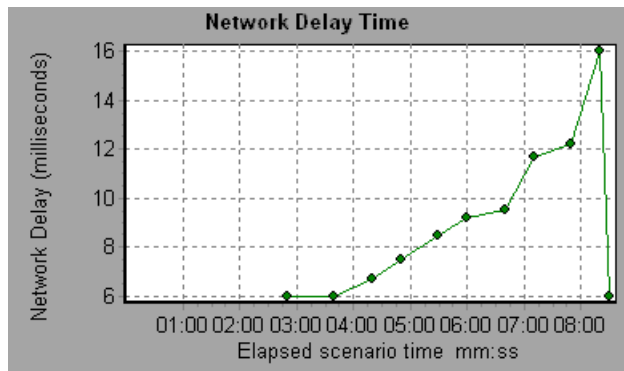
Network Sub-Path Time Graph

The Network Sub-Path Time graph displays the delay from the source machine to each node along the path according to the elapsed scenario time. Each segment is displayed as a separate line with a different color.

In the following example, four segments are shown. The graph indicates that one segment caused a delay of 70 milliseconds in the sixth minute.
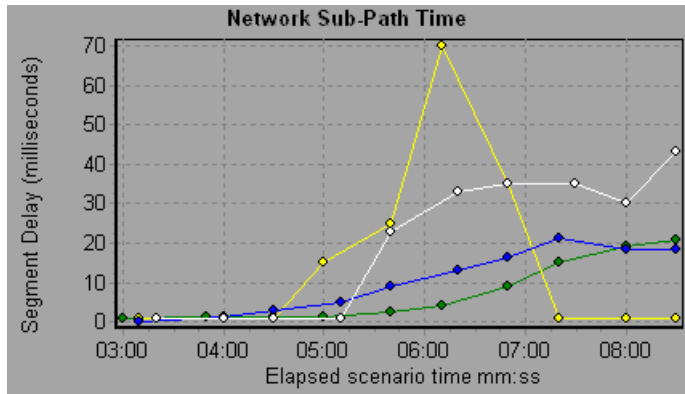
Note: The delays from the source machine to each of the nodes are measured concurrently, yet independently. It is therefore possible that the delay from the source machine to one of the nodes could be greater than the delay for the complete path between the source and destination machines.

Network Segment Delay Graph

The Network Segment Delay graph shows the delay for each segment of the path according to the elapsed scenario time. Each segment is displayed as a separate line with a different color.

In the following example, four segments are shown. The graph indicates that one segment caused a delay of 70 seconds in the sixth minute.
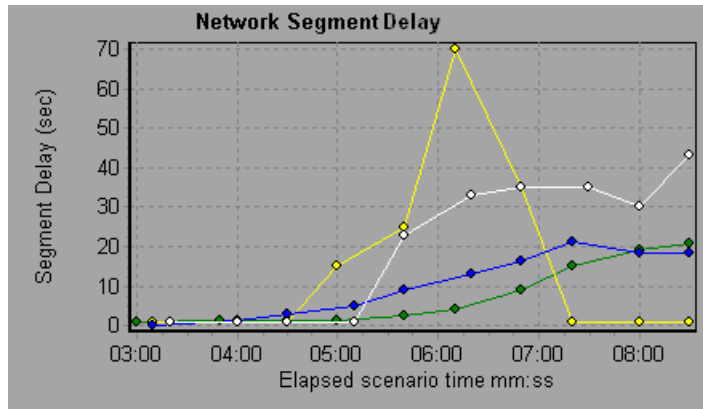
Note: The segment delays are measured approximately, and do not add up to the network path delay which is measured exactly. The delay for each segment of the path is estimated by calculating the delay from the source machine to one node and subtracting the delay from the source machine to another node. For example, the delay for segment B to C is calculated by measuring the delay from the source machine to point C, and subtracting the delay from the source machine to point B.

Verifying the Network as a Bottleneck

You can merge various graphs to determine if the network is a bottleneck. For example, using the Network Delay Time and Running Vusers graphs, you can determine how the number of Vusers affects the network delay. The Network Delay Time graph indicates the network delay during the scenario run. The Running Vusers graph shows the number of running Vusers.

In the following merged graph, the network delays are compared to the running Vusers. The graph shows that when all 10 Vusers were running, a network delay of 22 milliseconds occurred, implying that the network may be overloaded.

11
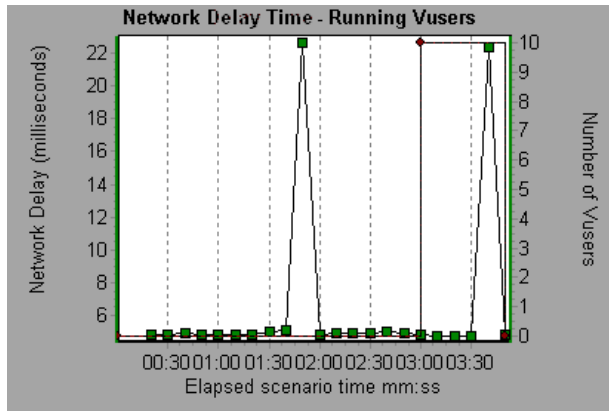
Firewall Server Monitor Graphs

After a scenario run, you can use the Firewall server monitor graphs to analyze Firewall server performance.

This chapter describes the:

➤ Check Point FireWall-1 Server Graph

About Firewall Server Monitor Graphs

Firewall server monitor graphs provide you with performance information for firewall servers. Note that in order to obtain data for these graphs, you need to activate the Firewall server online monitor before running the scenario. When you set up the online monitor for the Firewall server, you indicate which statistics and measurements to monitor. For more information on activating and configuring Firewall server monitors, see the *LoadRunner Controller User's Guide*.

Check Point FireWall-1 Server Graph

The Check Point FireWall-1 graph shows statistics on Check Point's Firewall server as a function of the elapsed scenario time.

The x-axis represents the time that has elapsed since the start of the scenario run. The y-axis represents the resource usage.
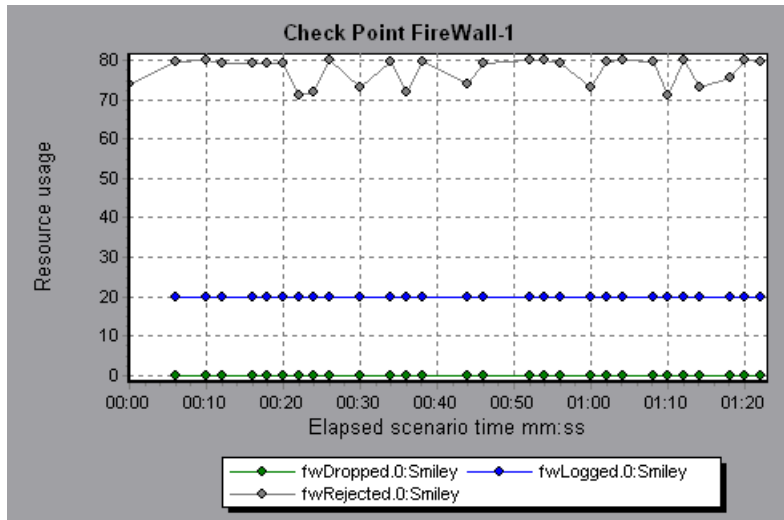
This graph displays the *fwDropped, fwLogged,* and *fwRejected* measurements during the first minute and twenty seconds of the scenario. Note that there are differences in the scale factor for the measurements: the scale factor for *fwDropped* is 1, the scale factor for *fwLogged* is 10, and the scale factor for *fwRejected* is 0.0001.

Note: To obtain data for this graph, you need to enable the Check Point FireWall-1 monitor (from the Controller) and select the default measurements you want to display, before running the scenario.

The following measurements are available for the Check Point FireWall-1 server:

Measurement	Description
fwRejected	The number of rejected packets.
fwDropped	The number of dropped packets.
fwLogged	The number of logged packets.

12

Web Server Resource Graphs

After a scenario run, you can use Web Server Resource graphs to analyze the performance of your Apache, Microsoft IIS, iPlanet/Netscape, and iPlanet (SNMP) server performance.

This chapter describes:

➤ Apache Server Graph

➤ Microsoft Information Internet Server (IIS) Graph

➤ iPlanet/Netscape Server Graph

➤ iPlanet (SNMP) Server Graph

About Web Server Resource Graphs

Web Server Resource graphs provide you with information about the resource usage of the Apache, Microsoft IIS, iPlanet/Netscape, and iPlanet (SNMP) Web servers. In order to obtain data for these graphs, you need to activate the online monitor for the server and specify which resources you want to measure before running the scenario. For information on activating and configuring the Web Server Resource monitors, see the *LoadRunner Controller User's Guide*.

In order to display all the measurements on a single graph, Analysis may scale them. The Legend tab indicates the scale factor for each resource. To obtain the true value, multiply the scale factor by the displayed value. For example, in the following graph the actual value of *KBytes Sent per second* in the second minute is 1; 10 multiplied by the scale factor of 1/10 (indicated in the Legend tab below).
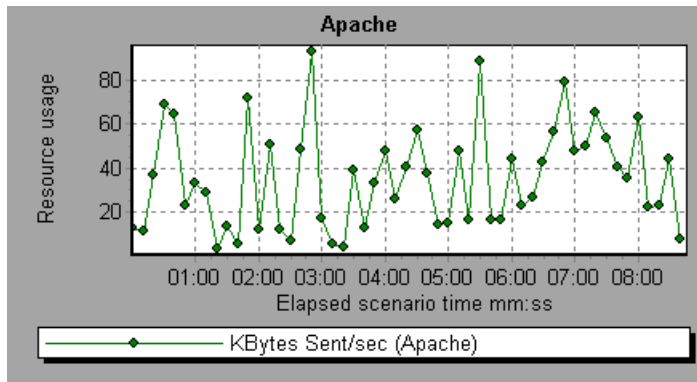
Apache Server Graph

The Apache Server graph shows server statistics as a function of the elapsed scenario time.

The x-axis represents the time that has elapsed since the start of the scenario run. The y-axis represents the resource usage.
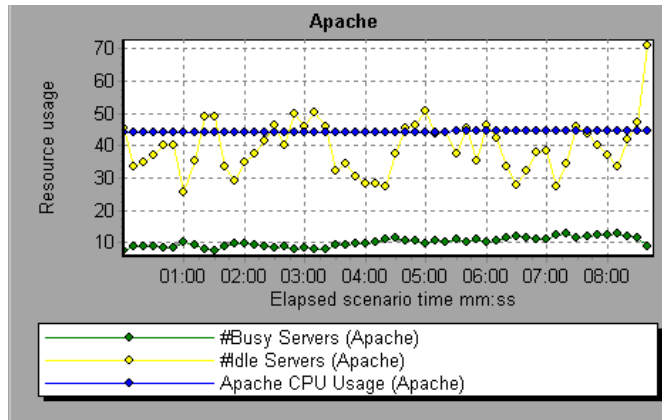
In the above graph, the CPU usage remained steady throughout the scenario. At the end of the scenario, the number of idle servers increased. The number of busy servers remained steady at 1 throughout the scenario, implying that the Vuser only accessed one Apache server.

Note that the scale factor for the *Busy Servers* measurement is 1/10 and the scale factor for *CPU usage* is 10.

Note: To obtain data for this graph, you need to enable the Apache online monitor (from the Controller) and select the default measurements you want to display, before running the scenario.

The following default measurements are available for the Apache server:

Measurement	Description
# Busy Servers	The number of servers in the Busy state
# Idle Servers	The number of servers in the Idle state
Apache CPU Usage	The percentage of time the CPU is utilized by the Apache server
Hits/sec	The HTTP request rate
KBytes Sent/sec	The rate at which data bytes are sent from the Web server

Note: The Apache monitor connects to the Web server in order to gather statistics, and registers one hit for each sampling. The Apache graph, therefore, always displays one hit per second, even if no clients are connected to the Apache server.

Microsoft Information Internet Server (IIS) Graph

The Microsoft IIS Server graph shows server statistics as a function of the elapsed scenario time.

The x-axis represents the time that has elapsed since the start of the scenario run. The y-axis represents the resource usage.
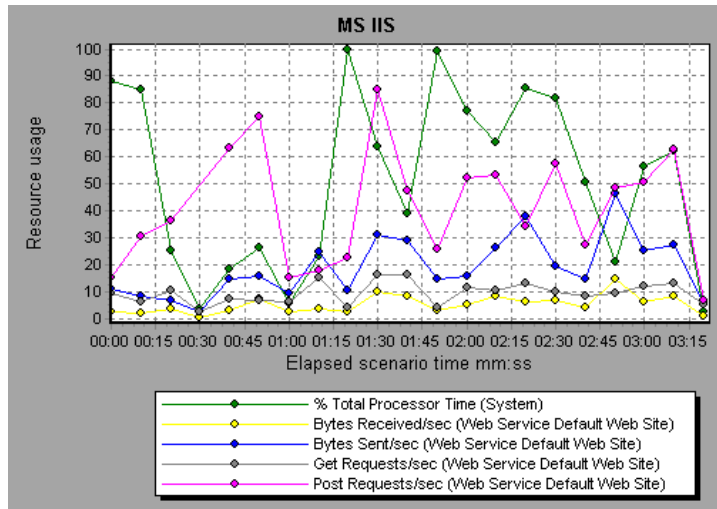
In the above graph, the *Bytes Received/sec* and *Get Requests/sec* measurements remained fairly steady throughout the scenario, while the *% Total Processor Time*, *Bytes Sent/sec*, and *Post Requests/sec* measurements fluctuated considerably.

Note: To obtain data for this graph, you need to enable the MS IIS online monitor (from the Controller) and select the default measurements you want to display, before running the scenario.

Note that the scale factor for the *Bytes Sent/sec* and *Bytes Received/sec* measurements is 1/100, and the scale factor for the *Post Requests/sec* measurement is 10.

The following default measurements are available for the IIS server:

Object	Measurement	Description
Web Service	**Bytes Sent/sec**	The rate at which the data bytes are sent by the Web service
Web Service	**Bytes Received/sec**	The rate at which the data bytes are received by the Web service
Web Service	**Get Requests/sec**	The rate at which HTTP requests using the GET method are made. Get requests are generally used for basic file retrievals or image maps, though they can be used with forms.
Web Service	**Post Requests/sec**	The rate at which HTTP requests using the POST method are made. Post requests are generally used for forms or gateway requests.
Web Service	**Maximum Connections**	The maximum number of simultaneous connections established with the Web service
Web Service	**Current Connections**	The current number of connections established with the Web service
Web Service	**Current NonAnonymous Users**	The number of users that currently have a non-anonymous connection using the Web service
Web Service	**Not Found Errors/sec**	The rate of errors due to requests that could not be satisfied by the server because the requested document could not be found. These are generally reported to the client as an HTTP 404 error code.
Process	**Private Bytes**	The current number of bytes that the process has allocated that cannot be shared with other processes.

iPlanet/Netscape Server Graph

The iPlanet/Netscape Server graph shows server statistics as a function of the elapsed scenario time.

The x-axis represents the time that has elapsed since the start of the scenario run. The y-axis represents the resource usage.
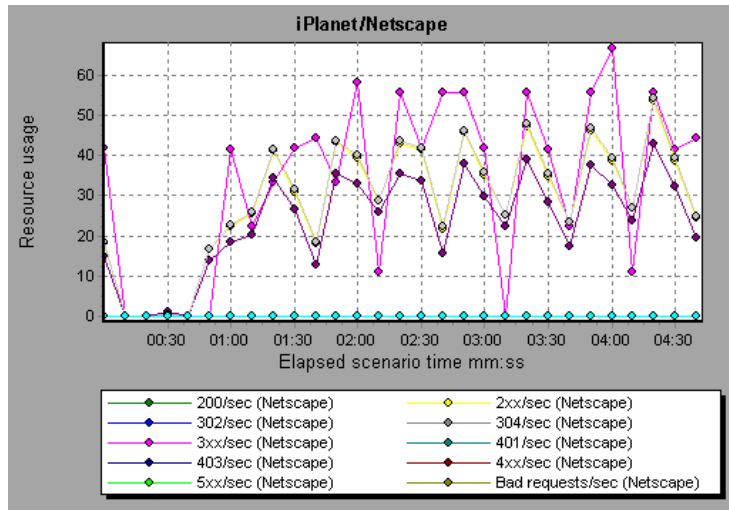
Note that the scale factor for the *302/sec* and *3xx/sec* measurements is 100, and the scale factor for the *Bytes Sent/sec* is 1/100.

Note: To obtain data for this graph, you need to enable the iPlanet/Netscape online monitor (from the Controller) and select the default measurements you want to display, before running the scenario.

The following default measurements are available for the iPlanet/Netscape server:

Measurement	Description
200/sec	The rate of successful transactions being processed by the server
2xx/sec	The rate at which the server handles status codes in the 200 to 299 range
302/sec	The rate of relocated URLs being processed by the server
304/sec	The rate of requests for which the server tells the user to use a local copy of a URL instead of retrieving a newer version from the server
3xx/sec	The rate at which the server handles status codes in the 300 to 399 range
401/sec	The rate of unauthorized requests handled by the server
403/sec	The rate of forbidden URL status codes handled by the server
4xx/sec	The rate at which the server handles status codes in the 400 to 499 range
5xx/sec	The rate at which the server handles status codes 500 and higher
Bad requests/sec	The rate at which the server handles bad requests
Bytes sent/sec	The rate at which bytes of data are sent from the Web server
Hits/sec	The HTTP request rate
xxx/sec	The rate of all status codes (2xx-5xx) handled by the server, excluding timeouts and other errors that did return an HTTP status code

iPlanet (SNMP) Server Graph

The iPlanet (SNMP) Server graph shows server statistics as a function of the elapsed scenario time.

The x-axis represents the time that has elapsed since the start of the scenario run. The y-axis represents the resource usage.

Note: To obtain data for this graph, you need to enable the iPlanet (SNMP) online monitor (from the Controller) and select the default measurements you want to display, before running the scenario.

The following default measurements are available for the iPlanet (SNMP) server:

Measurement	Description
iwsInstanceTable	iPlanet Web Server instances
iwsInstanceEntry	iPlanet Web Server instances
iwsInstanceIndex	Server instance index
iwsInstanceId	Server instance identifier
iwsInstanceVersion	Server instance software version
iwsInstanceDescription	Description of server instance
iwsInstanceOrganization	Organization responsible for server instance
iwsInstanceContact	Contact information for person(s) responsible for server instance
iwsInstanceLocation	Location of server instance
iwsInstanceStatus	Server instance status
iwsInstanceUptime	Server instance uptime
iwsInstanceDeathCount	Number of times server instance processes have died
iwsInstanceRequests	Number of requests processed

Measurement	Description
iwsInstanceInOctets	Number of octets received
iwsInstanceOutOctets	Number of octets transmitted
iwsInstanceCount2xx	Number of 200-level (Successful) responses issued
iwsInstanceCount3xx	Number of 300-level (Redirection) responses issued
iwsInstanceCount4xx	Number of 400-level (Client Error) responses issued
iwsInstanceCount5xx	Number of 500-level (Server Error) responses issued
iwsInstanceCountOther	Number of other (neither 2xx, 3xx, 4xx, nor 5xx) responses issued
iwsInstanceCount200	Number of 200 (OK) responses issued
iwsInstanceCount302	Number of 302 (Moved Temporarily) responses issued
iwsInstanceCount304	Number of 304 (Not Modified) responses issued
iwsInstanceCount400	Number of 400 (Bad Request) responses issued
iwsInstanceCount401	Number of 401 (Unauthorized) responses issued
iwsInstanceCount403	Number of 403 (Forbidden) responses issued
iwsInstanceCount404	Number of 404 (Not Found) responses issued
iwsInstanceCount503	Number of 503 (Unavailable) responses issued
iwsInstanceLoad 1MinuteAverage	System load average for 1 minute
iwsInstanceLoad 5MinuteAverage	System load average for 5 minutes
iwsInstanceLoad 15MinuteAverage	System load average for 15 minutes
iwsInstanceNetwork InOctets	Number of octets transmitted on the network per second

Measurement	Description
iwsInstanceNetwork OutOctets	Number of octets received on the network per second
iwsVsTable	iPlanet Web Server virtual servers
iwsVsEntry	iPlanet Web Server virtual server
iwsVsIndex	Virtual server index
iwsVsId	Virtual server identifier
iwsVsRequests	Number of requests processed
iwsVsInOctets	Number of octets received
iwsVsOutOctets	Number of octets transmitted
iwsVsCount2xx	Number of 200-level (Successful) responses issued
iwsVsCount3xx	Number of 300-level (Redirection) responses issued
iwsVsCount4xx	Number of 400-level (Client Error) responses issued
iwsVsCount5xx	Number of 500-level (Server Error) responses issued
iwsVsCountOther	Number of other (neither 2xx, 3xx, 4xx, nor 5xx) responses issued
iwsVsCount200	Number of 200 (OK) responses issued
iwsVsCount302	Number of 302 (Moved Temporarily) responses issued
iwsVsCount304	Number of 304 (Not Modified) responses issued
iwsVsCount400	Number of 400 (Bad Request) responses issued
iwsVsCount401	Number of 401 (Unauthorized) responses issued
iwsVsCount403	Number of 403 (Forbidden) responses issued
iwsVsCount404	Number of 404 (Not Found) responses issued
iwsVsCount503	Number of 503 (Unavailable) responses issued

Measurement	Description
iwsProcessTable	iPlanet Web Server processes
iwsProcessEntry	iPlanet Web Server process
iwsProcessIndex	Process index
iwsProcessId	Operating system process identifier
iwsProcessThreadCount	Number of request processing threads
iwsProcessThreadIdle	Number of request processing threads currently idle
iwsProcessConnection QueueCount	Number of connections currently in connection queue
iwsProcessConnection QueuePeak	Largest number of connections that have been queued simultaneously
iwsProcessConnection QueueMax	Maximum number of connections allowed in connection queue
iwsProcessConnection QueueTotal	Number of connections that have been accepted
iwsProcessConnection QueueOverflows	Number of connections rejected due to connection queue overflow
iwsProcessKeepalive Count	Number of connections currently in keepalive queue
iwsProcessKeepaliveMax	Maximum number of connections allowed in keepalive queue
iwsProcessSizeVirtual	Process size in kbytes
iwsProcessSizeResident	Process resident size in kbytes
iwsProcessFraction SystemMemoryUsage	Fraction of process memory in system memory
iwsListenTable	iPlanet Web Server listen sockets
iwsListenEntry	iPlanet Web Server listen socket
iwsListenIndex	Listen socket index

Measurement	Description
iwsListenId	Listen socket identifier
iwsListenAddress	Address socket is listening on
iwsListenPort	Port socket is listening on
iwsListenSecurity	Encryption support
iwsThreadPoolTable	iPlanet Web Server thread pools
iwsThreadPoolEntry	iPlanet Web Server thread pool
iwsThreadPoolIndex	Thread pool index
iwsThreadPoolId	Thread pool identifier
iwsThreadPoolCount	Number of requests queued
iwsThreadPoolPeak	Largest number of requests that have been queued simultaneously
iwsThreadPoolMax	Maximum number of requests allowed in queue
iwsCpuTable	iPlanet Web Server CPUs
iwsCpuEntry	iPlanet Web Server CPU
iwsCpuIndex	CPU index
iwsCpuId	CPU identifier
iwsCpuIdleTime	CPU Idle Time
iwsCpuUserTime	CPU User Time
iwsCpuKernelTime	CPU Kernel Time

13

Web Application Server Resource Graphs

After a scenario run, you can use Web Application Server Resource graphs to analyze Web application server performance.

This chapter describes:

➤ Ariba Graph

➤ ATG Dynamo Graph

➤ BroadVision Graph

➤ ColdFusion Graph

➤ Fujitsu INTERSTAGE Graph

➤ iPlanet (NAS) Graph

➤ Microsoft Active Server Pages (ASP) Graph

➤ Oracle9iAS HTTP Server Graph

➤ SilverStream Graph

➤ WebLogic (SNMP) Graph

➤ WebLogic (JMX) Graph

➤ WebSphere Graph

➤ WebSphere (EPM) Graph

About Web Application Server Resource Graphs

Web Application Server Resource graphs provide you with resource usage information about the Ariba, ATG Dynamo, BroadVision, ColdFusion, Fujitsu INTERSTAGE, iPlanet (NAS), Microsoft ASP, Oracle9iAS HTTP, SilverStream, WebLogic (SNMP), WebLogic (JMX), and WebSphere application servers.

In order to obtain data for these graphs, you need to activate the online monitor for the application server and specify which resources you want to measure before running the scenario. For information on activating and configuring the Web Application Server Resource monitors, see the *LoadRunner Controller User's Guide*.

When you open a Web Application Server Resource graph, you can filter it to show only the relevant application. When you need to analyze other applications, you can change the filter conditions and display the desired resources.

In order to display all the measurements on a single graph, Analysis may scale them. The Legend tab indicates the scale factor for each resource. To obtain the true value, multiply the scale factor by the displayed value. For more information on scaled measurements, see the example in "About Web Server Resource Graphs" on page 167.

Ariba Graph

The Ariba graph displays statistics about the resource usage on the Ariba server during the scenario run. The x-axis represents the time that has elapsed since the start of the scenario run. The y-axis represents the resource usage.

Note: To obtain data for this graph, you need to enable the Ariba online monitor (from the Controller) and select the default measurements you want to display, before running the scenario.

The following tables describe the default measurements available for the Ariba server:

Core Server Performance Counters

Measurement	Description
Total Connections	The cumulative number of concurrent user connections since Ariba Buyer was started.
Requisitions Finished	The instantaneous reading of the length of the worker queue at the moment this metric is obtained. The longer the worker queue, the more user requests are delayed for processing.
Worker Queue Length	The instantaneous reading of the length of the worker queue at the moment this metric is obtained. The longer the worker queue, the more user requests are delayed for processing.
Concurrent Connections	The instantaneous reading of the number of concurrent user connections at the moment this metric is obtained
Total Memory	The instantaneous reading of the memory (in KB) being used by Ariba Buyer at the moment this metric is obtained

Measurement	Description
Free Memory	The instantaneous reading of the reserved memory (in bytes) that is not currently in use at the moment this metric is obtained
Up Time	The amount of time (in hours and minutes) that Ariba Buyer has been running since the previous time it was started
Number of Threads	The instantaneous reading of the number of server threads in existence at the moment this metric is obtained
Number of Cached Objects	The instantaneous reading of the number of Ariba Buyer objects being held in memory at the moment this metric is obtained
Average Session Length	The average length of the user sessions (in seconds) of all users who logged out since previous sampling time. This value indicates on average how long a user stays connected to server.
Average Idle Time	The average idle time (in seconds) for all the users who are active since previous sampling time. The idle time is the period of time between two consecutive user requests from the same user.
Approves	The cumulative count of the number of approves that happened during the sampling period. An Approve consists of a user approving one Approvable.
Submits	The cumulative count of the number of Approvables submitted since previous sampling time
Denies	The cumulative count of the number of submitted Approvables denied since previous sampling time
Object Cache Accesses	The cumulative count of accesses (both reads and writes) to the object cache since previous sampling time
Object Cache Hits	The cumulative count of accesses to the object cache that are successful (cache hits) since previous sampling time

System Related Performance Counters

Measurement	Description
Database Response Time	The average response time (in seconds) to the database requests since the previous sampling time
Buyer to DB server Traffic	The cumulative number of bytes that Ariba Buyer sent to DB server since the previous sampling time.
DB to Buyer server Traffic	The cumulative number of bytes that DB server sent to Ariba Buyer since the previous sampling time
Database Query Packets	The average number of packets that Ariba Buyer sent to DB server since the previous sampling time
Database Response Packets	The average number of packets that DB server sent to Ariba Buyer since the previous sampling time

ATG Dynamo Graph

The ATG Dynamo graph displays statistics about the resource usage on the ATG Dynamo server during the scenario run. The x-axis represents the time that has elapsed since the start of the scenario run. The y-axis represents the resource usage.

Note: To obtain data for this graph, you need to enable the ATG Dynamo online monitor (from the Controller) and select the default measurements you want to display, before running the scenario.

The following tables describe the measurements that are available for the ATG Dynamo server:

d3System Measurement	Description
sysTotalMem	The total amount of memory currently available for allocating objects, measured in bytes
sysFreeMem	An approximation of the total amount of memory currently available for future allocated objects, measured in bytes
sysNumInfoMsgs	The number of system global info messages written
sysNumWarningMsgs	The number of system global warning messages written
sysNumErrorMsgs	The number of system global error messages written

d3LoadManagement Measurement	Description
lmIsManager	True if the Dynamo is running a load manager
lmManagerIndex	Returns the Dynamo's offset into the list of load managing entities

d3LoadManagement Measurement	Description
lmIsPrimaryManager	True if the load manager is an acting primary manager
lmServicingCMs	True if the load manager has serviced any connection module requests in the amount of time set as the connection module polling interval
lmCMLDRPPort	The port of the connection module agent
lmIndex	A unique value for each managed entity
lmSNMPPort	The port for the entry's SNMP agent
lmProbability	The probability that the entry will be given a new session
lmNewSessions	Indicates whether or not the entry is accepting new sessions, or if the load manager is allowing new sessions to be sent to the entry. This value is inclusive of any override indicated by lmNewSessionOverride.
lmNewSessionOverride	The override set for whether or not a server is accepting new sessions

d3SessionTracking Measurement	Description
stCreatedSessionCnt	The number of created sessions
stValidSessionCnt	The number of valid sessions
stRestoredSessionCnt	The number of sessions migrated to the server
StDictionaryServerStatus	d3Session Tracking

d3DRPServer Measurement	Description
drpPort	The port of the DRP server
drpTotalReqsServed	Total number of DRP requests serviced
drpTotalReqTime	Total service time in msecs for all DRP requests
drpAvgReqTime	Average service time in msecs for each DRP request
drpNewessions	True if the Dynamo is accepting new sessions

d3DBConnPooling Measurement	Description
dbPoolsEntry	A pooling service entry containing information about the pool configuration and current status
dbIndex	A unique value for each pooling service
dbPoolID	The name of the DB connection pool service
dbMinConn	The minimum number of connections pooled
dbMaxConn	The maximum number of connections pooled
dbMaxFreeConn	The maximum number of free pooled connections at a time
dbBlocking	Indicates whether or not the pool is to block out check outs
dbConnOut	Returns the number of connections checked out

d3DBConnPooling Measurement	Description
dbFreeResources	Returns the number of free connections in the pool. This number refers to connections actually created that are not currently checked out. It does not include how many more connections are allowed to be created as set by the maximum number of connections allowed in the pool.
dbTotalResources	Returns the number of total connections in the pool. This number refers to connections actually created and is not an indication of how many more connections may be created and used in the pool.

BroadVision Graph

The BroadVision graph displays performance statistics for all the servers/services available on the BroadVision application during the scenario run. The x-axis represents the time that has elapsed since the start of the scenario run. The y-axis represents the resource usage.

Note: To obtain data for this graph, you need to enable the BroadVision online monitor (from the Controller) and select the default measurements you want to display, before running the scenario.

The following table describes all the servers/services that are available:

Server	Multiple Instances	Description
adm_srv	No	One-To-One user administration server. There must be one.
alert_srv	No	Alert server handles direct IDL function calls to the Alert system.
bvconf_srv	No	One-To-One configuration management server. There must be one.
cmsdb	Yes	Visitor management database server.
cntdb	Yes	Content database server.
deliv_smtp_d	Yes	Notification delivery server for e-mail type messages. Each instance of this server must have its own ID, numbered sequentially starting with "1".
deliv_comp_d	No	Notification delivery completion processor.
extdbacc	Yes	External database accessor. You need at least one for each external data source.

Server	Multiple Instances	Description
genericdb	No	Generic database accessor handles content query requests from applications, when specifically called from the application. This is also used by the One-To-One Command Center.
hostmgr	Yes	Defines a host manager process for each machine that participates in One-To-One, but doesn't run any One-To-One servers. For example, you need a hostmgr on a machine that runs only servers. You don't need a separate hostmgr on a machine that already has one of the servers in this list.
g1_ofbe_srv	No	Order fulfillment back-end server.
g1_ofdb	Yes	Order fulfillment database server.
g1_om_srv	No	Order management server.
pmtassign_d	No	The payment archiving daemon routes payment records to the archives by periodically checking the invoices table, looking for records with completed payment transactions, and then moving those records into an archive table.
pmthdlr_d	Yes	For each payment processing method, you need one or more authorization daemons to periodically acquire the authorization when a request is made.
pmtsettle_d	Yes	Payment settlement daemon periodically checks the database for orders of the associated payment processing method that need to be settled, and then authorizes the transactions.
sched_poll_d	No	Notification schedule poller scans the database tables to determine when a notification must be run.
sched_srv	Yes	Notification schedule server runs the scripts that generate the visitor notification messages.

Performance Counters

Performance counters for each server/service are divided into logical groups according to the service type.

The following section describes all the available counters under each group. Please note that for some services the number of counters for the same group can be different.

Counter groups:

➤ BV_DB_STAT

➤ BV_SRV_CTRL

➤ BV_SRV_STAT

➤ NS_STAT

➤ BV_CACHE_STAT

➤ JS_SCRIPT_CTRL

➤ JS_SCRIPT_STAT

BV_DB_STAT

The database accessor processes have additional statistics available from the BV_DB_STAT memory block. These statistics provide information about database accesses, including the count of selects, updates, inserts, deletes, and stored procedure executions.

➤ DELETE - Count of deletes executions

➤ INSERT - Count of inserts executions

➤ SELECT - Count of selects executions

➤ SPROC - Count of stored procedure executions.

➤ UPDATE - Count of updates executions

BV_SRV_CTRL

➤ SHUTDOWN

NS_STAT

The NS process displays the namespace for the current One-To-One environment, and optionally can update objects in a name space.

➤ Bind

➤ List

➤ New

➤ Rebnd

➤ Rsolv

➤ Unbnd

BV_SRV_STAT

The display for Interaction Manager processes includes information about the current count of sessions, connections, idle sessions, threads in use, and count of CGI requests processed.

➤ **HOST** - Host machine running the process.

➤ **ID** - Instance of the process (of which multiple can be configured in the bv1to1.conf file), or engine ID of the Interaction Manager.

➤ **CGI** - Current count of CGI requests processed.

➤ **CONN** - Current count of connections.

➤ **CPU** - CPU percentage consumed by this process. If a process is using most of the CPU time, consider moving it to another host, or creating an additional process, possibly running on another machine. Both of these specifications are done in the bv1to1.conf file. The CPU % reported is against a single processor. If a server is taking up a whole CPU on a 4 processor machine, this statistic will report 100%, while the Windows NT Task Manager will report 25%. The value reported by this statistic is consistent with "% Processor Time" on the Windows NT Performance Monitor.

➤ **GROUP** - Process group (which is defined in the **bv1to1.conf** file), or Interaction Manager application name.

➤ **STIME** - Start time of server. The start times should be relatively close. Later times might be an indication that a server crashed and was automatically restarted.

➤ **IDL** - Total count of IDL requests received, not including those to the monitor.

➤ **IdlQ**

➤ **JOB**

➤ **LWP** - Number of light-weight processes (threads).

➤ **RSS** - Resident memory size of server process (in Kilobytes).

➤ **STIME** - System start time.

➤ **SESS** - Current count of sessions.

➤ **SYS** - Accumulated system mode CPU time (seconds).

➤ **THR** - Current count of threads.

➤ **USR** - Accumulated user mode CPU time (seconds).

➤ **VSZ** - Virtual memory size of server process (in kilobytes). If a process is growing in size, it probably has a memory leak. If it is an Interaction Manager process, the culprit is most likely a component or dynamic object (though Interaction Manager servers do grow and shrink from garbage collection during normal use).

BV_CACHE_STAT

Monitors the request cache status.

The available counters for each request are:

➤ **CNT- Request_Name-HIT** - Count of requests found in the cache.

➤ **CNT- Request_Name-MAX** - Maximum size of the cache in bytes

➤ **CNT- Request_Name-SWAP** - Count of items that got swapped out of the cache.

➤ **CNT- Request_Name-MISS** - Count of requests that were not in the cache.

➤ **CNT- Request_Name-SIZE** - Count of items currently in the cache.

Cache Metrics

Cache metrics are available for the following items:

➤ **AD**

➤ **ALERTSCHED** - Notification schedules are defined in the BV_ALERTSCHED and BV_MSGSCHED tables. They are defined by the One-To-One Command Center user or by an application.

➤ **CATEGORY_CONTENT**

➤ **DISCUSSION** - The One-To-One discussion groups provide moderated system of messages and threads of messages aligned to a particular topic. Use the Discussion group interfaces for creating, retrieving and deleting individual messages in a discussion group. To create, delete, or retrieve discussion groups, use the generic content management API. The BV_DiscussionDB object provides access to the threads and messages in the discussion group database.

➤ **EXT_FIN_PRODUCT**

➤ **EDITORIAL** - Using the Editorials content module, you can point cast and community cast personalized editorial content, and sell published text on your One-To-One site. You can solicit editorial content, such as investment reports and weekly columns, from outside authors and publishers, and create your own articles, reviews, reports, and other informative media. In addition to text, you can use images, sounds, music, and video presentations as editorial content.

➤ **INCENTIVE** - Contains sales incentives

➤ **MSGSCHED** - Contains the specifications of visitor-message jobs. Notification schedules are defined in the BV_ALERTSCHED and BV_MSGSCHED tables. They are defined by the One-To-One Command Center user or by an application.

➤ **MSGSCRIPT** - Contains the descriptions of the JavaScripts that generate visitor messages and alert messages. Contains the descriptions of the JavaScripts that generate targeted messages and alert messages. Use the Command Center to add message script information to this table by selecting the Visitor Messages module in the Notifications group. For more information, see the Command Center User's Guide.

➤ **PRODUCT** - BV_PRODUCT contains information about the products that a visitor can purchase.

➤ **QUERY** - BV_QUERY contains queries.

➤ **SCRIPT** - BV_SCRIPT contains page scripts.

➤ **SECURITIES**

➤ **TEMPLATE** - The Templates content module enables you to store in the content database any BroadVision page templates used on your One-To-One site. Combining BroadVision page templates with BroadVision dynamic objects in the One-To-One Design Center application is one way for site developers to create One-To-One Web sites. If your developers use these page templates, you can use the Command Center to enter and manage them in your content database. If your site doesn't use BroadVision page template, you will not use this content module.

JS_SCRIPT_CTRL

➤ CACHE

➤ DUMP

➤ FLUSH

➤ METER

➤ TRACE

JS_SCRIPT_STAT

➤ ALLOC

➤ ERROR

➤ FAIL

➤ JSPPERR

➤ RELEASE

➤ STOP

➤ SUCC

➤ SYNTAX

ColdFusion Graph

The ColdFusion graph displays statistics about the resource usage on the ColdFusion server during the scenario run. The x-axis represents the time that has elapsed since the start of the scenario run. The y-axis represents the resource usage.

Note: To obtain data for this graph, you need to enable the ColdFusion online monitor (from the Controller) and select the default measurements you want to display, before running the scenario.

The following measurements (under the ColdFusion Server object) are most commonly used when monitoring Allaire's ColdFusion server:

Measurement	Description
Avg. Database Time (msec)	The running average of the amount of time, in milliseconds, that it takes ColdFusion to process database requests.
Avg. Queue Time (msec)	The running average of the amount of time, in milliseconds, that requests spent waiting in the ColdFusion input queue before ColdFusion began to process the request.
Avg Req Time (msec)	The running average of the total amount of time, in milliseconds, that it takes ColdFusion to process a request. In addition to general page processing time, this value includes both queue time and database processing time.
Bytes In/sec	The number of bytes per second sent to the ColdFusion server.
Bytes Out/sec	The number of bytes per second returned by the ColdFusion server.
Cache Pops	Cache pops.
Database Hits/sec	This is the number of database hits generated per second by the ColdFusion server.

Measurement	Description
Page Hits/sec	This is the number of Web pages processed per second by the ColdFusion server.
Queued Requests	The number of requests currently waiting to be processed by the ColdFusion server.
Running Requests	The number of requests currently being actively processed by the ColdFusion server.
Timed Out Requests	The number of requests that timed out due to inactivity timeouts.

Fujitsu INTERSTAGE Graph

The Fujitsu INTERSTAGE graph displays statistics about the resource usage on the Fujitsu server during the scenario run. The x-axis represents the time that has elapsed since the start of the scenario run. The y-axis represents the resource usage.

Note: To obtain data for this graph, you need to enable the Fujitsu INTERSTAGE online monitor (from the Controller) and select the default measurements you want to display, before running the scenario.

The following default measurements are available for the Fujitsu INTERSTAGE server:

Measurement	Description
IspSumObjectName	The object name of the application for which performance information is measured
IspSumExecTimeMax	The maximum processing time of the application within a certain period of time
IspSumExecTimeMin	The minimum processing time of the application within a certain period of time
IspSumExecTimeAve	The average processing time of the application within a certain period of time
IspSumWaitTimeMax	The maximum time required for INTERSTAGE to start an application after a start request is issued
IspSumWaitTimeMin	The minimum time required for INTERSTAGE to start an application after a start request is issued
IspSumWaitTimeAve	The average time required for INTERSTAGE to start an application after a start request is issued
IspSumRequestNum	The number of requests to start an application
IspSumWaitReqNum	The number of requests awaiting application activation

iPlanet (NAS) Graph

This Web Application Server Resource graph displays statistics about the resource usage on the iPlanet (NAS) Web application server during the scenario run. The x-axis represents the time that has elapsed since the start of the scenario run. The y-axis represents the resource usage.

Note: To obtain data for this graph, you need to enable the iPlanet (NAS) online monitor (from the Controller) and select the default measurements you want to display, before running the scenario.

The following default measurements are available for the iPlanet (NAS) server:

Netscape Performance Counters

Measurement	Description
nasKesEngConn Retries	The maximum number of times the administration server will try to connect to an engine
nasKesEngMax Restart	The maximum number of times the administration server will restart an engine after a failure
nasKesEngAutoStart	Start all the engines at startup of the administration server
nasKesConfig HeartBeat	Heart Beat

KES Performance Counters

Measurement	Description
nasKesId	The id of the KES this engine belongs to
nasKesMinThread	The default minimum number of threads per engine
nasKesMaxThread	The default maximum number of threads per engine

Measurement	Description
nasKesLoadBalancer Disable	Enable or Disable the load balancer service
nasKesCpuLoad	The total cpu usage on this host
nasKesDiskLoad	The total disk usage on this host
nasKesMemLoad	The total memory usage on this host
nasKesRequestLoad	The number of requests on this NAS
nasKesCpuLoad Factor	The relative importance of CPU usage in computing the server load. This number is specified as a percent. The sum of all server load factors, CPULoad, DiskLoad, MemLoad and ExecReqs must equal 100%
nasKesDiskLoad Factor	The relative importance of Disk usage in computing the server load. This number is specified as a percent. The sum of all server load factors, CPULoad, DiskLoad, MemLoad and ExecReqs must equal 100%
nasKesMemLoad Factor	The relative importance of Memory usage in computing the server load. This number is specified as a percent. The sum of all server load factors, CPULoad, DiskLoad, MemLoad and ExecReqs must equal 100%
nasKesAppLogics RunningFactor	The relative importance of the number of times an AppLogic is run in computing the AppLogic execution performance. This figure is specified as a percent. The sum of all agent load factors, ResultCached, AvgExecTime, LastExecTime, and ServerLoad must equal 100%
nasKesResults CachedFactor	The relative importance of the cached results of an AppLogic in computing the AppLogic execution performance. This figure is specified as a percent. The sum of all agent load factors, ResultCached, AvgExecTime, LastExecTime, and ServerLoad must equal 100%

Measurement	Description
nasKesAvgExecTime Factor	The relative importance of the average execution time of an AppLogic in computing the AppLogic execution performance. This figure is specified as a percent. The sum of all agent load factors, ResultCached, AvgExecTime, LastExecTime, and ServerLoad must equal 100%
nasKesLastExec TimeFactor	The relative importance of the last execution time of an AppLogic in computing the AppLogic execution performance. This figure is specified as a percent. The sum of all agent load factors, ResultCached, AvgExecTime, LastExecTime, and ServerLoad must equal 100%
nasKesHitsFactor	The relative importance of the number of AppLogics running in computing the AppLogic execution performance. This figure is specified as a percent. The sum of all agent load factors, ResultCached, AvgExecTime, LastExecTime, and ServerLoad must equal 100%
nasKesServerLoad Factor	The relative importance of the server load (computed using the four server load factors) in computing APpLogic execution performance. The sum of all agent load factors, ResultCached, AvgExecTime, LastExecTime, and ServerLoad must equal 100%.
nasKesBroadcast Interval	The length of time in seconds, between each broadcast attempt from the load balancer daemon
nasKesApplogic BroadcastInterval	The length of time in seconds, between each broadcast of AppLogics load information across all the servers in the cluster. This should be greater than nasKesBroacastInterval
nasKesServer BroadcastInterval	The length of time in seconds, between each broadcast of server load information across all the servers in the cluster. This should be greater than nasKesBroacastInterval

Measurement	Description
nasKesServerLoad UpdateInterval	The length of time in seconds between each update of server load informations. A server load update applies the server load data that has been sampled up until the moment when the update occurs
nasKesCpuLoad UpdateInterval	The length of time, in seconds, between each sampling of CPU usage.
nasKesDiskLoad UpdateInterval	The length of time, in seconds, between each sampling of disk usage.
nasKesMemLoad UpdateInterval	The length of time, in seconds, between each sampling of memory thrashes.
nasKesTotalReqs UpdateInterval	The length of time, in seconds, between each sampling of the number of requests.
nasKesMaxHops	The maximum number of times a request can be load
nasKesODBCReq MinThread	The minimum number of threads reserved to process asynchronous requests
nasKesODBCReq MaxThread	The maximum number of threads reserved to process asynchronous requests
nasKesODBCCache MaxConns	The maximum number of connections opened between NAS and the database
nasKesODBCCache FreeSlots	The minimum number of cached connections established between NAS and the database
nasKesODBCCache Timeout	The time after which an idle connection is dropped
nasKesODBCCache Interval	The interval in seconds at which the cache cleaner will try to disconnect connections already idle for longer than the specified timeout
nasKesODBCConn GiveupTime	Maximum time the driver will try to connect to the database
nasKesODBCCache Debug	Turns on the connection cache debug information

Measurement	Description
nasKesODBCResult SetInitRows	The number of rows fetched at once from the database
nasKesODBCResult SetMaxRows	The maximum number of rows the cached result set can contain
nasKesODBCResult SetMaxSize	The maximum size of result set the driver will cache
nasKesODBCSql Debug	Turns on sql debug information
nasKesODBCEnable Parser	Turns on SQL parsing
nasKesORCLReqMin Thread	The minimum number of threads reserved to process asynchronous requests
nasKesORCLReq MaxThread	The maximum number of threads reserved to process asynchronous requests
nasKesORCLCache MaxConns	The maximum number of connections opened between NAS and the database
nasKesORCLCache FreeSlots	The minimum number of cached connections established between NAS and the database
nasKesORCLCache Timeout	The time after which an idle connection is dropped
nasKesORCLCache Interval	The interval in seconds at which the cache cleaner will try to disconnect connections already idle for longer than the specified timeout
nasKesORCLConn GiveupTime	The maximum time the driver will spend trying to obtain a connection to Oracle
nasKesORCLCache Debug	Turns on the connection cache debug information
nasKesORCLResult SetInitRows	The number of rows fetched at once from the database
nasKesORCLResult SetMaxRows	The maximum number of rows the cached result set can contain

Measurement	Description
nasKesORCLResult SetMaxSize	The maximum size of result set the driver will cache
nasKesORCLSql Debug	Turns on sql debug information
nasKesSYBReqMin Thread	The minimum number of threads reserved to process asynchronous requests
nasKesSYBReqMax Thread	The maximum number of threads reserved to process asynchronous request
nasKesSYBCache MaxConns	The maximum number of connections opened between NAS and the database
nasKesSYBCache FreeSlots	The minimum number of cached connections established between NAS and the database
nasKesSYBCache Timeout	The time after which an idle connection is dropped
nasKesSYBCache Interval	The interval in seconds at which the cache cleaner will try to disconnect connections already idle for longer than the specified timeout
nasKesSYBConn GiveupTime	The maximum time the driver will spend trying to obtain a connection to Sybase before giving up
nasKesSYBCache Debug	Turns on the connection cache debug information
nasKesSYBResultSet InitRows	The number of rows fetched at once from the database
nasKesSYBResultSet MaxRows	The maximum number of rows the cached result set can contain
nasKesSYBResultSet MaxSize	The maximum size of result set the driver will cache

Engine Performance Counters

Measurement	Description
nasEngKesPort	The port of the KXS this engine serves. This is supplied as part of the object id and cannot be modified after creation
nasEngPort	The TCP/IP port this engine is listening on. The port can only be specified at the creation of the engine. It is not allowed to modify it.
nasEngType	Type of the engine, executive(0), java(1000), c++(3000)
nasEngId	The id is an incremental number starting at 0. The id cannot be modified.
nasEngName	The name of this engine. This is an informational string that contains kcs, kxs, or kjs
nasEngNewConsole	Starts each engine in a new console window
nasEngStatus	The status column used to add, remove, enable or disable an engine. To create an engine, one needs to set. This follows rfc1443.
nasEngMinThread	The default minimum number of threads per engine
nasEngMaxThread	The default maximum number of threads per engine
nasEngReqRate	The rate at which requests arrive
nasEngTotalReq	The total number of requests processed since engine startup
nasEngReqNow	The number of requests being processed
nasEngReqWait	The requests waiting to be serviced
nasEngReqReady	The requests that are ready to be serviced
nasEngAvgReqTime	The average request processing time
nasEngThreadNow	Number of threads in use by the request manager
nasEngThreadWait	The number of idle threads
nasEngWebReq Queue	The number of web requests that are queued

Measurement	Description
nasEngFailedReq	The number of requests that failed
nasEngTotalConn	The total number of connections opened
nasEngTotalConn Now	The total number of connections in use
nasEngTotalAccept	The total number of connections listening to incoming requests
nasEngTotalAccept Now	The total number of connections listening to incoming connections in use
nasEngTotalSent	The total number of packets sent
nasEngTotalSent Bytes	The total number of bytes sent
nasEngTotalRecv	The total number of packets received
nasEngTotalRecv Bytes	The total number of bytes received
nasEngBindTotal	The number of AppLogic bound since startup
nasEngBindTotal Cached	The number of AppLogic cached since startup
nasEngTotalThreads	Total number of threads created in this process
nasEngCurrent Threads	Total number of threads in use in this process
nasEngSleeping Threads	Number of threads sleeping in this process
nasEngDAETotal Query	Total number of queries executed since startup
nasEngDAEQuery Now	The number of queries being processed
nasEngDAETotal Conn	The number of logical connections created since startup
nasEngDAEConn Now	The number of logical connections in use

Measurement	Description
nasEngDAECache Count	The number of caches
nasEngODBCQuery Total	Total number of queries executed since startup
nasEngODBC PreparedQueryTotal	Total number of odbc prepared queries executed since startup
nasEngODBCConn Total	Total number of connections opened since startup
nasEngODBCConn Now	Number of connections currently opened
nasEngORCLQuery Total	Total number of queries executed since startup
nasEngORCL PreparedQueryTotal	Total number of prepared queries executed since startup
nasEngORCLConn Total	Total number of connections established with Oracle since startup
nasEngORCLConn Now	Number of connections opened with Oracle now
nasEngSYBQuery Total	Total number of queries the driver processed since startup
nasEngSYBPrepared QueryTotal	Total number of prepared queries processed since startup
nasEngSYBConn Total	Total number of connections opened since startup
nasEngSYBConn Now	Number of SYB connections opened now
nasStatusTrapEntry	The kes definition
nasTrapKesIp Address	The IP Address of KES host
nasTrapKesPort	The port of the main engine of this NAS

Measurement	Description
nasTrapEngPort	The port of the engine generating this event
nasTrapEngState	The port of the engine generating this event

Microsoft Active Server Pages (ASP) Graph

The Microsoft Active Server Pages (ASP) graph displays statistics about the resource usage on the ASP server during the scenario run. The x-axis represents the time that has elapsed since the start of the scenario run. The y-axis represents the resource usage.

Note: To obtain data for this graph, you need to enable the Microsoft ASP online monitor (from the Controller) and select the default measurements you want to display, before running the scenario.

The following default measurements are available for Microsoft Active Server Pages:

Measurement	Description
Errors per Second	The number of errors per second.
Requests Wait Time	The number of milliseconds the most recent request was waiting in the queue.
Requests Executing	The number of requests currently executing.
Requests Queued	The number of requests waiting in the queue for service.
Requests Rejected	The total number of requests not executed because there were insufficient resources to process them.
Requests Not Found	The number of requests for files that were not found.
Requests/sec	The number of requests executed per second.
Memory Allocated	The total amount of memory, in bytes, currently allocated by Active Server Pages.
Errors During Script Run-Time	The number of failed requests due to run time errors.

Measurement	Description
Sessions Current	The current number of sessions being serviced.
Transactions/sec	The number of transactions started per second.

Oracle9iAS HTTP Server Graph

The Oracle9iAS HTTP graph displays statistics about the resource usage on the Oracle9iAS HTTP server during the scenario run. The x-axis represents the time that has elapsed since the start of the scenario run. The y-axis represents the resource usage.

Note: To obtain data for this graph, you need to enable the Oracle9iAS HTTP online monitor (from the Controller), and select the default measurements you want to display, before running the scenario.

The following table describes some of the modules that are available for the Oracle9iAS HTTP server:

Measurement	Description
mod_mime.c	Determines document types using file extensions
mod_mime_magic.c	Determines document types using "magic numbers"
mod_auth_anon.c	Provides anonymous user access to authenticated areas
mod_auth_dbm.c	Provides user authentication using DBM files
mod_auth_digest.c	Provides MD5 authentication
mod_cern_meta.c	Supports HTTP header metafiles
mod_digest.c	Provides MD5 authentication (deprecated by mod_auth_digest)
mod_expires.c	Applies Expires: headers to resources
mod_headers.c	Adds arbitrary HTTP headers to resources
mod_proxy.c	Provides caching proxy abilities
mod_rewrite.c	Provides powerful URI-to-filename mapping using regular expressions
mod_speling.c	Automatically corrects minor typos in URLs
mod_info.c	Provides server configuration information

Measurement	Description
mod_status.c	Displays server status
mod_usertrack.c	Provides user tracking using cookies
mod_dms.c	Provides access to DMS Apache statistics
mod_perl.c	Allows execution of perl scripts
mod_fastcgi.c	Supports CGI access to long-lived programs
mod_ssl.c	Provides SSL support
mod_plsql.c	Handles requests for Oracle stored procedures
mod_isapi.c	Provides Windows ISAPI extension support
mod_setenvif.c	Sets environment variables based on client information
mod_actions.c	Executes CGI scripts based on media type or request method
mod_imap.c	Handles imagemap files
mod_asis.c	Sends files that contain their own HTTP headers
mod_log_config.c	Provides user-configurable logging replacement for mod_log_common
mod_env.c	Passes environments to CGI scripts
mod_alias.c	Maps different parts of the host file system in the document tree, and redirects URLs
mod_userdir.c	Handles user home directories
mod_cgi.c	Invokes CGI scripts
mod_dir.c	Handles the basic directory
mod_autoindex.c	Provides automatic directory listings
mod_include.c	Provides server-parsed documents
mod_negotiation.c	Handles content negotiation
mod_auth.c	Provides user authentication using text files

Measurement	Description
mod_access.c	Provides access control based on the client hostname or IP address
mod_so.c	Supports loading modules (.so on UNIX, .dll on Win32) at run time
mod_oprocmgr.c	Monitors JServ processes and restarts them if they fail
mod_jserv.c	Routes HTTP requests to JServ server processes. Balances load across multiple JServs by distributing new requests in round-robin order
mod_ose.c	Routes requests to the JVM embedded in Oracle's database server
http_core.c	Handles requests for static Web pages

The following table describes the counters that are available for the Oracle9iAS HTTP server:

Measurement	Description
handle.minTime	The minimum time spent in the module handler
handle.avg	The average time spent in the module handler
handle.active	The number of threads currently in the handle processing phase
handle.time	The total amount of time spent in the module handler
handle.completed	The number of times the handle processing phase was completed
request.maxTime	The maximum amount of time required to service an HTTP request
request.minTime	The minimum amount of time required to service an HTTP request
request.avg	The average amount of time required to service an HTTP request

Measurement	Description
request.active	The number of threads currently in the request processing phase
request.time	The total amount of time required to service an HTTP request
request.completed	The number of times the request processing phase was completed
connection.maxTime	The maximum amount of time spent servicing any HTTP connection
connection.minTime	The minimum amount of time spent servicing any HTTP connection
connection.avg	The average amount of time spent servicing HTTP connections
connection.active	The number of connections with currently open threads
connection.time	The total amount of time spent servicing HTTP connections
connection.completed	The number of times the connection processing phase was completed
numMods.value	The number of loaded modules
childFinish.count	The number of times the Apache parent server started a child server, for any reason
childStart.count	The number of times "children" finished "gracefully." There are some ungraceful error/crash cases that are not counted in childFinish.count
Decline.count	The number of times each module declined HTTP requests
internalRedirect.count	The number of times that any module passed control to another module using an "internal redirect"
cpuTime.value	The total CPU time utilized by all processes on the Apache server (measured in CPU milliseconds)

Measurement	Description
heapSize.value	The total heap memory utilized by all processes on the Apache server (measured in kilobytes)
pid.value	The process identifier of the parent Apache process
upTime.value	The amount of time the server been running (measured in milliseconds)

SilverStream Graph

The SilverStream graph displays statistics about the resource usage on the SilverStream server during the scenario run. The x-axis represents the time that has elapsed since the start of the scenario run. The y-axis represents the resource usage.

Note: To obtain data for this graph, you need to enable the SilverStream online monitor (from the Controller) and select the default measurements you want to display, before running the scenario.

The following default measurements are available for the SilverStream server:

Measurement	Description
#Idle Sessions	The number of sessions in the Idle state.
Avg. Request processing time	The average request processing time.
Bytes Sent/sec	The rate at which data bytes are sent from the Web server.
Current load on Web Server	The percentage of load utilized by the SilverStream server, scaled at a factor of 25.
Hits/sec	The HTTP request rate.
Total sessions	The total number of sessions.
Free memory	The total amount of memory in the Java Virtual Machine currently available for future allocated objects.
Total memory	The total amount of memory in the Java Virtual Machine.
Memory Garbage Collection Count	The total number of times the JAVA Garbage Collector has run since the server was started.

Measurement	Description
Free threads	The current number of threads not associated with a client connection and available for immediate use.
Idle threads	The number of threads associated with a client connection, but not currently handling a user request.
Total threads	The total number of client threads allocated.

Note: The SilverStream monitor connects to the Web server in order to gather statistics, and registers one hit for each sampling. The SilverStream graph, therefore, always displays one hit per second, even if no clients are connected to the SilverStream server.

WebLogic (SNMP) Graph

The WebLogic (SNMP) graph displays statistics about the resource usage on the WebLogic (SNMP) server (version 6.0 and earlier) during the scenario run. The x-axis represents the time that has elapsed since the start of the scenario run. The y-axis represents the resource usage.

Note: To obtain data for this graph, you need to enable the WebLogic (SNMP) online monitor (from the Controller) and select the default measurements you want to display, before running the scenario.

The following default measurements are available for the WebLogic (SNMP) server (for versions earlier than 6.0):

Server Table

The Server Table lists all WebLogic (SNMP) servers that are being monitored by the agent. A server must be contacted or be reported as a member of a cluster at least once before it will appear in this table. Servers are only reported as a member of a cluster when they are actively participating in the cluster, or shortly thereafter.

Measurement	Description
ServerState	The state of the WebLogic server, as inferred by the SNMP agent. *Up* implies that the agent can contact the server. *Down* implies that the agent cannot contact the server.
ServerLoginEnable	This value is true if client logins are enabled on the server.
ServerMaxHeapSpace	The maximum heap size for this server, in KB
ServerHeapUsedPct	The percentage of heap space currently in use on the server
ServerQueueLength	The current length of the server execute queue

Measurement	Description
ServerQueueThroughput	The current throughput of execute queue, expressed as the number of requests processed per second
ServerNumEJBDeployment	The total number of EJB deployment units known to the server
ServerNumEJBBeansDeployed	The total number of EJB beans actively deployed on the server

Listen Table

The Listen Table is the set of protocol, IP address, and port combinations on which servers are listening. There will be multiple entries for each server: one for each protocol, ipAddr, port) combination. If clustering is used, the clustering-related MIB objects will assume a higher priority.

Measurement	Description
ListenPort	Port number.
ListenAdminOK	True if admin requests are allowed on this (protocol, ipAddr, port); otherwise false
ListenState	Listening if the (protocol, ipAddr, port) is enabled on the server; not Listening if it is not. The server may be listening but not accepting new clients if its server Login Enable state is false. In this case, existing clients will continue to function, but new ones will not.

ClassPath Table

The ClassPath Table is the table of classpath elements for Java, WebLogic (SNMP) server, and servlets. There are multiple entries in this table for each server. There may also be multiple entries for each path on a server. If clustering is used, the clustering-related MIB objects will assume a higher priority.

Measurement	Description
CPType	The type of CP element: Java, WebLogic, servlet. A Java CPType means the cpElement is one of the elements in the normal Java classpath. A WebLogic CPType means the cpElement is one of the elements in weblogic.class.path. A servlet CPType means the cpElement is one of the elements in the dynamic servlet classpath.
CPIndex	The position of an element within its path. The index starts at 1.

WebLogic (JMX) Graph

The WebLogic (JMX) graph displays statistics about the resource usage on the WebLogic (JMX) server (version 6.0 and higher) during the scenario run. The x-axis represents the time that has elapsed since the start of the scenario run. The y-axis represents the resource usage.

Note: To obtain data for this graph, you need to enable the WebLogic (JMX) online monitor (from the Controller) and select the default measurements you want to display, before running the scenario.

The following default measurements are available for the WebLogic (JMX) server (for versions higher than 6.0):

LogBroadcasterRuntime

Measurement	Description
MessagesLogged	The number of total log messages generated by this instance of the WebLogic server.
Registered	Returns "false" if the MBean represented by this object has been unregistered.
CachingDisabled	Private property that disables caching in proxies.

ServerRuntime

For more information on the measurements contained in each of the following measurement categories, see Mercury Interactive's Load Testing Monitors Web site:
http://www-svca.mercuryinteractive.com/resources/library/technical/loadtesting_monitors/supported.html

➤ ServletRuntime

➤ WebAppComponentRuntime

➤ EJBStatefulHomeRuntime

➤ JTARuntime

➤ JVMRuntime

➤ EJBEntityHomeRuntime.

➤ DomainRuntime

➤ EJBComponentRuntime

➤ DomainLogHandlerRuntime

➤ JDBCConnectionPoolRuntime

➤ ExecuteQueueRuntime

➤ ClusterRuntime

➤ JMSRuntime

➤ TimeServiceRuntime

➤ EJBStatelessHomeRuntime

➤ WLECConnectionServiceRuntime

ServerSecurityRuntime

Measurement	Description
UnlockedUsersTotalCount	Returns the number of times a user has been unlocked on the server
InvalidLoginUsersHighCount	Returns the high-water number of users with outstanding invalid login attempts for the server
LoginAttemptsWhileLockedTotalCount	Returns the cumulative number of invalid logins attempted on the server while the user was locked
Registered	Returns "false" if the MBean represented by this object has been unregistered.
LockedUsersCurrentCount	Returns the number of currently locked users on the server

Measurement	Description
CachingDisabled	Private property that disables caching in proxies.
InvalidLoginAttemptsTotalCount	Returns the cumulative number of invalid logins attempted on the server
UserLockoutTotalCount	Returns the cumulative number of user lockouts done on the server

WebSphere Graph

The WebSphere and WebSphere 4.x - 5.x graphs display statistics about the resource usage on the WebSphere3.x server during the scenario run. The x-axis represents the time that has elapsed since the start of the scenario run. The y-axis represents the resource usage.

Note: To obtain data for this graph, you need to enable the WebSphere online monitor (from the Controller) and select the default measurements you want to display, before running the scenario.

The following measurements are available for the WebSphere 3.x, 4.x, and 5.x servers:

Run-Time Resources

Contains resources related to the Java Virtual Machine run time, as well as the ORB.

Measurement	Description
MemoryFree	The amount of free memory remaining in the Java Virtual Machine
MemoryTotal	The total memory allocated for the Java Virtual Machine
MemoryUse	The total memory in use within the Java Virtual Machine

BeanData

Every home on the server provides performance data, depending upon the type of bean deployed in the home. The top level bean data holds an aggregate of all the containers.

Measurement	Description
BeanCreates	The number of beans created. Applies to an individual bean that is either 'stateful' or 'entity'
EntityBeanCreates	The number of entity beans created
BeanRemoves	The number of entity beans pertaining to a specific bean that have been removed. Applies to an individual bean that is either 'stateful' or 'entity'
EntityBeanRemoves	The number of entity beans removed
StatefulBeanCreates	The number of stateful beans created
StatefulBeanRemoves	The number of stateful bean removed
BeanPassivates	The number of bean passivates pertaining to a specific bean. Applies to an individual bean that is either 'stateful' or 'entity'
EntityBeanPassivates	The number of entity bean passivates
StatefulBeanPassivates	The number of stateful bean passivates
BeanActivates	The number of bean activates pertaining to a specific bean. Applies to an individual bean that is either 'stateful' or 'entity'
EntityBeanActivates	The number of entity bean activates
StatefulBeanActivates	The number of stateful bean activates
BeanLoads	The number of times the bean data was loaded. Applies to entity
BeanStores	The number of times the bean data was stored in the database. Applies to entity

Measurement	Description
BeanInstantiates	The number of times a bean object was created. This applies to an individual bean, regardless of its type.
StatelessBeanInstantiates	The number of times a stateless session bean object was created
StatefulBeanInstantiates	The number of times a stateful session bean object was created
EntityBeanInstantiates	The number of times an entity bean object was created
BeanDestroys	The number of times an individual bean object was destroyed. This applies to any bean, regardless of its type
StatelessBeanDestroys	The number of times a stateless session bean object was destroyed
StatefulBeanDestroys	The number of times a stateful session bean object was destroyed
EntityBeanDestroys	The number of times an entity bean object was destroyed
BeansActive	The average number of instances of active beans pertaining to a specific bean. Applies to an individual bean that is either 'stateful' or 'entity'
EntityBeansActive	The average number of active entity beans
StatefulBeansActive	The average number of active session beans
BeansLive	The average number of bean objects of this specific type that are instantiated but not yet destroyed. This applies to an individual bean, regardless of its type.
StatelessBeansLive	The average number of stateless session bean objects that are instantiated but not yet destroyed
StatefulBeansLive	The average number of stateful session bean objects that are instantiated but not yet destroyed

Measurement	Description
EntityBeansLive	The average number of entity bean objects that are instantiated but not yet destroyed
BeanMethodRT	The average method response time for all methods defined in the remote interface to this bean. Applies to all beans
BeanMethodActive	The average number of methods being processed concurrently. Applies to all beans
BeanMethodCalls	The total number of method calls against this bean

BeanObjectPool

The server holds a cache of bean objects. Each home has a cache and there is therefore one BeanObjectPoolContainer per container. The top level BeanObjectPool holds an aggregate of all the containers data.

Measurement	Description
BeanObjectPoolContainer	The pool of a specific bean type
BeanObject	The pool specific to a home
NumGet	The number of calls retrieving an object from the pool
NumGetFound	The number of calls to the pool that resulted in finding an available bean
NumPuts	The number of beans that were released to the pool
NumPutsDiscarded	The number of times releasing a bean to the pool resulted in the bean being discarded because the pool was full
NumDrains	The number of times the daemon found the pool was idle and attempted to clean it

Measurement	Description
DrainSize	The average number of beans discarded by the daemon during a clean
BeanPoolSize	The average number of beans in the pool

OrbThreadPool

These are resources related to the ORB thread pool that is on the server.

Measurement	Description
ActiveThreads	The average number of active threads in the pool
TotalThreads	The average number of threads in the pool
PercentTimeMaxed	The average percent of the time that the number of threads in the pool reached or exceeded the desired maximum number
ThreadCreates	The number of threads created
ThreadDestroys	The number of threads destroyed
ConfiguredMaxSize	The configured maximum number of pooled threads

DBConnectionMgr

These are resources related to the database connection manager. The manager consists of a series of data sources, as well as a top-level aggregate of each of the performance metrics.

Measurement	Description
DataSource	Resources related to a specific data source specified by the "name" attribute
ConnectionCreates	The number of connections created
ConnectionDestroys	The number of connections released
ConnectionPoolSize	The average size of the pool, i.e., number of connections

Measurement	Description
ConnectionAllocates	The number of times a connection was allocated
ConnectionWaiters	The average number of threads waiting for a connection
ConnectionWaitTime	The average time, in seconds, of a connection grant
ConnectionTime	The average time, in seconds, that a connection is in use
ConnectionPercentUsed	The average percentage of the pool that is in use
ConnectionPercentMaxed	The percentage of the time that all connections are in use

TransactionData

These are resources that pertain to transactions.

Measurement	Description
NumTransactions	The number of transactions processed
ActiveTransactions	The average number of active transactions
TransactionRT	The average duration of each transaction
BeanObjectCount	The average number of bean object pools involved in a transaction
RolledBack	The number of transactions rolled back
Commited	The number of transactions committed
LocalTransactions	The number of transactions that were local
TransactionMethodCount	The average number of methods invoked as part of each transaction
Timeouts	The number of transactions that timed out due to inactivity timeouts
TransactionSuspended	The average number of times that a transaction was suspended

ServletEngine

These are resources that are related to servlets and JSPs.

Measurement	Description
ServletsLoaded	The number of servlets currently loaded
ServletRequests	The number of requests serviced
CurrentRequests	The number of requests currently being serviced
ServletRT	The average response time for each request
ServletsActive	The average number of servlets actively processing requests
ServletIdle	The amount of time that the server has been idle (i.e., time since last request)
ServletErrors	The number of requests that resulted in an error or an exception
ServletBeanCalls	The number of bean method invocations that were made by the servlet
ServletBeanCreates	The number of bean references that were made by the servlet
ServletDBCalls	The number of database calls made by the servlet
ServletDBConAlloc	The number of database connections allocated by the servlet
SessionLoads	The number of times the servlet session data was read from the database
SessionStores	The number of times the servlet session data was stored in the database
SessionSize	The average size, in bytes, of a session data
LoadedSince	The time that has passed since the server was loaded (UNC time)

Sessions

These are general metrics regarding the HTTP session pool.

Measurement	Description
SessionsCreated	The number of sessions created on the server
SessionsActive	The number of currently active sessions
SessionsInvalidated	The number of invalidated sessions. May not be valid when using sessions in the database mode
SessionLifetime	Contains statistical data of sessions that have been invalidated. Does not include sessions that are still alive

WebSphere (EPM) Graph

The WebSphere (EPM) graph displays statistics about the resource usage on the WebSphere 3.5.x server during the scenario run. The x-axis represents the time that has elapsed since the start of the scenario run. The y-axis represents the resource usage.

Note: To obtain data for this graph, you need to enable the WebSphere (EPM) online monitor (from the Controller) and select the default measurements you want to display, before running the scenario.

The following measurements are available for the WebSphere (EPM) server:

Run Time Resources

Contains resources related to the Java Virtual Machine run time, as well as the ORB.

Measurement	Description
MemoryFree	The amount of free memory remaining in the Java Virtual Machine
MemoryTotal	The total memory allocated for the Java Virtual Machine
MemoryUse	The total memory in use within the Java Virtual Machine

BeanData

Every home on the server provides performance data, depending upon the type of bean deployed in the home. The top level bean data holds an aggregate of all the containers.

Measurement	Description
BeanCreates	The number of beans created. Applies to an individual bean that is either 'stateful' or 'entity'
EntityBeanCreates	The number of entity beans created
BeanRemoves	The number of entity beans pertaining to a specific bean that have been removed. Applies to an individual bean that is either 'stateful' or 'entity'
EntityBeanRemoves	The number of entity beans removed
StatefulBeanCreates	The number of stateful beans created
StatefulBeanRemoves	The number of stateful bean removed
BeanPassivates	The number of bean passivates pertaining to a specific bean. Applies to an individual bean that is either 'stateful' or 'entity'
EntityBeanPassivates	The number of entity bean passivates
StatefulBeanPassivates	The number of stateful bean passivates
BeanActivates	The number of bean activates pertaining to a specific bean. Applies to an individual bean that is either 'stateful' or 'entity'
EntityBeanActivates	The number of entity bean activates
StatefulBeanActivates	The number of stateful bean activates
BeanLoads	The number of times the bean data was loaded. Applies to entity
BeanStores	The number of times the bean data was stored in the database. Applies to entity

Measurement	Description
BeanInstantiates	The number of times a bean object was created. This applies to an individual bean, regardless of its type.
StatelessBeanInstantiates	The number of times a stateless session bean object was created
StatefulBeanInstantiates	The number of times a stateful session bean object was created
EntityBeanInstantiates	The number of times an entity bean object was created
BeanDestroys	The number of times an individual bean object was destroyed. This applies to any bean, regardless of its type
StatelessBeanDestroys	The number of times a stateless session bean object was destroyed
StatefulBeanDestroys	The number of times a stateful session bean object was destroyed
EntityBeanDestroys	The number of times an entity bean object was destroyed
BeansActive	The average number of instances of active beans pertaining to a specific bean. Applies to an individual bean that is either 'stateful' or 'entity'
EntityBeansActive	The average number of active entity beans
StatefulBeansActive	The average number of active session beans
BeansLive	The average number of bean objects of this specific type that are instantiated but not yet destroyed. This applies to an individual bean, regardless of its type.
StatelessBeansLive	The average number of stateless session bean objects that are instantiated but not yet destroyed
StatefulBeansLive	The average number of stateful session bean objects that are instantiated but not yet destroyed

Measurement	Description
EntityBeansLive	The average number of entity bean objects that are instantiated but not yet destroyed
BeanMethodRT	The average method response time for all methods defined in the remote interface to this bean. Applies to all beans
BeanMethodActive	The average number of methods being processed concurrently. Applies to all beans
BeanMethodCalls	The total number of method calls against this bean

BeanObjectPool

The server holds a cache of bean objects. Each home has a cache and there is therefore one BeanObjectPoolContainer per container. The top level BeanObjectPool holds an aggregate of all the containers data.

Measurement	Description
BeanObjectPoolContainer	The pool of a specific bean type
BeanObject	The pool specific to a home
NumGet	The number of calls retrieving an object from the pool
NumGetFound	The number of calls to the pool that resulted in finding an available bean
NumPuts	The number of beans that were released to the pool
NumPutsDiscarded	The number of times releasing a bean to the pool resulted in the bean being discarded because the pool was full
NumDrains	The number of times the daemon found the pool was idle and attempted to clean it

Measurement	Description
DrainSize	The average number of beans discarded by the daemon during a clean
BeanPoolSize	The average number of beans in the pool

OrbThreadPool

These are resources related to the ORB thread pool that is on the server.

Measurement	Description
ActiveThreads	The average number of active threads in the pool
TotalThreads	The average number of threads in the pool
PercentTimeMaxed	The average percent of the time that the number of threads in the pool reached or exceeded the desired maximum number
ThreadCreates	The number of threads created
ThreadDestroys	The number of threads destroyed
ConfiguredMaxSize	The configured maximum number of pooled threads

DBConnectionMgr

These are resources related to the database connection manager. The manager consists of a series of data sources, as well as a top-level aggregate of each of the performance metrics.

Measurement	Description
DataSource	Resources related to a specific data source specified by the "name" attribute
ConnectionCreates	The number of connections created
ConnectionDestroys	The number of connections released
ConnectionPoolSize	The average size of the pool, i.e., number of connections

Measurement	Description
ConnectionAllocates	The number of times a connection was allocated
ConnectionWaiters	The average number of threads waiting for a connection
ConnectionWaitTime	The average time, in seconds, of a connection grant
ConnectionTime	The average time, in seconds, that a connection is in use
ConnectionPercentUsed	The average percentage of the pool that is in use
ConnectionPercentMaxed	The percentage of the time that all connections are in use

TransactionData

These are resources that pertain to transactions.

Measurement	Description
NumTransactions	The number of transactions processed
ActiveTransactions	The average number of active transactions
TransactionRT	The average duration of each transaction
BeanObjectCount	The average number of bean object pools involved in a transaction
RolledBack	The number of transactions rolled back
Commited	The number of transactions committed
LocalTransactions	The number of transactions that were local
TransactionMethodCount	The average number of methods invoked as part of each transaction
Timeouts	The number of transactions that timed out due to inactivity timeouts
TransactionSuspended	The average number of times that a transaction was suspended

ServletEngine

These are resources that are related to servlets and JSPs.

Measurement	Description
ServletsLoaded	The number of servlets currently loaded
ServletRequests	The number of requests serviced
CurrentRequests	The number of requests currently being serviced
ServletRT	The average response time for each request
ServletsActive	The average number of servlets actively processing requests
ServletIdle	The amount of time that the server has been idle (i.e., time since last request)
ServletErrors	The number of requests that resulted in an error or an exception
ServletBeanCalls	The number of bean method invocations that were made by the servlet
ServletBeanCreates	The number of bean references that were made by the servlet
ServletDBCalls	The number of database calls made by the servlet
ServletDBConAlloc	The number of database connections allocated by the servlet
SessionLoads	The number of times the servlet session data was read from the database
SessionStores	The number of times the servlet session data was stored in the database
SessionSize	The average size, in bytes, of a session data
LoadedSince	The time that has passed since the server was loaded (UNC time)

Sessions

These are general metrics regarding the HTTP session pool.

Measurement	Description
SessionsCreated	The number of sessions created on the server
SessionsActive	The number of currently active sessions
SessionsInvalidated	The number of invalidated sessions. May not be valid when using sessions in the database mode
SessionLifetime	Contains statistical data of sessions that have been invalidated. Does not include sessions that are still alive

14

Database Server Resource Graphs

After running a scenario, you can use Database Server Resource graphs to analyze the resource usage of your DB2, Oracle, SQL Server, and Sybase databases.

This chapter describes:

➤ DB2 Graph

➤ Oracle Graph

➤ SQL Server Graph

➤ Sybase Graph

About Database Server Resource Graphs

The Database Server Resource graphs show statistics for several database servers. Currently DB2, Oracle, SQL Server, and Sybase databases are supported. These graphs require that you specify the resources you want to measure *before* running the scenario. For more information, see the section on online monitors in the *LoadRunner Controller User's Guide*.

DB2 Graph

The DB2 graph shows the resource usage on the DB2 database server machine as a function of the elapsed scenario time. The x-axis represents the time that has elapsed since the start of the scenario run. The y-axis represents the resource usage.

Note: In order to monitor the DB2 database server machine, you must first set up the DB2 monitor environment. You then enable the DB2 monitor (from the Controller) by selecting the counters you want the monitor to measure.

The following tables describe the default counters that can be monitored on a DB2 server:

DatabaseManager

Measurement	Description
rem_cons_in	The current number of connections initiated from remote clients to the instance of the database manager that is being monitored.
rem_cons_in_exec	The number of remote applications that are currently connected to a database and are currently processing a unit of work within the database manager instance being monitored.
local_cons	The number of local applications that are currently connected to a database within the database manager instance being monitored.
local_cons_in_exec	The number of local applications that are currently connected to a database within the database manager instance being monitored and are currently processing a unit of work.
con_local_dbases	The number of local databases that have applications connected.

Measurement	Description
agents_registered	The number of agents registered in the database manager instance that is being monitored (coordinator agents and subagents).
agents_waiting_on_token	The number of agents waiting for a token so they can execute a transaction in the database manager.
idle_agents	The number of agents in the agent pool that are currently unassigned to an application and are therefore "idle".
agents_from_pool	The number of agents assigned from the agent pool
agents_created_empty_pool	The number of agents created because the agent pool was empty.
agents_stolen	The number of times that agents are stolen from an application. Agents are stolen when an idle agent associated with an application is reassigned to work on a different application.
comm_private_mem	The amount of private memory that the instance of the database manager has currently committed at the time of the snapshot.
inactive_gw_agents	The number of DRDA agents in the DRDA connections pool that are primed with a connection to a DRDA database, but are inactive.
num_gw_conn_switches	The number of times that an agent from the agents pool was primed with a connection and was stolen for use with a different DRDA database.
sort_heap_allocated	The total number of allocated pages of sort heap space for all sorts at the level chosen and at the time the snapshot was taken.
post_threshold_sorts	The number of sorts that have requested heaps after the sort heap threshold has been reached.

Measurement	Description
piped_sorts_requested	The number of piped sorts that have been requested.
piped_sorts_accepted	The number of piped sorts that have been accepted.

Database

Measurement	Description
appls_cur_cons	Indicates the number of applications that are currently connected to the database.
appls_in_db2	Indicates the number of applications that are currently connected to the database, and for which the database manager is currently processing a request.
total_sec_cons	The number of connections made by a sub-agent to the database at the node.
num_assoc_agents	At the application level, this is the number of sub-agents associated with an application. At the database level, it is the number of sub-agents for all applications.
sort_heap_allocated	The total number of allocated pages of sort heap space for all sorts at the level chosen and at the time the snapshot was taken.
total_sorts	The total number of sorts that have been executed.
total_sort_time	The total elapsed time (in milliseconds) for all sorts that have been executed.
sort_overflows	The total number of sorts that ran out of sort heap and may have required disk space for temporary storage.
active_sorts	The number of sorts in the database that currently have a sort heap allocated.

Measurement	Description
total_hash_joins	The total number of hash joins executed.
total_hash_loops	The total number of times that a single partition of a hash join was larger than the available sort heap space.
hash_join_overflows	The number of times that hash join data exceeded the available sort heap space
hash_join_small_overflows	The number of times that hash join data exceeded the available sort heap space by less than 10%.
pool_data_l_reads	Indicates the number of logical read requests for data pages that have gone through the buffer pool.
pool_data_p_reads	The number of read requests that required I/O to get data pages into the buffer pool.
pool_data_writes	Indicates the number of times a buffer pool data page was physically written to disk.
pool_index_l_reads	Indicates the number of logical read requests for index pages that have gone through the buffer pool.
pool_index_p_reads	Indicates the number of physical read requests to get index pages into the buffer pool.
pool_index_writes	Indicates the number of times a buffer pool index page was physically written to disk.
pool_read_time	Provides the total amount of elapsed time spent processing read requests that caused data or index pages to be physically read from disk to buffer pool.
pool_write_time	Provides the total amount of time spent physically writing data or index pages from the buffer pool to disk.
files_closed	The total number of database files closed.
pool_async_data_reads	The number of pages read asynchronously into the buffer pool.

Measurement	Description
pool_async_data_writes	The number of times a buffer pool data page was physically written to disk by either an asynchronous page cleaner, or a pre-fetcher. A pre-fetcher may have written dirty pages to disk to make space for the pages being pre-fetched.
pool_async_index_writes	The number of times a buffer pool index page was physically written to disk by either an asynchronous page cleaner, or a pre-fetcher. A pre-fetcher may have written dirty pages to disk to make space for the pages being pre-fetched.
pool_async_index_reads	The number of index pages read asynchronously into the buffer pool by a pre-fetcher.
pool_async_read_time	The total elapsed time spent reading by database manager pre-fetchers.
pool_async_write_time	The total elapsed time spent writing data or index pages from the buffer pool to disk by database manager page cleaners.
pool_async_data_read_reqs	The number of asynchronous read requests.
pool_lsn_gap_clns	The number of times a page cleaner was invoked because the logging space used had reached a pre-defined criterion for the database.
pool_drty_pg_steal_clns	The number of times a page cleaner was invoked because a synchronous write was needed during the victim buffer replacement for the database.
pool_drty_pg_thrsh_clns	The number of times a page cleaner was invoked because a buffer pool had reached the dirty page threshold criterion for the database.
prefetch_wait_time	The time an application spent waiting for an I/O server (pre-fetcher) to finish loading pages into the buffer pool.
pool_data_to_estore	The number of buffer pool data pages copied to extended storage.

Measurement	Description
pool_index_to_estore	The number of buffer pool index pages copied to extended storage.
pool_data_from_estore	The number of buffer pool data pages copied from extended storage.
pool_index_from_estore	The number of buffer pool index pages copied from extended storage.
direct_reads	The number of read operations that do not use the buffer pool.
direct_writes	The number of write operations that do not use the buffer pool.
direct_read_reqs	The number of requests to perform a direct read of one or more sectors of data.
direct_write_reqs	The number of requests to perform a direct write of one or more sectors of data.
direct_read_time	The elapsed time (in milliseconds) required to perform the direct reads.
direct_write_time	The elapsed time (in milliseconds) required to perform the direct writes.
cat_cache_lookups	The number of times that the catalog cache was referenced to obtain table descriptor information.
cat_cache_inserts	The number of times that the system tried to insert table descriptor information into the catalog cache.
cat_cache_overflows	The number of times that an insert into the catalog cache failed due the catalog cache being full.
cat_cache_heap_full	The number of times that an insert into the catalog cache failed due to a heap-full condition in the database heap.

Measurement	Description
pkg_cache_lookups	The number of times that an application looked for a section or package in the package cache. At a database level, it indicates the overall number of references since the database was started, or monitor data was reset.
pkg_cache_inserts	The total number of times that a requested section was not available for use and had to be loaded into the package cache. This count includes any implicit prepares performed by the system.
pkg_cache_num_overflows	The number of times that the package cache overflowed the bounds of its allocated memory.
appl_section_lookups	Lookups of SQL sections by an application from its SQL work area.
appl_section_inserts	Inserts of SQL sections by an application from its SQL work area.
sec_logs_allocated	The total number of secondary log files that are currently being used for the database.
log_reads	The number of log pages read from disk by the logger.
log_writes	The number of log pages written to disk by the logger.
total_log_used	The total amount of active log space currently used (in bytes) in the database.
locks_held	The number of locks currently held.
lock_list_in_use	The total amount of lock list memory (in bytes) that is in use.
deadlocks	The total number of deadlocks that have occurred.
lock_escals	The number of times that locks have been escalated from several row locks to a table lock.

Measurement	Description
x_lock_escals	The number of times that locks have been escalated from several row locks to one exclusive table lock, or the number of times an exclusive lock on a row caused the table lock to become an exclusive lock.
lock_timeouts	The number of times that a request to lock an object timed-out instead of being granted.
lock_waits	The total number of times that applications or connections waited for locks.
lock_wait_time	The total elapsed time waited for a lock.
locks_waiting	Indicates the number of agents waiting on a lock.
rows_deleted	The number of row deletions attempted.
rows_inserted	The number of row insertions attempted.
rows_updated	The number of row updates attempted.
rows_selected	The number of rows that have been selected and returned to the application.
int_rows_deleted	The number of rows deleted from the database as a result of internal activity.
int_rows_updated	The number of rows updated from the database as a result of internal activity.
int_rows_inserted	The number of rows inserted into the database as a result of internal activity caused by triggers.
static_sql_stmts	The number of static SQL statements that were attempted.
dynamic_sql_stmts	The number of dynamic SQL statements that were attempted.
failed_sql_stmts	The number of SQL statements that were attempted, but failed.
commit_sql_stmts	The total number of SQL COMMIT statements that have been attempted.

Measurement	Description
rollback_sql_stmts	The total number of SQL ROLLBACK statements that have been attempted.
select_sql_stmts	The number of SQL SELECT statements that were executed.
uid_sql_stmts	The number of SQL UPDATE, INSERT, and DELETE statements that were executed.
ddl_sql_stmts	This element indicates the number of SQL Data Definition Language (DDL) statements that were executed.
int_auto_rebinds	The number of automatic rebinds (or recompiles) that have been attempted.
int_commits	The total number of commits initiated internally by the database manager.
int_rollbacks	The total number of rollbacks initiated internally by the database manager.
int_deadlock_rollbacks	The total number of forced rollbacks initiated by the database manager due to a deadlock. A rollback is performed on the current unit of work in an application selected by the database manager to resolve the deadlock.
binds_precompiles	The number of binds and pre-compiles attempted.

Application

Measurement	Description
agents_stolen	The number of times that agents are stolen from an application. Agents are stolen when an idle agent associated with an application is reassigned to work on a different application.
num_assoc_agents	At the application level, this is the number of sub-agents associated with an application. At the database level, it is the number of sub-agents for all applications.
total_sorts	The total number of sorts that have been executed.
total_sort_time	The total elapsed time (in milliseconds) for all sorts that have been executed.
sort_overflows	The total number of sorts that ran out of sort heap and may have required disk space for temporary storage.
total_hash_joins	The total number of hash joins executed.
total_hash_loops	The total number of times that a single partition of a hash join was larger than the available sort heap space.
hash_join_overflows	The number of times that hash join data exceeded the available sort heap space
hash_join_small_overflows	The number of times that hash join data exceeded the available sort heap space by less than 10%.
pool_data_l_reads	Indicates the number of logical read requests for data pages that have gone through the buffer pool.
pool_data_p_reads	The number of read requests that required I/O to get data pages into the buffer pool.
pool_data_writes	Indicates the number of times a buffer pool data page was physically written to disk.

Measurement	Description
pool_index_l_reads	Indicates the number of logical read requests for index pages that have gone through the buffer pool.
pool_index_p_reads	Indicates the number of physical read requests to get index pages into the buffer pool.
pool_index_writes	Indicates the number of times a buffer pool index page was physically written to disk.
pool_read_time	Provides the total amount of elapsed time spent processing read requests that caused data or index pages to be physically read from disk to buffer pool.
prefetch_wait_time	The time an application spent waiting for an I/O server (pre-fetcher) to finish loading pages into the buffer pool.
pool_data_to_estore	The number of buffer pool data pages copied to extended storage.
pool_index_to_estore	The number of buffer pool index pages copied to extended storage.
pool_data_from_estore	The number of buffer pool data pages copied from extended storage.
pool_index_from_estore	The number of buffer pool index pages copied from extended storage.
direct_reads	The number of read operations that do not use the buffer pool.
direct_writes	The number of write operations that do not use the buffer pool.
direct_read_reqs	The number of requests to perform a direct read of one or more sectors of data.
direct_write_reqs	The number of requests to perform a direct write of one or more sectors of data.
direct_read_time	The elapsed time (in milliseconds) required to perform the direct reads.

Measurement	Description
direct_write_time	The elapsed time (in milliseconds) required to perform the direct writes.
cat_cache_lookups	The number of times that the catalog cache was referenced to obtain table descriptor information.
cat_cache_inserts	The number of times that the system tried to insert table descriptor information into the catalog cache.
cat_cache_overflows	The number of times that an insert into the catalog cache failed due the catalog cache being full.
cat_cache_heap_full	The number of times that an insert into the catalog cache failed due to a heap-full condition in the database heap.
pkg_cache_lookups	The number of times that an application looked for a section or package in the package cache. At a database level, it indicates the overall number of references since the database was started, or monitor data was reset.
pkg_cache_inserts	The total number of times that a requested section was not available for use and had to be loaded into the package cache. This count includes any implicit prepares performed by the system.
appl_section_lookups	Lookups of SQL sections by an application from its SQL work area.
appl_section_inserts	Inserts of SQL sections by an application from its SQL work area.
uow_log_space_used	The amount of log space (in bytes) used in the current unit of work of the monitored application.
locks_held	The number of locks currently held.
deadlocks	The total number of deadlocks that have occurred.

Measurement	Description
lock_escals	The number of times that locks have been escalated from several row locks to a table lock.
x_lock_escals	The number of times that locks have been escalated from several row locks to one exclusive table lock, or the number of times an exclusive lock on a row caused the table lock to become an exclusive lock.
lock_timeouts	The number of times that a request to lock an object timed-out instead of being granted.
lock_waits	The total number of times that applications or connections waited for locks.
lock_wait_time	The total elapsed time waited for a lock.
locks_waiting	Indicates the number of agents waiting on a lock.
uow_lock_wait_time	The total amount of elapsed time this unit of work has spent waiting for locks.
rows_deleted	The number of row deletions attempted.
rows_inserted	The number of row insertions attempted.
rows_updated	The number of row updates attempted.
rows_selected	The number of rows that have been selected and returned to the application.
rows_written	The number of rows changed (inserted, deleted or updated) in the table.
rows_read	The number of rows read from the table.
int_rows_deleted	The number of rows deleted from the database as a result of internal activity.
int_rows_updated	The number of rows updated from the database as a result of internal activity.
int_rows_inserted	The number of rows inserted into the database as a result of internal activity caused by triggers.

Measurement	Description
open_rem_curs	The number of remote cursors currently open for this application, including those cursors counted by 'open_rem_curs_blk'.
open_rem_curs_blk	The number of remote blocking cursors currently open for this application.
rej_curs_blk	The number of times that a request for an I/O block at server was rejected and the request was converted to non-blocked I/O.
acc_curs_blk	The number of times that a request for an I/O block was accepted.
open_loc_curs	The number of local cursors currently open for this application, including those cursors counted by 'open_loc_curs_blk'.
open_loc_curs_blk	The number of local blocking cursors currently open for this application.
static_sql_stmts	The number of static SQL statements that were attempted.
dynamic_sql_stmts	The number of dynamic SQL statements that were attempted.
failed_sql_stmts	The number of SQL statements that were attempted, but failed.
commit_sql_stmts	The total number of SQL COMMIT statements that have been attempted.
rollback_sql_stmts	The total number of SQL ROLLBACK statements that have been attempted.
select_sql_stmts	The number of SQL SELECT statements that were executed.
uid_sql_stmts	The number of SQL UPDATE, INSERT, and DELETE statements that were executed.
ddl_sql_stmts	This element indicates the number of SQL Data Definition Language (DDL) statements that were executed.

Measurement	Description
int_auto_rebinds	The number of automatic rebinds (or recompiles) that have been attempted.
int_commits	The total number of commits initiated internally by the database manager.
int_rollbacks	The total number of rollbacks initiated internally by the database manager.
int_deadlock_rollbacks	The total number of forced rollbacks initiated by the database manager due to a deadlock. A rollback is performed on the current unit of work in an application selected by the database manager to resolve the deadlock.
binds_precompiles	The number of binds and pre-compiles attempted.

Oracle Graph

The Oracle graph displays information from Oracle V$ tables: Session statistics, V$SESSTAT, system statistics, V$SYSSTAT, and other table counters defined by the user in the custom query.

In the following Oracle graph, the V$SYSSTAT resource values are shown as a function of the elapsed scenario time.
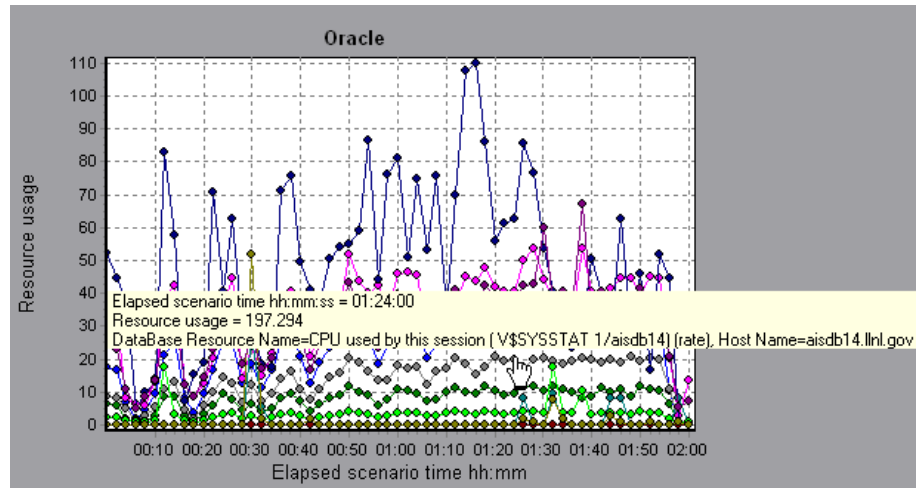
Note: To obtain data for this graph, you need to enable the Oracle online monitor (from the Controller) and select the default measurements you want to display, before running the scenario.

The following measurements are most commonly used when monitoring the Oracle server (from the V$SYSSTAT table):

Measurement	Description
CPU used by this session	This is the amount of CPU time (in 10s of milliseconds) used by a session between the time a user call started and ended. Some user calls can be completed within 10 milliseconds and, as a result, the start and end user-call time can be the same. In this case, 0 milliseconds are added to the statistic. A similar problem can exist in the operating system reporting, especially on systems that suffer from many context switches.
Bytes received via SQL*Net from client	The total number of bytes received from the client over Net8
Logons current	The total number of current logons
Opens of replaced files	The total number of files that needed to be reopened because they were no longer in the process file cache
User calls	Oracle allocates resources (Call State Objects) to keep track of relevant user call data structures every time you log in, parse, or execute. When determining activity, the ratio of user calls to RPI calls gives you an indication of how much internal work gets generated as a result of the type of requests the user is sending to Oracle.
SQL*Net roundtrips to/from client	The total number of Net8 messages sent to, and received from, the client
Bytes sent via SQL*Net to client	The total number of bytes sent to the client from the foreground process(es)
Opened cursors current	The total number of current open cursors

Measurement	Description
DB block changes	Closely related to consistent changes, this statistic counts the total number of changes that were made to all blocks in the SGA that were part of an update or delete operation. These are changes that are generating redo log entries and hence will be permanent changes to the database if the transaction is committed. This statistic is a rough indication of total database work and indicates (possibly on a per-transaction level) the rate at which buffers are being dirtied.
Total file opens	The total number of file opens being performed by the instance. Each process needs a number of files (control file, log file, database file) in order to work against the database.

SQL Server Graph

The SQL Server graph shows the standard Windows resources on the SQL server machine.

The x-axis represents the time that has elapsed since the start of the scenario run. The y-axis represents the resource usage.
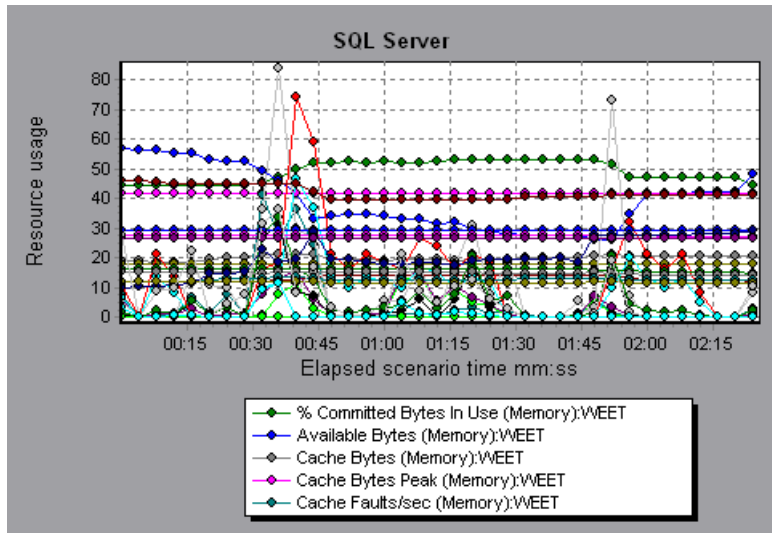
Note: To obtain data for this graph, you need to enable the SQL Server online monitor (from the Controller) and select the default measurements you want to display, before running the scenario.

The following table describes the default counters that can be monitored on version 6.5 of the SQL Server:

Measurement	Description
% Total Processor Time (NT)	The average percentage of time that all the processors on the system are busy executing non-idle threads. On a multi-processor system, if all processors are always busy, this is 100%, if all processors are 50% busy this is 50% and if 1/4th of the processors are 100% busy this is 25%. It can be viewed as the fraction of the time spent doing useful work. Each processor is assigned an Idle thread in the Idle process which consumes those unproductive processor cycles not used by any other threads.
Cache Hit Ratio	The percentage of time that a requested data page was found in the data cache (instead of being read from disk)
I/O - Batch Writes/sec	The number of 2K pages written to disk per second, using Batch I/O. The checkpoint thread is the primary user of Batch I/O.
I/O - Lazy Writes/sec	The number of 2K pages flushed to disk per second by the Lazy Writer
I/O - Outstanding Reads	The number of physical reads pending
I/O - Outstanding Writes	The number of physical writes pending
I/O - Page Reads/sec	The number of physical page reads per second
I/O - Transactions/sec	The number of Transact-SQL command batches executed per second

Measurement	Description
User Connections	The number of open user connections
% Processor Time (Win 2000)	The percentage of time that the processor is executing a non-idle thread. This counter was designed as a primary indicator of processor activity. It is calculated by measuring the time that the processor spends executing the thread of the idle process in each sample interval, and subtracting that value from 100%. (Each processor has an idle thread which consumes cycles when no other threads are ready to run). It can be viewed as the percentage of the sample interval spent doing useful work. This counter displays the average percentage of busy time observed during the sample interval. It is calculated by monitoring the time the service was inactive, and then subtracting that value from 100%.

Sybase Graph

The Sybase graph shows the resource usage on the Sybase database server machine as a function of the elapsed scenario time. The x-axis represents the time that has elapsed since the start of the scenario run. The y-axis represents the resource usage.

Note: In order to monitor the Sybase database server machine, you must first set up the Sybase monitor environment. You then enable the Sybase monitor (from the Controller) by selecting the counters you want the monitor to measure.

The following tables describe the measurements that can be monitored on a Sybase server:

Object	Measurement	Description
Network	**Average packet size (Read)**	Reports the number of network packets received
	Average packet size (Send)	Reports the number of network packets sent
	Network bytes (Read)	Reports the number of bytes received, over the sampling interval
	Network bytes (Read)/sec	Reports the number of bytes received, per second
	Network bytes (Send)	Reports the number of bytes sent, over the sampling interval
	Network bytes (Send)/sec	Reports the number of bytes sent, per second
	Network packets (Read)	Reports the number of network packets received, over the sampling interval
	Network packets (Read)/sec	Reports the number of network packets received, per second
	Network packets (Send)	Reports the number of network packets sent, over the sampling interval
	Network packets (Send)/sec	Reports the number of network packets sent, per second
Memory	**Memory**	Reports the amount of memory, in bytes, allocated for the page cache

Object	Measurement	Description
Disk	**Reads**	Reports the number of reads made from a database device
	Writes	Reports the number of writes made to a database device
	Waits	Reports the number of times that access to a device had to wait
	Grants	Reports the number of times access to a device was granted
Engine	**Server is busy (%)**	Reports the percentage of time during which the Adaptive Server is in a "busy" state
	CPU time	Reports how much "busy" time was used by the engine
	Logical pages (Read)	Reports the number of data page reads, whether satisfied from cache or from a database device
	Pages from disk (Read)	Reports the number of data page reads that could not be satisfied from the data cache
	Pages stored	Reports the number of data pages written to a database device
Stored Procedures	**Executed (sampling period)**	Reports the number of times a stored procedure was executed, over the sampling interval
	Executed (session)	Reports the number of times a stored procedure was executed, during the session
	Average duration (sampling period)	Reports the time, in seconds, spent executing a stored procedure, over the sampling interval
	Average duration (session)	Reports the time, in seconds, spent executing a stored procedure, during the session

Object	Measurement	Description
Locks	**% Requests**	Reports the percentage of successful requests for locks
	Locks count	Reports the number of locks. This is an accumulated value.
	Granted immediately	Reports the number of locks that were granted immediately, without having to wait for another lock to be released
	Granted after wait	Reports the number of locks that were granted after waiting for another lock to be released
	Not granted	Reports the number of locks that were requested but not granted
	Wait time (avg.)	Reports the average wait time for a lock
SqlSrvr	**Locks/sec**	Reports the number of locks. This is an accumulated value.
	% Processor time (server)	Reports the percentage of time that the Adaptive Server is in a "busy" state
	Transactions	Reports the number of committed Transact-SQL statement blocks (transactions)
	Deadlocks	Reports the number of deadlocks
Cache	**% Hits**	Reports the percentage of times that a data page read could be satisfied from cache without requiring a physical page read
	Pages (Read)	Reports the number of data page reads, whether satisfied from cache or from a database device

Object	Measurement	Description
Cache	Pages (Read)/sec	Reports the number of data page reads, whether satisfied from cache or from a database device, per second
	Pages from disk (Read)	Reports the number of data page reads that could not be satisfied from the data cache
	Pages from disk (Read)/sec	Reports the number of data page reads, per second, that could not be satisfied from the data cache
	Pages (Write)	Reports the number of data pages written to a database device
	Pages (Write)/sec	Reports the number of data pages written to a database device, per second
Process	% Processor time (process)	Reports the percentage of time that a process running a given application was in the "Running" state (out of the time that all processes were in the "Running" state)
	Locks/sec	Reports the number of locks, by process. This is an accumulated value.
	% Cache hit	Reports the percentage of times that a data page read could be satisfied from cache without requiring a physical page read, by process
	Pages (Write)	Reports the number of data pages written to a database device, by process
Transaction	Transactions	Reports the number of committed Transact-SQL statement blocks (transactions), during the session

Object	Measurement	Description
Transaction	**Rows (Deleted)**	Reports the number of rows deleted from database tables during the session
	Inserts	Reports the number of insertions into a database table during the session
	Updates	Reports the updates to database tables during the session
	Updates in place	Reports the sum of expensive, in-place and not-in-place updates (everything except updates deferred) during the session
	Transactions/sec	Reports the number of committed Transact-SQL statement blocks (transactions) per second
	Rows (Deleted)/sec	Reports the number of rows deleted from database tables, per second
	Inserts/sec	Reports the number of insertions into a database table, per second
	Updates/sec	Reports the updates to database tables, per second
	Updates in place/sec	Reports the sum of expensive, in-place and not-in-place updates (everything except updates deferred), per second

15

Streaming Media Graphs

After a scenario run, you can use the Streaming Media graphs to analyze RealPlayer Client, RealPlayer Server, and Windows Media Server performance.

This chapter describes the:

➤ Real Client Graph

➤ Real Server Graph

➤ Windows Media Server Graph

➤ Media Player Client Graph

About Streaming Media Graphs

Streaming Media Resource graphs provide you with performance information for the RealPlayer Client, RealPlayer Server, and Windows Media Server machines. Note that in order to obtain data for Streaming Media Resource graphs, you need to install the RealPlayer Client and activate the online monitor for the RealPlayer Server or Windows Media Server before running the scenario. When you set up the online monitor for the RealPlayer Server or Windows Media Server, you indicate which statistics and measurements to monitor. For more information on installing and configuring the Streaming Media Resource monitors, see the *LoadRunner Controller User's Guide*.

In order to display all the measurements on a single graph, Analysis may scale them. The Legend tab indicates the scale factor for each resource. To obtain the true value, multiply the scale factor by the displayed value.

For example, in the following graph the actual value of *RTSP Clients* two minutes into the scenario is 200; 20 multiplied by the scale factor of 10.
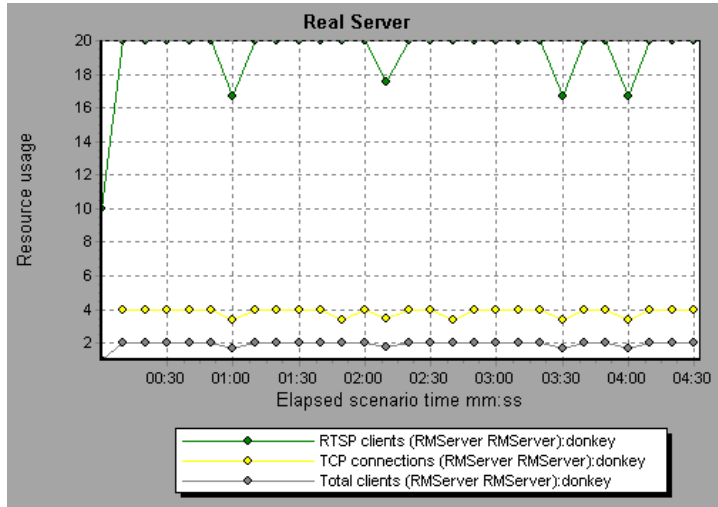
Real Client Graph

The Real Client graph shows statistics on the RealPlayer client machine as a function of the elapsed scenario time.

The x-axis represents the time that has elapsed since the start of the scenario run. The y-axis represents the resource usage.
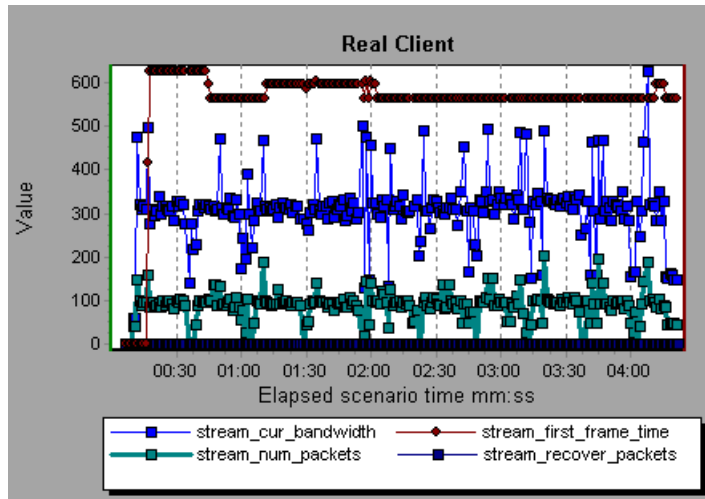
This graph displays the *Total Number of Packets, Number of Recovered Packets, Current Bandwidth,* and *First Frame Time* measurements during the first four and a half minutes of the scenario. Note that the scale factor is the same for all of the measurements.

The following table describes the RealPlayer Client measurements that are monitored:

Measurement	Description
Current Bandwidth (Kbits/sec)	The number of kilobytes in the last second
Buffering Event Time (sec)	The average time spent on buffering
Network Performance	The ratio (percentage) between the current bandwidth and the actual bandwidth of the clip

Measurement	Description
Percentage of Recovered Packets	The percentage of error packets that were recovered
Percentage of Lost Packets	The percentage of packets that were lost
Percentage of Late Packets	The percentage of late packets
Time to First Frame Appearance (sec)	The time for first frame appearance (measured from the start of the replay)
Number of Buffering Events	The average number of all buffering events
Number of Buffering Seek Events	The average number of buffering events resulting from a seek operation
Buffering Seek Time	The average time spent on buffering events resulting from a seek operation
Number of Buffering Congestion Events	The average number of buffering events resulting from network congestion
Buffering Congestion Time	The average time spent on buffering events resulting from network congestion
Number of Buffering Live Pause Events	The average number of buffering events resulting from live pause
Buffering Live Pause Time	The average time spent on buffering events resulting from live pause

Real Server Graph

The Real Server graph shows RealPlayer server statistics as a function of the elapsed scenario time.

The x-axis represents the time that has elapsed since the start of the scenario run. The y-axis represents the resource usage.
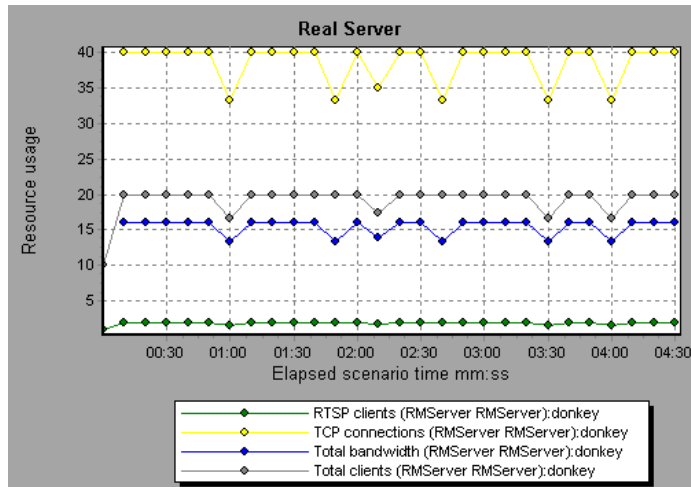
In this graph, the number of *RTSP Clients* remained steady during the first four and a half minutes of the scenario. The *Total Bandwidth* and number of *Total Clients* fluctuated slightly. The number of *TCP Connections* fluctuated more significantly.

Note that the scale factor for the *TCP Connections* and *Total Clients* measurements is 10, and the scale factor for *Total Bandwidth* is 1/1000.
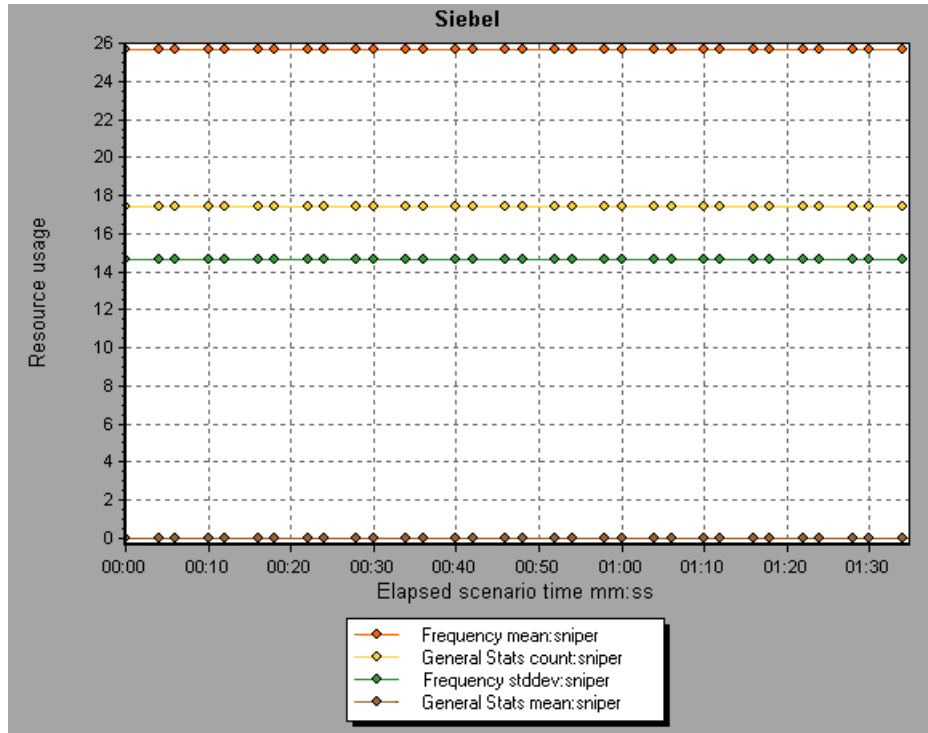

Note: To obtain data for this graph, you need to enable the RealPlayer Server online monitor (from the Controller) and select the default measurements you want to display, before running the scenario.

The following default measurements are available for the RealPlayer Server:

Measurement	Description
Encoder Connections	The number of active encoder connections
HTTP Clients	The number of active clients using HTTP
Monitor Connections	The number of active server monitor connections
Multicast Connections	The number of active multicast connections
PNA Clients	The number of active clients using PNA
RTSP Clients	The number of active clients using RTSP
Splitter Connections	The number of active splitter connections
TCP Connections	The number of active TCP connections
Total Bandwidth	The number of bits per second being consumed
Total Clients	The total number of active clients
UDP Clients	The number of active UDP connections

Windows Media Server Graph

The Windows Media Server graph shows the Windows Media server statistics as a function of the elapsed scenario time. The x-axis represents the time that has elapsed since the start of the scenario run. The y-axis represents the resource usage.
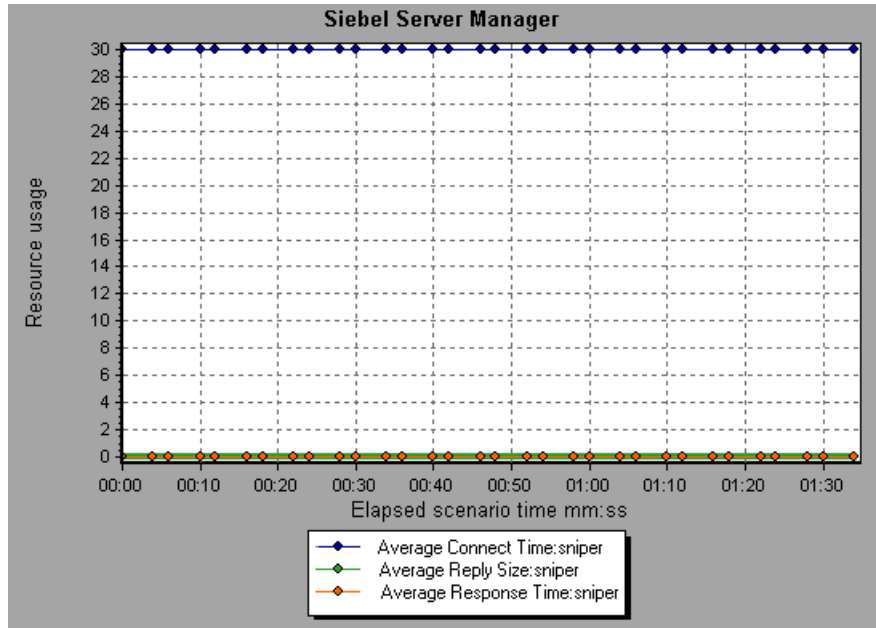

Note: To obtain data for this graph, you need to enable the Windows Media Server online monitor (from the Controller) and select the default measurements you want to display, before running the scenario.

The following default measurements are available for the Windows Media Server:

Measurement	Description
Active Live Unicast Streams (Windows)	The number of live unicast streams that are being streamed
Active Streams	The number of streams that are being streamed
Active TCP Streams	The number of TCP streams that are being streamed
Active UDP Streams	The number of UDP streams that are being streamed
Aggregate Read Rate	The total, aggregate rate (bytes/sec) of file reads
Aggregate Send Rate	The total, aggregate rate (bytes/sec) of stream transmission
Connected Clients	The number of clients connected to the server
Connection Rate	The rate at which clients are connecting to the server
Controllers	The number of controllers currently connected to the server
HTTP Streams	The number of HTTP streams being streamed
Late Reads	The number of late read completions per second

Measurement	Description
Pending Connections	The number of clients that are attempting to connect to the server, but are not yet connected. This number may be high if the server is running near maximum capacity and cannot process a large number of connection requests in a timely manner.
Stations	The number of station objects that currently exist on the server
Streams	The number of stream objects that currently exist on the server
Stream Errors	The cumulative number of errors occurring per second

Media Player Client Graph

The Media Player Client graph shows statistics on the Windows Media Player client machine as a function of the elapsed scenario time.

The x-axis represents the time that has elapsed since the start of the scenario run. The y-axis represents the resource usage.
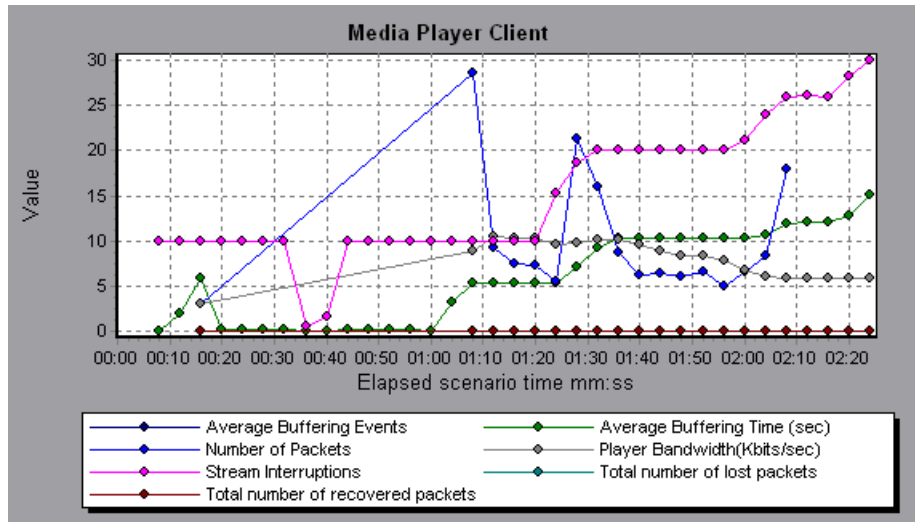
In this graph, the *Total number of recovered packets* remained steady during the first two and a half minutes of the scenario. The *Number of Packets* and *Stream Interruptions* fluctuated significantly. The *Average Buffering Time* increased moderately, and the *Player Bandwidth* increased and then decreased moderately.

Note that the scale factor for the *Stream Interruptions* and *Average Buffering Events* measurements is 10, and the scale factor for *Player Bandwidth* is 1/10.

The following table describes the Media Player Client measurements that are monitored:

Measurement	Description
Average Buffering Events	The number of times Media Player Client had to buffer incoming media data due to insufficient media content
Average Buffering Time (sec)	The time spent by Media Player Client waiting for sufficient amount of media data in order to continue playing media clip
Current bandwidth (Kbits/sec)	The number of kbits per second received
Number of Packets	The number of packets sent by server for a particular media clip
Stream Interruptions	The number of interruptions encountered by Media Player Client while playing a media clip. This measurement includes the number of times Media Player Client had to buffer incoming media data, and any errors that occurred during playback.
Stream Quality (Packet-level)	The percentage ratio of packets received to total packets
Stream Quality (Sampling-level)	The percentage of stream samples received on time (no delays in reception)
Total number of recovered packets	The number of lost packets that were recovered. This value is only relevant during network playback.
Total number of lost packets	The number of lost packets that were not recovered. This value is only relevant during network playback.

16

ERP/CRM Server Resource Graphs

After a scenario run, you can use the ERP/CRM server resource monitor graphs to analyze ERP/CRM server resource performance.

This chapter describes the:

➤ SAP Graph

➤ SAP Portal Graph

➤ Siebel Web Server Graph

➤ Siebel Server Manager Graph

About ERP/CRM Server Resource Graphs

ERP/CRM server resource monitor graphs provide you with performance information for ERP/CRM servers. Note that in order to obtain data for these graphs, you need to activate the ERP/CRM server resource online monitor before running the scenario. When you set up the online monitor for ERP/CRM server resources, you indicate which statistics and measurements to monitor. For more information on activating and configuring ERP/CRM server resource monitors, see the *LoadRunner Controller User's Guide*.

SAP Graph

The SAP graph shows the resource usage of a SAP R/3 system server as a function of the elapsed scenario time.

The x-axis represents the time that has elapsed since the start of the scenario run. The y-axis represents the resource usage.
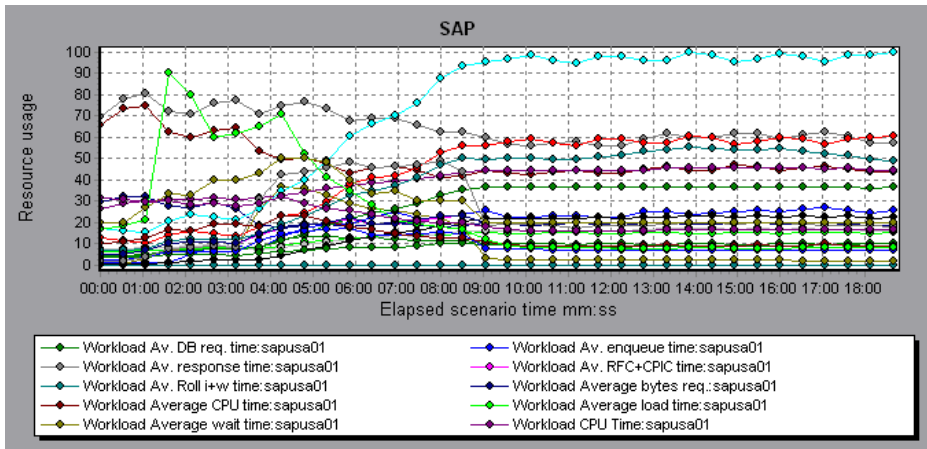
Note: To obtain data for this graph, you need to enable the SAP online monitor (from the Controller) and select the default measurements you want to display, before running the scenario.

Note: There are differences in the scale factor for some of the measurements.

The following are the most commonly monitored counters for a SAP R/3 system server:

Measurement	Description
Average CPU time	The average CPU time used in the work process.
Average response time	The average response time, measured from the time a dialog sends a request to the dispatcher work process, through the processing of the dialog, until the dialog is completed and the data is passed to the presentation layer. The response time between the SAP GUI and the dispatcher is not included in this value.
Average wait time	The average amount of time that an unprocessed dialog step waits in the dispatcher queue for a free work process. Under normal conditions, the dispatcher work process should pass a dialog step to the application process immediately after receiving the request from the dialog step. Under these conditions, the average wait time would be a few milliseconds. A heavy load on the application server or on the entire system causes queues at the dispatcher queue.
Average load time	The time needed to load and generate objects, such as ABAP source code and screen information, from the database.
Database calls	The number of parsed requests sent to the database.
Database requests	The number of logical ABAP requests for data in the database. These requests are passed through the R/3 database interface and parsed into individual database calls. The proportion of database calls to database requests is important. If access to information in a table is buffered in the SAP buffers, database calls to the database server are not required. Therefore, the ratio of calls/requests gives an overall indication of the efficiency of table buffering. A good ratio would be 1:10.
Roll ins	The number of rolled-in user contexts.

Measurement	Description
Roll outs	The number of rolled-out user contexts.
Roll in time	The processing time for roll ins.
Roll out time	The processing time for roll outs.
Roll wait time	The queue time in the roll area. When synchronous RFCs are called, the work process executes a roll out and may have to wait for the end of the RFC in the roll area, even if the dialog step is not yet completed. In the roll area, RFC server programs can also wait for other RFCs sent to them.
Average time per logical DB call	The average response time for all commands sent to the database system (in milliseconds). The time depends on the CPU capacity of the database server, the network, the buffering, and on the input/output capabilities of the database server. Access times for buffered tables are many magnitudes faster and are not considered in the measurement.

SAP Portal Graph

The SAP Portal graph shows the resource usage of your SAP Portal server as a function of the elapsed scenario time.
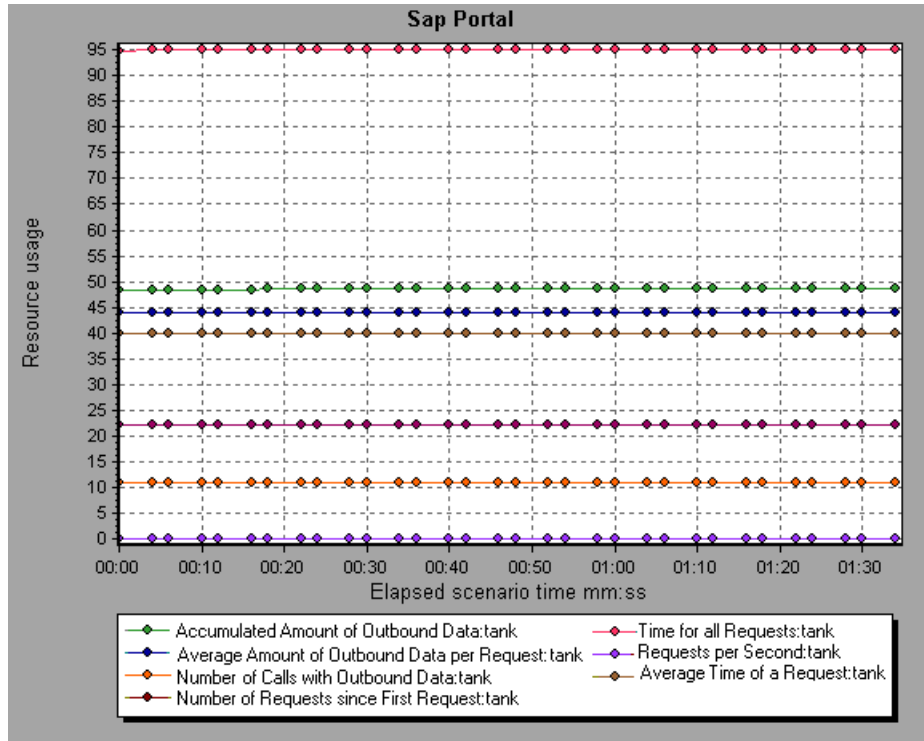
Note: To obtain data for this graph, you need to enable the SAP Portal monitor (from the Controller) and select the default measurements you want to display, before running the scenario.

The following are the monitored counters for a SAP Portal system server:

Measurement	Description
Accumulated Amount of Outbound Data (bytes)	The accumulated amount of outbound data, measured in bytes.
Time for all Requests (ms)	The total time, in milliseconds, taken for processing all requests.
Average Amount of Outbound Data per Request (bytes)	The average amount of outbound data per request, measured in bytes.
Average Number of Component Calls per Request (bytes)	The average number of component calls per request, measured in bytes.
Average Time of a Request (ms)	The average amount of time, in milliseconds, taken to process a request.
Number of Calls with Outbound Data	The total number of calls with outbound data.
Number of Component Calls for all Requests	The total number of component calls for all requests.
Number of Requests since First Request	The total number of requests since the first request was made.
Requests per Second	The number of requests made per second.
Time Stamp of First Request	The time stamp of the first request.

Siebel Web Server Graph

The Siebel Web Server graph shows the resource usage of your Siebel Web Server as a function of the elapsed scenario time.

Note: To obtain data for this graph, you need to enable the Siebel Web Server monitor (from the Controller) and select the default measurements you want to display, before running the scenario.

The following are the monitored counters for a Siebel Web Server:

Measurement	Description
Anonymous sessions requested from the pool	The number of anonymous sessions requested from the pool.
Open Session Time	The time users experience logging on to the system.
Anon Session Removed	The number of anonymous sessions removed from the pool.
Anon Session Available	The number of anonymous sessions available in the pool.
Anonymous sessions returns to the pool	The number of anonymous sessions returned to the pool.
Response Time	The time taken to respond to a user request.
Close Session Time	The time users experience logging off the system.
Request Time	The time taken to process the user request.

Siebel Server Manager Graph

The Siebel Server Manager graph shows the resource usage of your Siebel Server Manager server as a function of the elapsed scenario time.

Note: To obtain data for this graph, you need to enable the Siebel Server Manager monitor (from the Controller) and select the default measurements you want to display, before running the scenario.

The following are the monitored counters for a Siebel Server Manager server:

Measurement	Description
Average Connect Time	The average connection time.
Average Reply Size	The average size of a user reply.
Average Request Size	The average size of a user request.
Average Requests Per Session	The average number of user requests per session.
Average Response Time	The average amount of time that it takes the server to respond to a request.
Average Think Time	The average amount of think time taken to respond to a request.
Avg SQL Execute Time	The average SQL execute time.
Avg SQL Fetch Time	The average SQL fetch time.
Avg SQL Parse Time	The average SQL parse time.
CPU Time	The CPU time used in the work process.
Elapsed Time	The total amount of elapsed time.
Num of DBConn Retries	The number of database connection retries.
Num of DLRbk Retries	The number of DLRbk retries.
Num of Exhausted Retries	The total number of retries that expired.
Number of SQL Executes	The total number of SQL executes.
Number of SQL Fetches	The total number of SQL fetches.
Number of SQL Parses	The total number of SQL parses.
Number of Sleeps	The number of sleeps.

Measurement	Description
Object Manager Errors	The total number of object manager errors.
Reply Messages	The total number of reply messages.
Request Messages	The total number of request messages.
SQL Execute Time	The total SQL execute time.
SQL Fetch Time	The total SQL fetch time.
SQL Parse Time	The total SQL parse time.
Sleep Time	The total sleep time.
Tests Attempted	The number of tests attempted.
Tests Failed	The number of tests that failed.
Tests Successful	The number of tests that were successful.
Total Reply Size	The total reply size, measured in bytes.
Total Request Size	The total request size, measured in bytes.
Total Response Time	The total response time.
Total Tasks	The total number of tasks.
Total Think Time	The total think time.

17

Java Performance Graphs

After a scenario run, you can use the Java performance monitor graphs to analyze the performance of Java™ 2 Platform, Enterprise Edition (J2EE) objects, Enterprise Java Bean (EJB) objects and Java-based applications.

This chapter describes the:

➤ EJB Breakdown

➤ EJB Average Response Time Graph

➤ EJB Call Count Graph

➤ EJB Call Count Distribution Graph

➤ EJB Call Count Per Second Graph

➤ EJB Total Operation Time Graph

➤ EJB Total Operation Time Distribution Graph

➤ J2EE Graph

About Java Performance Graphs

Java performance graphs provide you with performance information for Java™ 2 Platform, Enterprise Edition (J2EE) objects, Enterprise Java Bean (EJB) objects, and Java-based applications, using J2EE and EJB. To obtain data for these graphs, you need to activate the various Java performance monitors before running the scenario.

When you set up the Java performance online monitors, you indicate which statistics and measurements to monitor. For more information on activating and configuring the Java performance monitors, see the *LoadRunner Controller User's Guide*.

EJB Breakdown

The EJB Breakdown summarizes fundamental result data about EJB classes or methods and presents it in table format. Using the EJB Breakdown table, you can quickly identify the Java classes or methods which consume the most time during the test. The table can be sorted by column, and the data can be viewed either by EJB class or EJB method.

The columns, their description, and the graphs that represent them are summarized in the following table:

Legend Column	Description	Graph Name
Average Response Time	Shows how long, on average, a class or method takes to perform.	EJB Average Response Time Graph
Call Count	Specifies the number of times the class or method was invoked.	EJB Call Count Distribution Graph
Total Response Time	Specifies how much time was spent overall on the class or method. This is calculated by multiplying the first two data columns together	EJB Total Operation Time Distribution Graph

Classes are listed in the **EJB Class** column in the form *Class:Host*. In the table above, the class *examples.ejb.basic.beanManaged.AccountBean* took an average of 22.632 milliseconds to execute and was called 1,327 times. Overall, this class took 30 seconds to execute.

To sort the list by a column, select the column heading. The list above is sorted by **Average Response Time** which contains the triangle icon specifying a sort in descending order.

The table initially displays EJB classes, but you can also view the list of EJB methods incorporated within the classes.

Viewing EJB Methods

To view the methods of a selected class:

Select the **EJB Methods** radio button, or double-click on the class row. The methods of the specified class are listed in the **EJB Method** column.

EJB Average Response Time Graph

The EJB Average Response Time graph specifies the average time EJB classes or methods take to perform during the scenario.
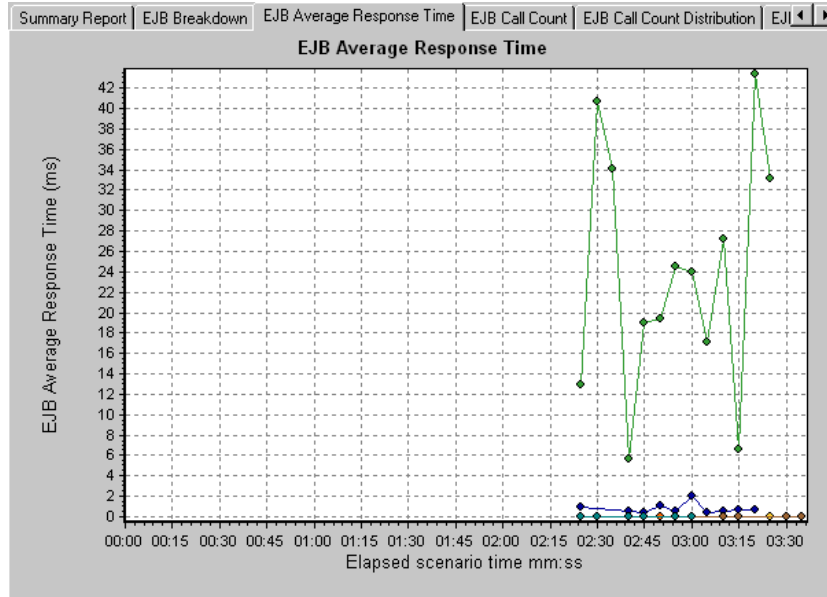
The graph's x-axis indicates the elapsed time from the beginning of the scenario run. The y-axis indicates how much time an EJB class or method takes to execute.

Each class or method is represented by a different colored line on the graph. The legend frame (which is found below the graph) identifies the classes by color:

Color	Scale	Measurement	Minimum	Average	Maximum
☑	1	examples.ejb.basic.beanManaged.AccountBean	0	22.632	311.667
☑	1	examples.ejb.basic.containerManaged.AccountBean	0	0.903	20.667
☑	1	examples.ejb.extensions.readMostly.StockBean	0	0.007	0.083
☑	1	examples.ejb.extensions.readMostly.StockWriterBean	0	0.019	0.031

This legend shows that the green colored line belongs to the EJB class *examples.ejb.basic.beanManaged.AccountBean*. Looking at the graph above, we see that this class has higher response times than all other EJB classes. At 3:20 minutes into the scenario, it records an average response time of 43 milliseconds.

Note that the 43 second data point is an average, taken from all data points recorded within a 5 second interval (the default granularity). You can change the length of this sample interval - see Changing the Granularity of the Data.

Hint: To highlight a specific class line in the graph, select the class row in the legend.

The table at first displays EJB classes, but you can also view the list of EJB methods incorporated within the classes:

Viewing EJB Methods

To view the average response time of the individual methods within an EJB class, you can either use drill-down or filtering techniques. For more information, see "Drilling Down in a Graph," on page 52 and "Filtering and Sorting Graph Data," on page 43.

EJB Call Count Graph

The EJB Call Count graph displays the number of times EJB classes and methods are invoked during the test.
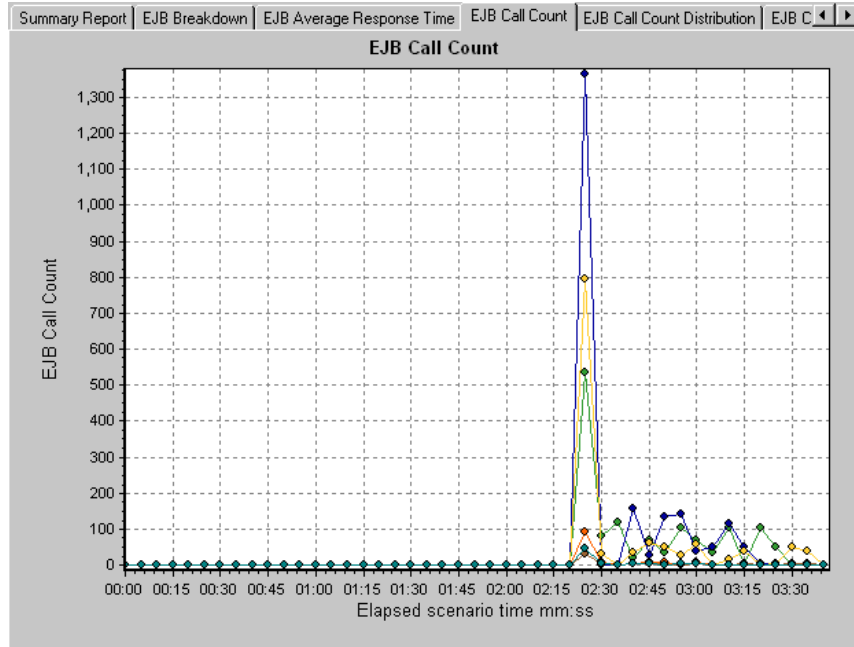
The graph's x-axis indicates the elapsed time from the beginning of the scenario run. The y-axis indicates how many calls were made to an EJB class or method.

Each class or method is represented by a different colored line on the graph. The legend frame (which is found below the graph) identifies the classes by color:

Color	Scale	Measurement	Minimum	Average	Maximum
☑	1	examples.ejb.basic.beanManaged.AccountBean	0	22.632	311.667
☑	1	examples.ejb.basic.containerManaged.AccountBean	0	0.903	20.667
☑	1	examples.ejb.extensions.readMostly.StockBean	0	0.007	0.083
☑	1	examples.ejb.extensions.readMostly.StockWriterBean	0	0.019	0.031

This legend shows that the green colored line belongs to the EJB class *examples.ejb.basic.beanManaged.AccountBean.* Looking at the graph above, we see that calls to this class begins 2:20 minutes into the scenario run. There are 537 calls at the 2:25 minute point.

Hint: To highlight a specific class line in the graph, select the class row in the legend.

Changing Result Granularity

See "Changing the Granularity of the Data," on page 55.

The EJB Call Count Per Second Graph is identical to this graph using a granularity of one second.

Viewing EJB Methods

To view the call count of individual methods within an EJB class, you can either use drill-down or filtering techniques. For more information, see "Drilling Down in a Graph," on page 52 and "Filtering and Sorting Graph Data," on page 43.

EJB Call Count Distribution Graph

The EJB Call Count Distribution graph shows the percentage of calls made to each EJB class compared to all EJB classes. It can also show the percentage of calls made to a specific EJB method compared to other methods within the class.
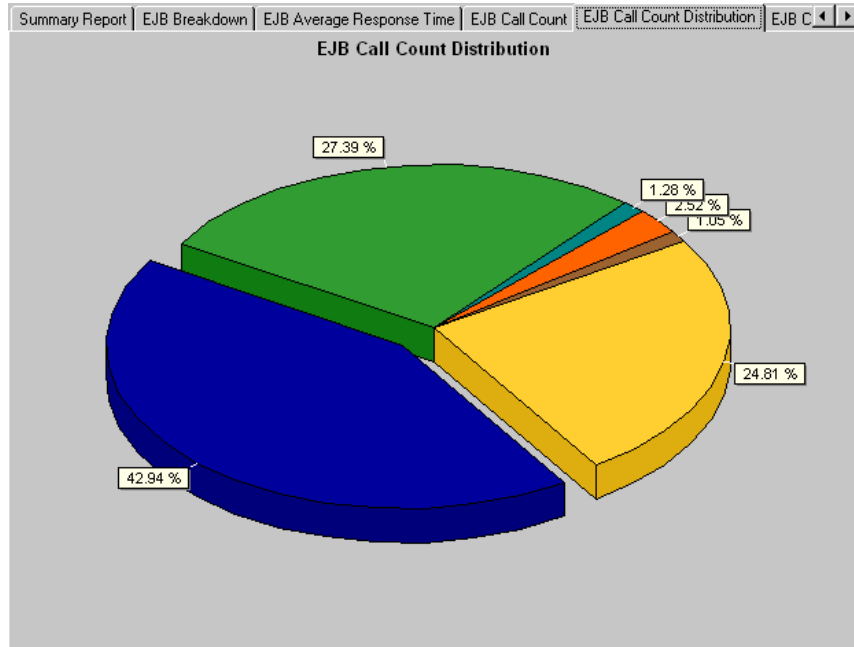
The number of calls made to the class or method is listed in the **Call Count** column of the EJB Breakdown table.

Each class or method is represented by a different colored area on the pie graph. The legend frame (which is found below the graph) identifies the classes by color:

This legend shows that the green colored line belongs to the EJB class *examples.ejb.basic.beanManaged.AccountBean.* Looking at the graph above, we see that 27.39% of calls are made to this class. The actual figures can be seen in the **Call Count** column of the EJB Breakdown table: there are 1327 calls to this class out of a total of 4844 calls.

Hint: To highlight a specific class line in the graph, select the class row in the legend.

Changing Result Granularity

See "Changing the Granularity of the Data," on page 55.

Viewing EJB Methods

To view the call count of individual methods within an EJB class, you can either use drill-down or filtering techniques. For more information, see "Drilling Down in a Graph," on page 52 and "Filtering and Sorting Graph Data," on page 43.

EJB Call Count Per Second Graph

The EJB Call Count Per Second graph shows the number of times per second an EJB Class or method is invoked.
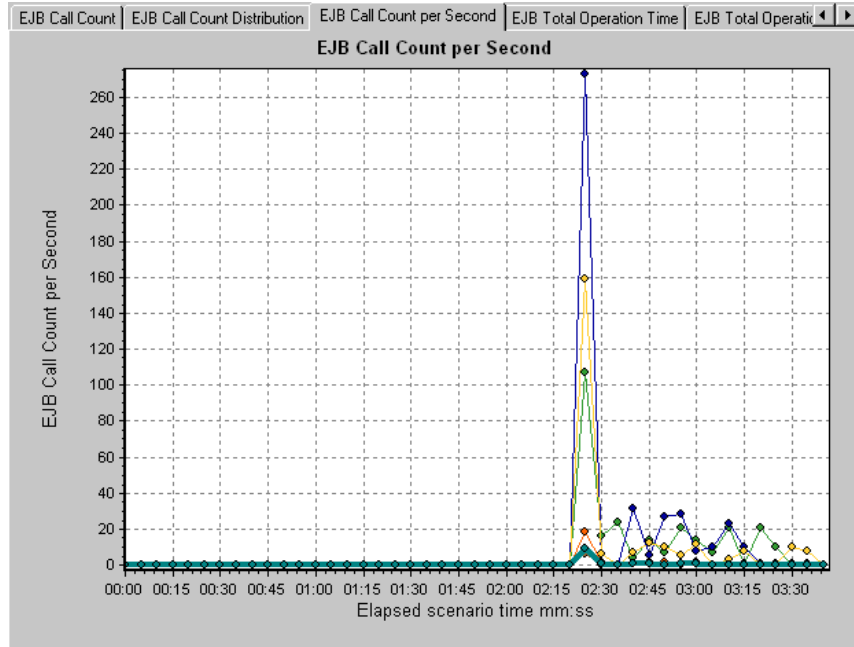
This graph is similar to the EJB Call Count Graph except that the y-axis indicates how many invocations were made to an EJB class or method per second.

Each class or method is represented by a different colored line on the graph. The legend frame (which is found below the graph) identifies the classes by color:

This legend shows that the green colored line belongs to the EJB class *examples.ejb.basic.beanManaged.AccountBean.* Looking at the graph above, we see that calls to this class begins 2:20 minutes into the scenario run. There are 107 calls per second at the 2:25 minute mark.

Hint: To highlight a specific class line in the graph, select the class row in the legend.

Changing Result Granularity

See "Changing the Granularity of the Data," on page 55.

Viewing EJB Methods

To view the call count of individual methods within an EJB class, you can either use drill-down or filtering techniques. For more information, see "Drilling Down in a Graph," on page 52 and "Filtering and Sorting Graph Data," on page 43.

EJB Total Operation Time Graph

The EJB Total Operation Time graph displays the amount of time each EJB class or method takes to execute during the test. Use it to identify those classes or methods which take up an excessive amount of time.
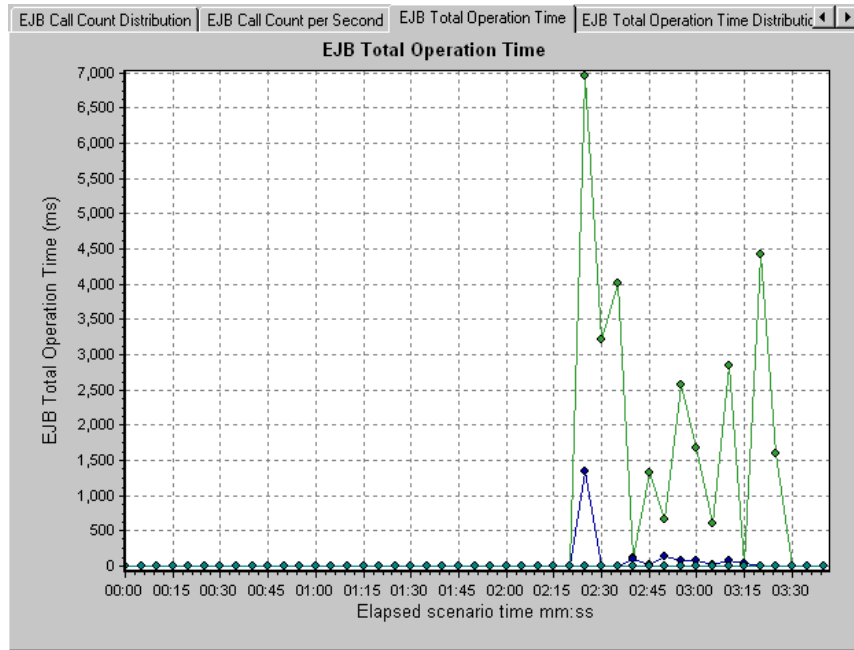
The graph's x-axis indicates the elapsed time from the beginning of the scenario run. The y-axis indicates the total time an EJB class or method is in operation.

Each class or method is represented by a different colored line on the graph. The legend frame (which is found below the graph) identifies the classes by color:

303

This legend shows that the green colored line belongs to the EJB class *examples.ejb.basic.beanManaged.AccountBean.* Looking at the graph above, we see that throughout the scenario, this class consumes more time than any other, especially at 2 minutes and 25 seconds into the scenario run, where all calls to this class take nearly 7 seconds.

Hint: To highlight a specific class line in the graph, select the class row in the legend.

Changing Result Granularity

See "Changing the Granularity of the Data," on page 55.

Viewing EJB Methods

To view the call count of individual methods within an EJB class, you can either use drill-down or filtering techniques. For more information, see "Drilling Down in a Graph," on page 52 and "Filtering and Sorting Graph Data," on page 43.

EJB Total Operation Time Distribution Graph

The EJB Total Operation Time Distribution graph shows the percentage of time a specific EJB class takes to execute compared to all EJB classes. It can also show the percentage of time an EJB method takes to execute compared to all EJB methods within the class. Use it to identify those classes or methods which take up an excessive amount of time.

Each class or method is represented by a different colored area on the pie graph. The legend frame (which is found below the graph) identifies the classes by color:

Color	Scale	Measurement	Minimum	Average	Maximum
☑	1	examples.ejb.basic.beanManaged.AccountBean	0	22.632	311.667
☑	1	examples.ejb.basic.containerManaged.AccountBean	0	0.903	20.667
☑	1	examples.ejb.extensions.readMostly.StockBean	0	0.007	0.083
☑	1	examples.ejb.extensions.readMostly.StockWriterBean	0	0.019	0.031

This legend shows that the green colored line belongs to the EJB class *examples.ejb.basic.beanManaged.AccountBean.* Looking at the graph above, we see that this class takes up 94.08% of the EJB operational time.

Hint: To highlight a specific class line in the graph, select the class row in the legend.

Changing Result Granularity

See "Changing the Granularity of the Data," on page 55.

Viewing EJB Methods

To view the call count of individual methods within an EJB class, you can either use drill-down or filtering techniques. For more information, see "Drilling Down in a Graph," on page 52 and "Filtering and Sorting Graph Data," on page 43.

J2EE Graph

The J2EE graph shows the resource usage of Java™ 2 Platform, Enterprise Edition (J2EE) objects as a function of the elapsed scenario time.

The x-axis represents the time that has elapsed since the start of the scenario run. The y-axis represents the resource usage.
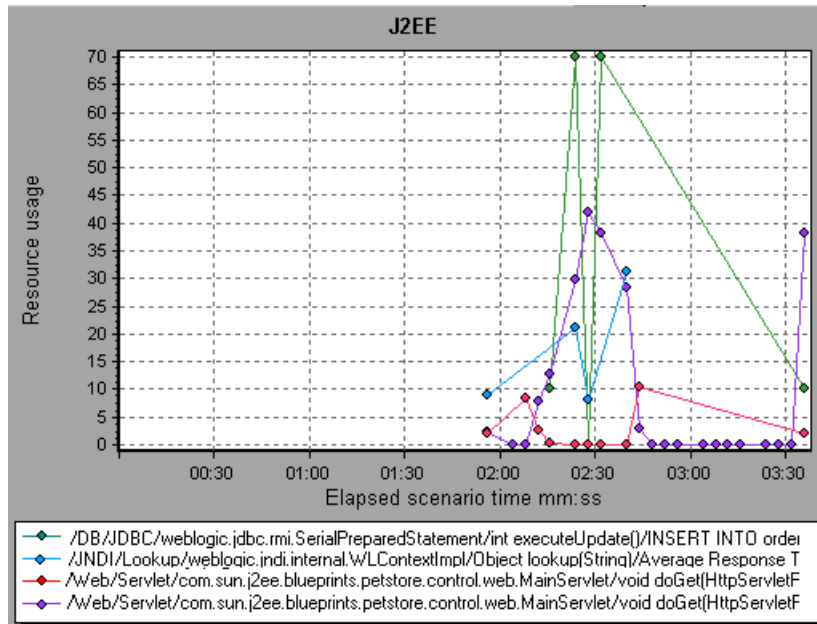
Note: To obtain data for this graph, you must first install and activate the J2EE monitor on the application server machine. You then configure the J2EE monitor on the client machine by selecting the counters you want the monitor to measure.

The following J2EE counters are available for monitoring Servlets/JSP's, JNDI, EJB, and JDBC method calls:

Measurement	Description
Average Response Time	The average response time, in milliseconds, of the J2EE object being monitored.
Method Calls per Second	The number of J2EE object method calls per second.

18

Application Deployment Solutions Graphs

After a scenario run, you use the Application Deployment Solutions graphs to analyze Citrix MetaFrame XP or 1.8 server performance.

This chapter describes:

➤ Citrix MetaFrame XP Graph

About Application Deployment Solutions Graphs

LoadRunner's Citrix MetaFrame XP monitor provides you with information about the application deployment usage of the Citrix MetaFrame XP and 1.8 servers during a scenario execution. In order to obtain performance data, you need to activate the online monitor for the server and specify which resources you want to measure before executing the scenario.

For more information on activating and configuring the Application Deployment Solutions monitor, see the *LoadRunner Controller User's Guide*.

Citrix MetaFrame XP Graph

Citrix MetaFrame is an Application Deployment solution which delivers applications across networks. The Citrix MetaFrame Resource Monitor is an Application Deployment Solution monitor, which provides performance information for the Citrix MetaFrame and 1.8 servers.

Note: To obtain data for this graph, you need to enable the Citrix MetaFrame XP monitor (from the Controller) and select the default measurements you want to display, before running the scenario.

The Citrix MetaFrame XP graph displays statistics about resource usage on the Citrix server during the scenario run.
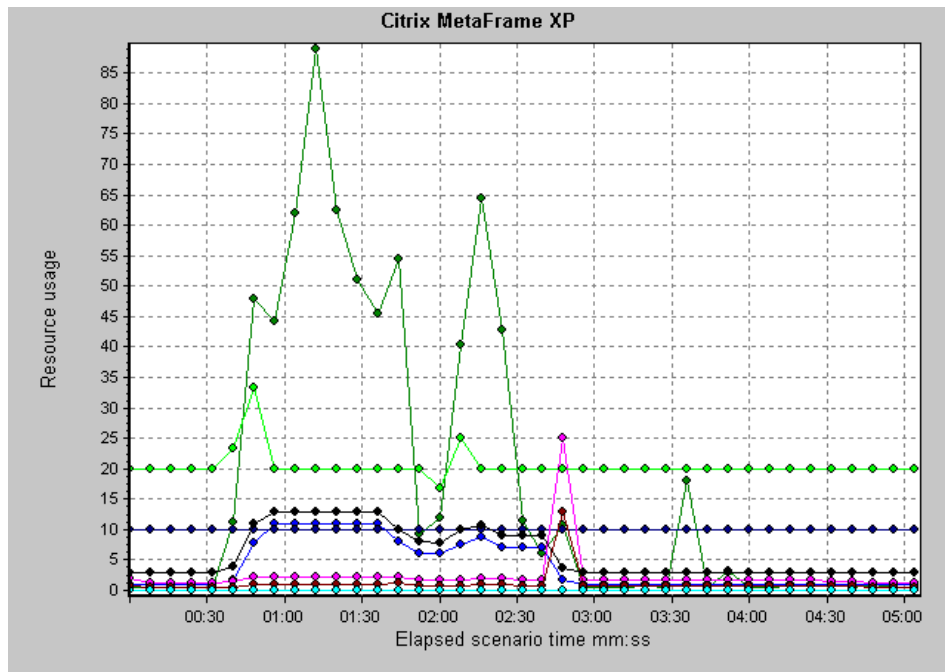
The following Citrix measurements are available:

Non-Virtual Counters

Measurement	Description
% Disk Time	The percentage of elapsed time that the selected disk drive is busy servicing read or write requests.
% Processor Time	The percentage of time that the processor is executing a non-Idle thread. This counter is a primary indicator of processor activity. It is calculated by measuring the time that the processor spends executing the thread of the Idle process in each sample interval, and subtracting that value from 100%. (Each processor has an Idle thread which consumes cycles when no other threads are ready to run). It can be viewed as the percentage of the sample interval spent doing useful work. This counter displays the average percentage of busy time observed during the sample interval. It is calculated by monitoring the time the service was inactive, and then subtracting that value from 100%.
File data Operations/sec	The rate that the computer is issuing Read and Write operations to file system devices. It does not include File Control Operations.

Measurement	Description
Interrupts/sec	The average number of hardware interrupts the processor is receiving and servicing in each second. It does not include DPCs, which are counted separately. This value is an indirect indicator of the activity of devices that generate interrupts, such as the system clock, the mouse, disk drivers, data communication lines, network interface cards and other peripheral devices. These devices normally interrupt the processor when they have completed a task or require attention. Normal thread execution is suspended during interrupts. Most system clocks interrupt the processor every 10 milliseconds, creating a background of interrupt activity. This counter displays the difference between the values observed in the last two samples, divided by the duration of the sample interval.
Output Session Line Speed	This value represents the line speed from server to client for a session in bps.
Input Session Line Speed	This value represents the line speed from client to server for a session in bps.
Page Faults/sec	A count of the Page Faults in the processor. A page fault occurs when a process refers to a virtual memory page that is not in its Working Set in main memory. A Page Fault will not cause the page to be fetched from disk if that page is on the standby list, and hence already in main memory, or if it is in use by another process with whom the page is shared.

Measurement	Description
Pages/sec	The number of pages read from the disk or written to the disk to resolve memory references to pages that were not in memory at the time of the reference. This is the sum of Pages Input/sec and Pages Output/sec. This counter includes paging traffic on behalf of the system Cache to access file data for applications. This value also includes the pages to/from non-cached mapped memory files. This is the primary counter to observe if you are concerned about excessive memory pressure (that is, thrashing), and the excessive paging that may result.
Pool Nonpaged Bytes	The number of bytes in the Nonpaged Pool, a system memory area where space is acquired by operating system components as they accomplish their appointed tasks. Nonpaged Pool pages cannot be paged out to the paging file, but instead remain in main memory as long as they are allocated.
Private Bytes	The current number of bytes this process has allocated that cannot be shared with other processes.
Processor Queue Length	The instantaneous length of the processor queue in units of threads. This counter is always 0 unless you are also monitoring a thread counter. All processors use a single queue in which threads wait for processor cycles. This length does not include the threads that are currently executing. A sustained processor queue length greater than two generally indicates processor congestion. This is an instantaneous count, not an average over the time interval.
Threads	The number of threads in the computer at the time of data collection. Notice that this is an instantaneous count, not an average over the time interval. A thread is the basic executable entity that can execute instructions in a processor.

Measurement	Description
Latency – Session Average	This value represents the average client latency over the life of a session.
Latency – Last Recorded	This value represents the last recorded latency measurement for this session.
Latency – Session Deviation	This value represents the difference between the minimum and maximum measured values for a session.
Input Session Bandwidth	This value represents the bandwidth from client to server traffic for a session in bps.
Input Session Compression	This value represents the compression ratio for client to server traffic for a session.
Output Session Bandwidth	This value represents the bandwidth from server to client traffic for a session in bps.
Output Session Compression	This value represents the compression ratio for server to client traffic for a session.
Output Session Linespeed	This value represents the line speed from server to client for a session in bps.

Virtual Channel Counters

Measurement	Description
Input Audio Bandwidth	This value represents the bandwidth from client to server traffic on the audio mapping channel. This is measured in bps.
Input Clipboard Bandwidth	This value represents the bandwidth from client to server traffic on the clipboard mapping channel. This is measured in bps.
Input COM1 Bandwidth	This value represents the bandwidth from client to server traffic on the COM1 channel. This is measured in bps.
Input COM2 Bandwidth	This value represents the bandwidth from client to server traffic on the COM2 channel. This is measured in bps.
Input COM Bandwidth	This value represents the bandwidth from client to server traffic on the COM channel. This is measured in bps.
Input Control Channel Bandwidth	This value represents the bandwidth from client to server traffic on the ICA control channel. This is measured in bps.
Input Drive Bandwidth	This value represents the bandwidth from client to server traffic on the client drive mapping channel. This is measured in bps.
Input Font Data Bandwidth	This value represents the bandwidth from client to server traffic on the local text echo font and keyboard layout channel. This is measured in bps.
Input Licensing Bandwidth	This value represents the bandwidth from server to client traffic on the licensing channel. This is measured in bps.
Input LPT1 Bandwidth	This value represents the bandwidth from client to server traffic on the LPT1 channel. This is measured in bps.

Measurement	Description
Input LPT2 Bandwidth	This value represents the bandwidth from client to server traffic on the LPT2 channel. This is measured in bps.
Input Management Bandwidth	This value represents the bandwidth from client to server traffic on the client management channel. This is measured in bps.
Input PN Bandwidth	This value represents the bandwidth from client to server traffic on the Program Neighborhood channel. This is measured in bps.
Input Printer Bandwidth	This value represents the bandwidth from client to server traffic on the printer spooler channel. This is measured in bps.
Input Seamless Bandwidth	This value represents the bandwidth from client to server traffic on the Seamless channel. This is measured in bps.
Input Text Echo Bandwidth	This value represents the bandwidth from client to server traffic on the local text echo data channel. This is measured in bps.
Input Thinwire Bandwidth	This value represents the bandwidth from client to server traffic on the Thinwire (graphics) channel. This is measured in bps.
Input VideoFrame Bandwidth	This value represents the bandwidth from client to server traffic on the VideoFrame channel. This is measured in bps.
Output Audio Bandwidth	This value represents the bandwidth from server to client traffic on the audio mapping channel. This is measured in bps.
Output Clipboard Bandwidth	This value represents the bandwidth from server to client traffic on he clipboard mapping channel. This is measured in bps.
Output COM1 Bandwidth	This value represents the bandwidth from server to client traffic on the COM1 channel. This is measured in bps.

Measurement	Description
Output COM2 Bandwidth	This value represents the bandwidth from server to client traffic on the COM2 channel. This is measured in bps.
Output COM Bandwidth	This value represents the bandwidth from server to client traffic on the COM channel. This is measured in bps.
Output Control Channel Bandwidth	This value represents the bandwidth from server to client traffic on the ICA control channel. This is measured in bps.
Output Drive Bandwidth	This value represents the bandwidth from server to client traffic on the client drive channel. This is measured in bps.
Output Font Data Bandwidth	This value represents the bandwidth from server to client traffic on the local text echo font and keyboard layout channel. This is measured in bps.
Output Licensing Bandwidth	This value represents the bandwidth from server to client traffic on the licensing channel. This is measured in bps.
Output LPT1 Bandwidth	This value represents the bandwidth from server to client traffic on the LPT1 channel. This is measured in bps.
Output LPT2 Bandwidth	This value represents the bandwidth from server to client traffic on the LPT2 channel. This is measured in bps.
Output Management Bandwidth	This value represents the bandwidth from server to client traffic on the client management channel. This is measured in bps.
Output PN Bandwidth	This value represents the bandwidth from server to client traffic on the Program Neighborhood channel. This is measured in bps.
Output Printer Bandwidth	This value represents the bandwidth from server to client traffic on the printer spooler channel. This is measured in bps.

Measurement	Description
Output Seamless Bandwidth	This value represents the bandwidth from server to client traffic on the Seamless channel. This is measured in bps.
Output Text Echo Bandwidth	This value represents the bandwidth from server to client traffic on the local text echo data channel. This is measured in bps.
Output Thinwire Bandwidth	This value represents the bandwidth from server to client traffic on the Thinwire (graphics) channel. This is measured in bps.
Output VideoFrame Bandwidth	This value represents the bandwidth from server to client traffic on the VideoFrame channel. This is measured in bps.

19

Middleware Performance Graphs

After a scenario run, you can use the Middleware Performance monitor graphs to analyze the performance of the Tuxedo and the IBM WebSphere MQ Servers.

This chapter describes the:

➤ Tuxedo Resources Graph

➤ IBM WebSphere MQ Graph

About Middleware Performance Graphs

A primary factor in a transaction's response time is the middleware performance usage. LoadRunner's Middleware Performance monitors provide you with information about the middleware performance usage of the Tuxedo and IBM WebSphere MQ servers during a scenario execution. In order to obtain performance data, you need to activate the online monitor for the server and specify which resources you want to measure before executing the scenario.

For more information on activating and configuring the Middleware Performance monitors, see the *LoadRunner Controller User's Guide*.

Tuxedo Resources Graph

The Tuxedo Resources graph provides information about the server, load generator machine, workstation handler, and queue in a Tuxedo system.

The x-axis represents the elapsed time. The y-axis represents the resource usage.
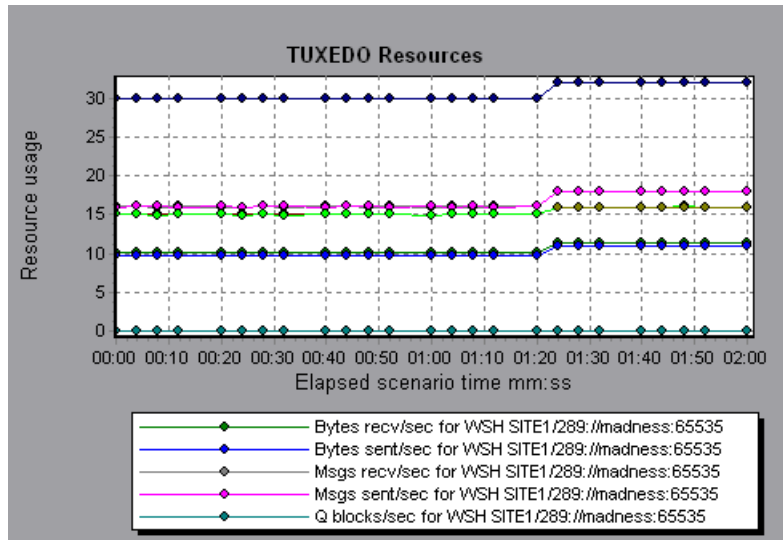
Note: To obtain data for this graph, you need to enable the TUXEDO monitor (from the Controller) and select the default measurements you want to display, before running the scenario.

The following table lists the available Tuxedo monitor measurements:

Monitor	Measurements
Server	**Requests per second** - How many server requests were handled per second
	Workload per second -The workload is a weighted measure of the server requests. Some requests could have a different weight than others. By default, the workload is always 50 times the number of requests.
Machine	**Workload completed per second** - The total workload on all the servers for the machine that was completed, per unit time
	Workload initiated per second - The total workload on all the servers for the machine that was initiated, per unit time
	Current Accessers - Number of clients and servers currently accessing the application either directly on this machine or through a workstation handler on this machine.
	Current Clients - Number of clients, both native and workstation, currently logged in to this machine.
	Current Transactions - Number of in use transaction table entries on this machine.
Queue	**Bytes on queue** - The total number of bytes for all the messages waiting in the queue
	Messages on queue - The total number of requests that are waiting on queue. By default this is 0.

Monitor	Measurements
Workstation Handler (WSH)	**Bytes received per second** - The total number of bytes received by the workstation handler, per unit time
	Bytes sent per second - The total number of bytes sent back to the clients by the workstation handler, per unit time
	Messages received per second - The number of messages received by the workstation handler, per unit time
	Messages sent per second - The number of messages sent back to the clients by the workstation handler, per unit time
	Number of queue blocks per second - The number of times the queue for the workstation handler blocked, per unit time. This gives an idea of how often the workstation handler was overloaded.

IBM WebSphere MQ Graph

The IBM WebSphere MQ graph shows the resource usage of IBM WebSphere MQ Server channel and queue performance counters as a function of the elapsed scenario time.

The x-axis represents the time that has elapsed since the start of the scenario run. The y-axis represents the resource usage.
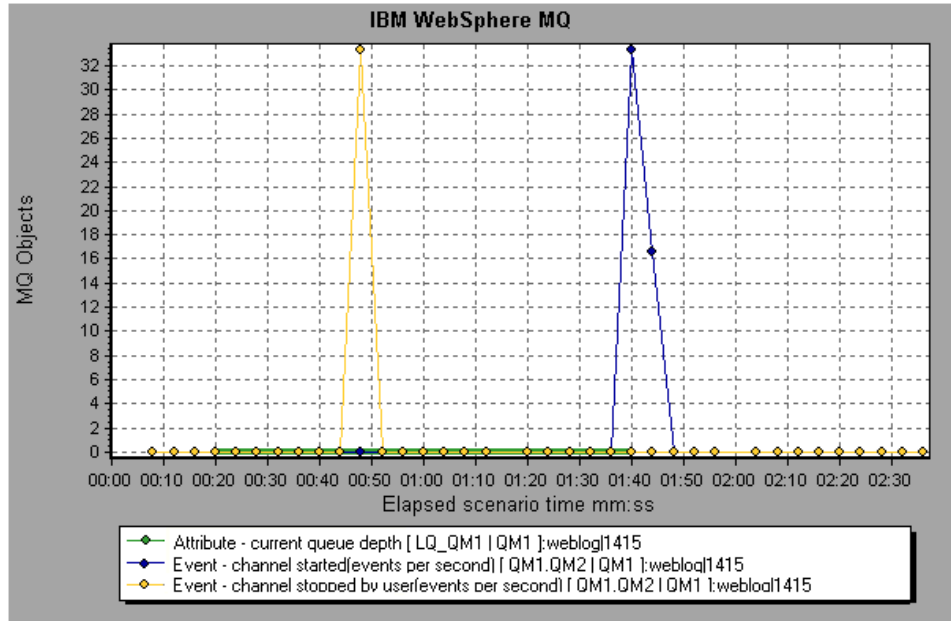
Note: To obtain data for this graph, you need to enable the IBM WebSphere MQ monitor (from the Controller) and select the default measurements you want to display, before running the scenario.

The following IBM WebSphere MQ counters are available:

Queue Performance Counters

Measurement	Description
Event - Queue Depth High (events per second)	An event triggered when the queue depth reaches the configured maximum depth.
Event - Queue Depth Low (events per second)	An event triggered when the queue depth reaches the configured minimum depth.
Event - Queue Full (events per second)	An event triggered when an attempt is made to put a message on a queue that is full.
Event - Queue Service Interval High (events per second)	An event triggered when no messages are put to or retrieved from a queue within the timeout threshold.
Event - Queue Service Interval OK (events per second)	An event triggered when a message has been put to or retrieved from a queue within the timeout threshold.
Status - Current Depth	Current count of messages on a local queue. This measurement applies only to local queues of the monitored queue manager.
Status - Open Input Count	Current count of open input handles. Input handles are opened so that an application may "put" messages to a queue.
Status - Open Output Count	Current count of open output handles. Output handles are opened so that an application may "get" messages from a queue.

Channel Performance Counters

Measurement	Description
Event - Channel Activated (events per second)	Event generated when a channel, waiting to become active but inhibited from doing so due to a shortage of queue manager channel slots, becomes active due to the sudden availability of a channel slot.
Event - Channel Not Activated (events per second)	Event generated when a channel, attempts to become active but inhibited from doing so due to a shortage of queue manager channel slots.
Event - Channel Started (events per second)	Event generated when a channel is started.
Event - Channel Stopped (events per second)	Event generated when a channel is stopped, regardless of source of stoppage.
Event - Channel Stopped by User (events per second)	Event generated when a channel is stopped by a user.
Status - Channel State	The current state of a channel. Channels pass through several states from STOPPED (inactive state) to RUNNING (fully active state). Channel states range from 0 (STOPPED) to 6 (RUNNING).
Status - Messages Transferred	The count of messages that have been sent over the channel. If no traffic is occurring over the channel, this measurement will be zero. If the channel has not been started since the queue manager was started, no measurement will be available.
Status - Buffer Received	The count of buffers that have been received over the channel. If no traffic is occurring over the channel, this measurement will be zero. If the channel has not been started since the queue manager was started, no measurement will be available.

Measurement	Description
Status - Buffer Sent	The count of buffers that have been sent over the channel. If no traffic is occurring over the channel, this measurement will be zero. If the channel has not been started since the queue manager was started, no measurement will be available.
Status - Bytes Received	The count of bytes that have been received over the channel. If no traffic is occurring over the channel, this measurement will appear as zero. If the channel has not been started since the queue manager was started, no measurement will be available.
Status - Bytes Sent	The count of bytes that have been sent over the channel. If no traffic is occurring over the channel, this measurement will appear as zero. If the channel has not been started since the queue manager was started, no measurement will be available.

20

Cross Result and Merged Graphs

The Analysis utility lets you compare results and graphs to determine the source of a problem.

This chapter discusses:

➤ Cross Result Graphs

➤ Generating Cross Result Graphs

➤ Merging Graphs

About Cross Result and Merged Graphs

Comparing results is essential for determining bottlenecks and problems. You use Cross Result graphs to compare the results of multiple scenario runs. You create Merged graphs to compare results from different graphs within the same scenario run.

Cross Result Graphs

Cross Result graphs are useful for:

➤ benchmarking hardware

➤ testing software versions

➤ determining system capacity

If you want to benchmark two hardware configurations, you run the same scenario with both configurations and compare the transaction response times using a single Cross Result graph.

Suppose that your vendor claims that a new software version is optimized to run quicker than a previous version. You can verify this claim by running the same scenario on both versions of the software, and comparing the scenario results.

You can also use Cross Result graphs to determine your system's capacity. You run scenarios using different numbers of Vusers running the same script. By analyzing Cross Result graphs, you can determine the number of users that cause unacceptable response times.

In the following example, two scenario runs are compared by crossing their results, *res12*, and *res15*. The same script was executed twice—first with 100 Vusers and then with 50 Vusers.

In the first run, the average transaction time was approximately 59 seconds. In the second run, the average time was 4.7 seconds. It is apparent that the system works much slower with a greater load.
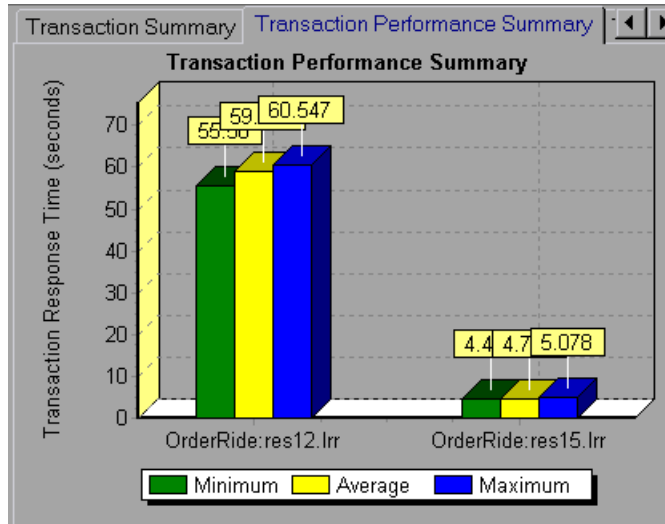
The Cross Result graphs have an additional filter and group by category: *Result Name*. The above graph is filtered to the *OrderRide* transaction for results *res12*, and *res15*, grouped by *Result Name*.

Generating Cross Result Graphs

You can create a Cross Result graph for two or more result sets. The Cross Result dialog box enables you to compare the results of multiple scenario runs.

To generate a Cross Result graph:

1 Choose **File** > **Cross With Result**. The Cross Results dialog box opens.
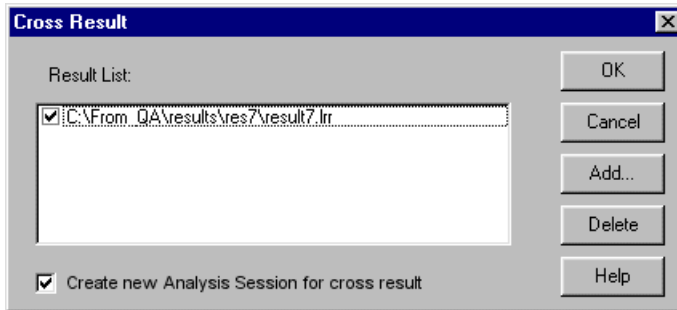
2 Click **Add** to add an additional result set to the **Result List**. The Select Result Files for Cross Results dialog box opens.

3 Locate a results directory and select its result file (*.lrr*). Click **OK**. The scenario is added to the Result List.

4 Repeat steps 2 and 3 until all the results you want to compare are in the Result List.

5 When you generate a Cross Result graph, by default it is saved as a new Analysis session. To save it in an existing session, clear the **Create New Analysis Session for Cross Result** box.

6 Click **OK**. Analysis processes the result data and asks for a confirmation to open the default graphs.

After you generate a Cross Result graph, you can filter it to display specific scenarios and transactions. You can also manipulate the graph by changing the granularity, zoom, and scale. For more information, see Chapter 2, "Working with Analysis Graphs."

Merging Graphs

Analysis lets you merge the results of two graphs from the same scenario into a single graph. The merging allows you to compare several different measurements at once. For example, you can make a merged graph to display the network delay and number of running Vusers, as a function of the elapsed time.

You can merge all graphs with a common x-axis.

To create a merged graph:

 1 Select a graph in the tree view or select its tab to make it active.

 2 Choose **View > Merge Graphs** or click **Merge Graphs**. The Merge Graphs dialog box opens and displays the name of the active graph.
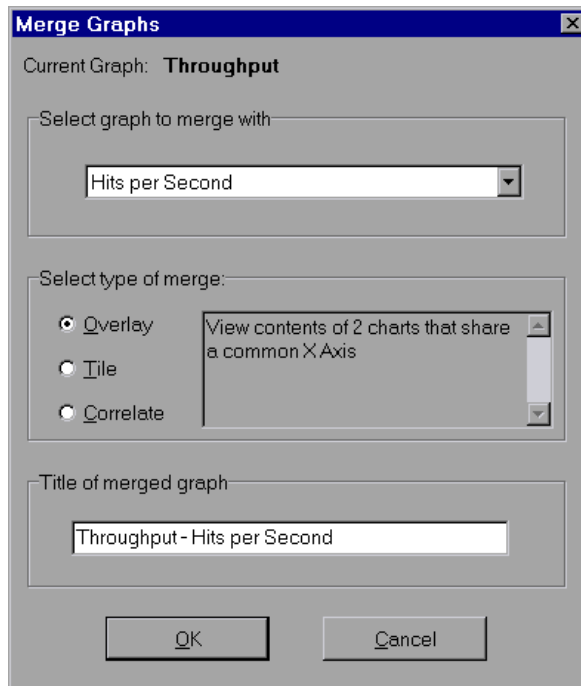
 3 Select a graph with which you want to merge your active graph. Only the graphs with a common x-axis to the active graph are available.

4 Select the merge type and a title for for the merged graph. By default, Analysis combines the titles of the two graphs being merged. For more information, see "Understanding the Merge Graphs Dialog Box," on page 332.

5 Click **OK**.

6 Filter the graph just as you would filter any ordinary graph.

Understanding the Merge Graphs Dialog Box

The Merge Graphs dialog box enables you to merge two graphs into a single graph. In order to merge graphs, the graphs' x-axes must be the same measurement. For example, you can merge Web Throughput and Hits per Second graphs, because their x-axes are Scenario Elapsed Time.

Analysis provides three types of merging:

➤ Overlay

➤ Tile

➤ Correlate

Overlay: Superimpose the contents of two graphs that share a common x-axis. The left y-axis on the merged graph shows the current graph's values. The right y-axis shows the values of the graph that was merged. There is no limit to the number of graphs that you can overlay. When you overlay two graphs, the y-axis for each graph is displayed separately to the right and left of the graph. When you overlay more than two graphs, Analysis displays a single y-axis, scaling the different measurements accordingly.

In the following example, the Throughput and Hits per Second graph are overlaid with one another.
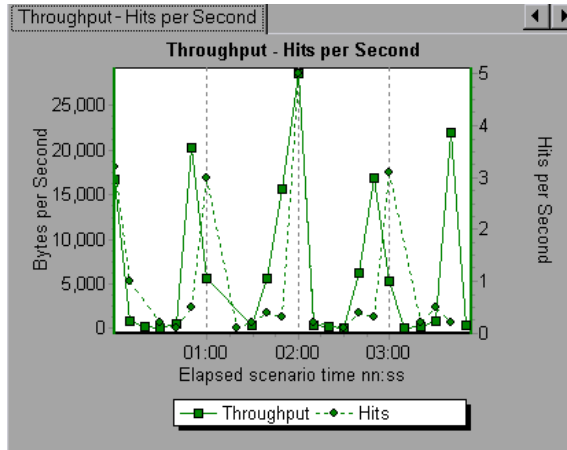
Tile: View contents of two graphs that share a common x-axis in a tiled layout, one above the other. In the following example the Throughput and Hits per Second graph are tiled one above the other.
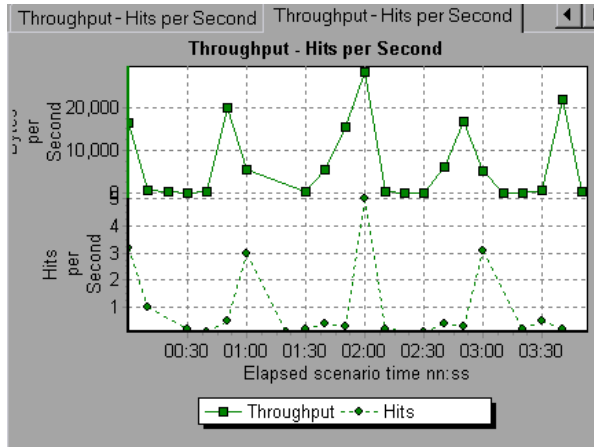
Correlate: Plot the y-axis of two graphs against each other. The active graph's y-axis becomes the x-axis of the merged graph. The y-axis of the graph that was merged, becomes the merged graph's y-axis.

333

In the following example, the Throughput and Hits per Second graph are correlated with one another. The x-axis displays the Bytes per Second (the Throughput measurement) and the y-axis shows the Hits per Second.
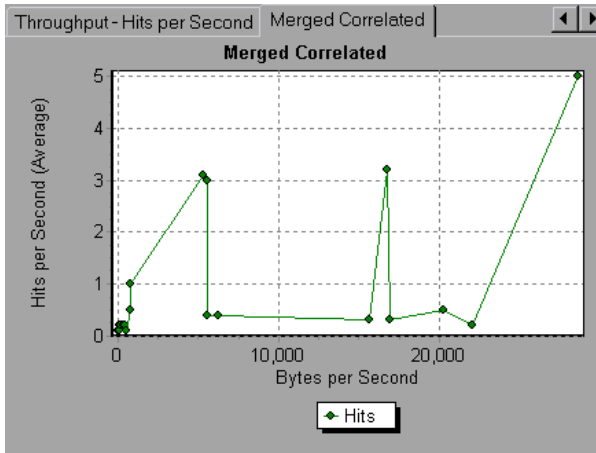
Select Graph to merge with: The drop-down list shows all of the open graphs that share a common x-axis measurement with the current graph. Select one of the graphs in the list.

Select type of merge: Select one of the following options:

➤ **Overlay:** View contents of two graphs that share a common x-axis. The left y-axis on the merged graph shows the current graph's values. The right y-axis shows the values of the graph that was merged with the current graph.

➤ **Tile:** View contents of two graphs that share a common x-axis in a tiled layout, one above the other.

➤ **Correlate:** Plot the y-axes of two graphs against each other. The active graph's y-axis becomes the x-axis of the merged graph. The y-axis of the graph that was merged, becomes the merged graph's y-axis.

Title of Merged Graph: Enter a title for the merged graph. This title will appear in the tree view in the left pane of the Analysis window.

21

Understanding Analysis Reports

After running a scenario, you can use the Analysis reports to analyze the performance of your application.

This chapter describes:

➤ Viewing Summary Reports

➤ Creating HTML Reports

➤ Working with Transaction Reports

➤ Data Point Report

➤ Failed Transaction Report

➤ Failed Vuser Report

➤ Data Point Report

➤ Detailed Transaction Report

➤ Transaction Performance by Vuser Report

You can also create a report in Microsoft Word format. See "Creating a Microsoft Word Report," on page 359.

About Analysis Reports

After running a scenario, you can view reports that summarize your system's performance. Analysis provides the following reporting tools:

➤ Summary report

➤ HTML reports

➤ Transaction reports

The Summary report provides general information about the scenario run. You can view the Summary report at any time from the Analysis window.

You can instruct Analysis to create an HTML report. Analysis creates an HTML report for each one of the open graphs.

Transaction reports provide performance information about the transactions defined within the Vuser scripts. These reports give you a statistical breakdown of your results and allow you to print and export the data.

Viewing Summary Reports

The Summary report provides general information about scenario execution. This report is always available from the tree view or as a tab in the Analysis window.

The Summary report lists statistics about the scenario run and provides links to the following graphs: Running Vusers, Throughput, Hits Per Second, HTTP Responses per Second, Transaction Summary, and Average Transaction Response Time.

At the bottom of the page, the Summary report displays a table containing the scenario's transaction data. Included in this data is a 90 Percent column, indicating the maximum response time for ninety percent of the transactions.

| Summary Report | Running Vusers | Hits per Second | Throughput | Transaction Summary | Average Transaction Response Time |

Analysis Summary   Period: 29/07/2002 14:32:24 - 29/07/20⁻

Scenario Name: Scenario1
Results in session: C:\Documents and Settings\richardl\Local Settings\Temp\res\res.lrr.
Duration: 43 seconds.

Statistics Summary

⊙ **Maximum Running Vusers:**   1
⊙ **Total Throughput (bytes):**   158,410
⊙ **Average Throughput (bytes/second):**   3,684
⊙ **Total Hits:**   34
⊙ **Average Hits per Second:**   0.791   **View HTTP Responses Summary**

Transaction Summary

⊙ **Transactions:**   Total   Total   Total   **Average Response Time**
                         passed: 4   failed: 0   Stopped: 0

Transaction Name	Minimum	Average	Maximum	Std	90 Percent	Pass	Fail	
Action1_Transaction	33.75	33.75	33.75	0	33.748	1	0	0
login	5.531	5.531	5.531	0	5.527	1	0	0

You can save the Summary report to an Excel file by selecting **View** > **Export Summary to Excel**.

Creating HTML Reports

Analysis lets you create HTML reports for your scenario run. It creates a separate report for each one of the open graphs and a Summary report. The Summary report is identical to the Summary report that you access from the Analysis window. The report also provides a link to an Excel file containing the graph data.

To create HTML reports:

1 Open all graphs that you want to include in the report.

2 Choose **Reports** > **HTML Report** or click **Create HTML Report**. The Select Report Filename and Path dialog box opens.

3 Specify a path and file name for the HTML report and click **OK**. Analysis saves a Summary report called by the file name in the selected folder, and the rest of the graphs in a folder with the same name as the file name. When you create an HTML report, Analysis opens your default browser and displays the Summary report.

4 To view an HTML report for one of the graphs, click its link in the left frame.

5 To copy the HTML reports to another location, be sure to copy the filename and the folder with the same name. For example, if you named your HTML report *test1*, copy *test1.html* and the folder *test1* to the desired location.

Working with Transaction Reports

LoadRunner's Transaction reports are divided into the following categories:

➤ Activity

➤ Performance

Activity reports provide information about the number of Vusers and the number of transactions executed during the scenario run. The available Activity reports are *Scenario Execution*, *Failed Transaction*, and *Failed Vusers*.

Performance reports analyze Vuser performance and transaction times. The available Performance reports are *Data Point*, *Detailed Transaction*, and *Transaction Performance by Vuser*.

In order to view a report, you must generate it from the Analysis window. LoadRunner reports are displayed in a Report Viewer. You can print, save, or export the data using the viewer.

Selecting and Displaying Reports

Analysis provides several built-in reports which contain detailed summaries about the scenario, the transactions and Vusers.

To display a report:

1 Open the desired Analysis session file (.lra extension), or LoadRunner result file (.lrr extension), if it is not already open.

2 From the Reports menu choose a report. The report is generated and displayed. You can display multiple copies of the same report.

The Report Viewer

Each report is displayed in its own report viewer. Each viewer contains a header and a toolbar.

Report Header

The header displays general run time information.
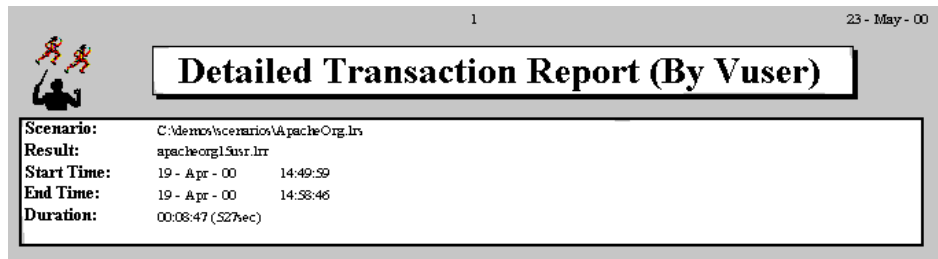
The report header contains the following information:

Title: The name of the report.

Scenario: The name of the scenario described in the report.

Result: The pathname of the scenario results directory.

Start time: The time at which the Run Scenario command was executed.

End time: The time at which the scenario script was terminated.

Duration: The total run time of the scenario.

Report Viewer Toolbar

Each report viewer has a toolbar that lets you perform operations on displayed reports.

Print *Zoom*
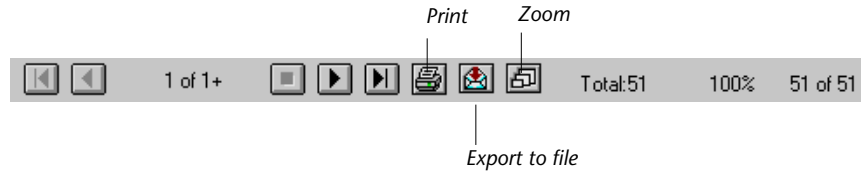
Export to file

The report viewer toolbar contains the following buttons:

Zoom : Toggles between an actual size, full page, and magnified views of the report.

Print : Prints the displayed report.

Export to file : Exports the displayed information to a text file.

If there are multiple values for the y-axis, as in the Transaction Performance by Vuser graph (min, average, and max), all of the plotted values are displayed.

Scenario Execution Report

The Scenario Execution report is an Activity report that provides details about major events that occurred during the scenario run. This includes information on every Vuser, such as when it was ready to run and for how long it ran.

Group: **g1**

Vuser	Host	Ready At	Running At	Duration	Termination Status
Vuser1	10.1.1.30	14:52:58	14:53:30	00:05:43 (343sec)	Abort
Vuser2	10.1.1.30	14:52:58	14:53:30	00:05:43 (343sec)	Abort
Vuser3	10.1.1.30	14:52:58	14:53:30	00:05:43 (343sec)	Abort
Vuser4	10.1.1.30	14:52:58	14:53:30	00:05:43 (343sec)	Abort
Vuser5	10.1.1.30	14:52:58	14:53:30	00:05:43 (343sec)	Abort

Summary

Vusers:	5						
Passed:	0	Failed:	0	Error:	0	Stopped:	5

Failed Transaction Report

The Failed Transaction report is an Activity report that provides details about the beginning time, end time, and duration of the failed, but completed transaction.

Group: g1

Vuser :   Vuser1

Transaction	Start time	End time	Duration
Apache_Server	14:58:36.117	14:58:37.319	00:00:01.202
Surf_Apache	14:58:33.043	14:58:37.319	00:00:04.276

Vuser :   Vuser2

Transaction	Start time	End time	Duration
Apache_Server	14:58:35.265	14:58:37.328	00:00:02.063
Surf_Apache	14:58:24.810	14:58:37.328	00:00:12.518

Failed Vuser Report

The Failed Vuser report is an Activity report that provides details about all Vusers that were in the ERROR, STOPPED, or DONE:FAILED states during the scenario execution. The *Ready At* and *Running At* times are relative to the computer's system clock.

Group: **g2**

Vuser	Host	Ready At	Running At	Duration	Termination Status
Vuser1	localhost	14:53:26	14:54:42	00:05:14 (314sec)	Abort
Vuser2	localhost	14:53:26	14:54:42	00:05:14 (314sec)	Abort
Vuser3	localhost	14:53:26	14:54:42	00:05:14 (314sec)	Abort
Vuser4	localhost	14:53:26	14:54:42	00:05:14 (314sec)	Abort
Vuser5	localhost	14:53:26	14:54:42	00:05:14 (314sec)	Abort

Summary

Vusers: 5

Failed:	0	*Error:*	0	*Stopped:*	5

In this scenario, all five Vusers were stopped.

Data Point Report

LoadRunner enables you to record your own data for analysis. You instruct LoadRunner to record the value of an external function or variable, also known as a *data point*, during the scenario run. Using the gathered data, LoadRunner creates a graph and report for the data point.

The data point is set by including an **lr_user_data_point** function (**user_data_point** for GUI Vusers) in your Vuser script. For more information, refer to the online *LoadRunner Online Function Reference*.

The **Data Point** graph shows the value of the data point during the scenario run. The *x*-axis represents the number of seconds that elapsed since the start time of the run. The *y-axis* displays the value of each recorded data point statement.

The **Data Point** report is a Performance report that lists the name of the data point, its value, and the time its value was recorded. The values are displayed for each Group and Vuser.

Group: Group1

		Data Point	Time
Vuser id: 1			
	memory	19.00	13:37:16
	memory	1.00	13:37:20
	memory	9.00	13:37:32
	memory	1.00	13:37:36
	memory	1.00	13:37:40
Vuser id: 2			
	memory	6.00	13:37:05
	memory	8.00	13:37:20
	memory	9.00	13:37:32
	memory	1.00	13:37:36
	memory	1.00	13:37:40

Detailed Transaction Report

The Detailed Transaction (by Vuser) report is a Performance report that provides a list of all transactions executed by each Vuser during a scenario. The report provides details about the execution time of each transaction per Vuser.

Group: zorb

_Vuser_id:_ 1

Transaction	Start time	End time	Duration	Think Time	Wasted Time	Result
end_section	14:07:59.029	14:08:00.045	00:00:01.016	00:00:01.016	00:00:00.000	Pass
General	14:05:27.748	14:06:06.482	00:00:38.618	00:00:21.715	00:00:00.116	Pass
General	14:06:06.482	14:07:02.701	00:00:37.104	00:00:21.716	00:00:19.115	Pass
General	14:07:02.701	14:07:59.029	00:00:37.212	00:00:21.715	00:00:19.116	Pass
mc_run	14:05:37.310	14:05:41.764	00:00:04.453	00:00:02.609	00:00:00.001	Pass
mc_run	14:06:16.045	14:06:20.248	00:00:04.202	00:00:02.609	00:00:00.001	Pass
mc_run	14:07:12.264	14:07:15.920	00:00:03.655	00:00:02.609	00:00:00.001	Pass

The following values are reported:

Start time: the system time at the beginning of the transaction

End time: the actual system time at the end of the transaction, including the think time and wasted time.

Duration: the duration of the transaction in the following format: hrs:minutes:seconds:milliseconds. This value includes think time, but does not include wasted time.

Think time: the Vuser's think time delay during the transaction.

Wasted time: the LoadRunner internal processing time not attributed to the transaction time or think time. (primarily RTE Vusers)

Results: the final transaction status, either Pass or Fail.

Transaction Performance by Vuser Report

The Transaction Performance Summary by Vuser report is a Performance report that displays the time required by each Vuser to perform transactions during the scenario. The report indicates if the transaction was successful and what the minimum, maximum, and average times were for each Vuser. This report is useful when you have several different types of Vusers in a scenario and you want to characterize performance for each type.

Transaction: Apache_Home

Group: **g1**

				P e r f o r m a n c e (sec)			
Vuser	Pass	Fail	Min	Avg	Max	STD
Vuser1	40	0	0.22	1.06	3.80	0.99
Vuser2	37	0	0.22	1.37	7.40	1.60
Vuser3	40	1	0.22	1.08	5.67	1.17
Vuser4	39	0	0.23	1.04	3.91	0.97
Vuser5	39	1	0.22	0.95	2.92	0.75
Total: 5	195	2	0.22	1.10	7.40	

22

Managing Results Using TestDirector

LoadRunner's integration with TestDirector lets you manage Analysis result sessions using TestDirector, Mercury Interactive's test management tool.

This chapter describes:

➤ Connecting to and Disconnecting from TestDirector

➤ Opening an Existing Session Using TestDirector

➤ Creating a New Session Using TestDirector

➤ Saving Sessions to a TestDirector Project

About Managing Results Using TestDirector

LoadRunner works together with TestDirector to provide an efficient method for storing and retrieving scenarios and collecting results. You store scenarios and results in a TestDirector project and organize them into unique groups.

In order for LoadRunner to access a TestDirector project, you must connect it to the Web server on which TestDirector is installed. You can connect to either a local or remote Web server.

For more information on working with TestDirector, refer to the *TestDirector User's Guide*.

Connecting to and Disconnecting from TestDirector

If you are working with both LoadRunner and TestDirector, LoadRunner can communicate with your TestDirector project. You can connect or disconnect LoadRunner from a TestDirector project at any time during an Analysis session.

Connecting LoadRunner to TestDirector

The connection process has two stages. First, you connect LoadRunner to a local or remote TestDirector Web server. This server handles the connections between LoadRunner and the TestDirector project.

Next, you choose the project you want LoadRunner to access. The project stores scenarios and results for the application you are testing. Note that TestDirector projects are password protected, so you must provide a user name and a password.

To connect LoadRunner to TestDirector:

1 In Analysis, choose **Tools** > **TestDirector Connection**. The TestDirector Connection dialog box opens.
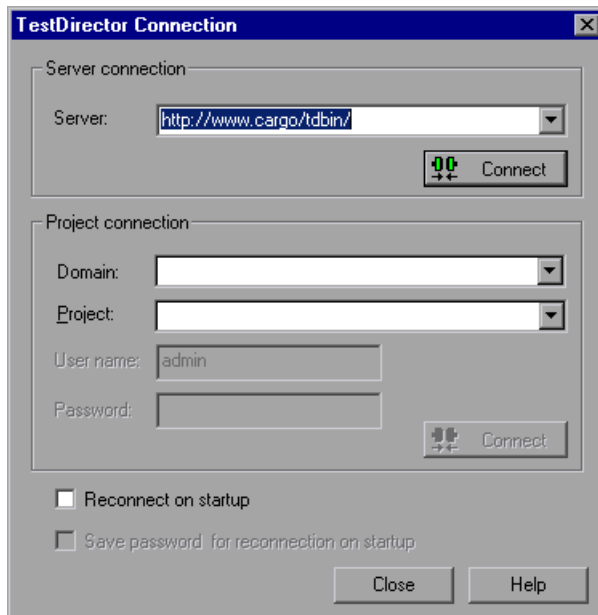
2 In the **Server** box, type the URL address of the Web server on which TestDirector is installed.

Note: You can choose a Web server accessible via a Local Area Network (LAN) or a Wide Area Network (WAN).

---

3 Click **Connect**. Once the connection to the server is established, the server's name is displayed in the Server box.

4 Enter the project connection information, as described in "Understanding the TestDirector Connection Dialog Box," on page 353.

5 Click **Connect** to connect LoadRunner to the selected project.

Once the connection to the selected project is established, the project's name is displayed in read-only format in the Project box.

6 Select reconnection options, as described in "Understanding the TestDirector Connection Dialog Box," on page 353.

7 Click **Close**.

Disconnecting LoadRunner from TestDirector

You can disconnect LoadRunner from a selected TestDirector project and Web server.

To disconnect LoadRunner from TestDirector:

1 In the Controller, choose **Tools** > **TestDirector Connection**. The TestDirector Connection dialog box opens.
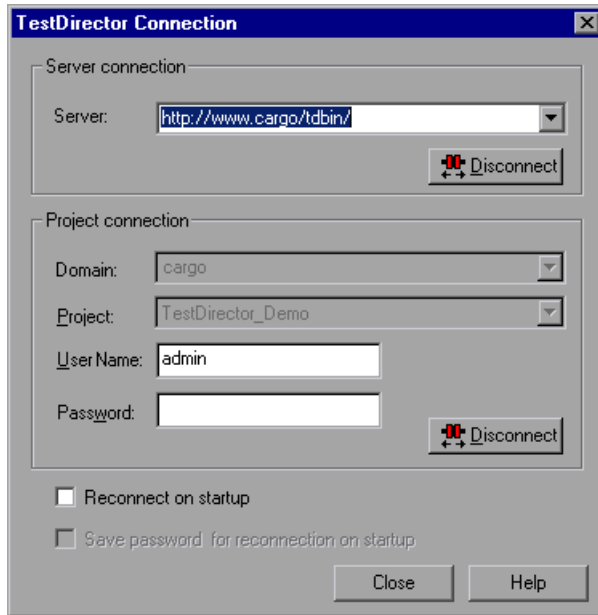
2 To disconnect LoadRunner from the selected project, click **Disconnect** in the **Project Connection** section.

3 To disconnect LoadRunner from the selected server, click **Disconnect** in the **Server Connection** section.

4 Click **Close**.

Understanding the TestDirector Connection Dialog Box

The TestDirector Connection dialog box enables you to open a connection to a TestDirector project. TestDirector uses a project repository to help you to organize and manage your scenario results and Analysis sessions.

Server Connection: Before you can work with a TestDirector project, you must open a connection to the server that hosts the project.

   Server: Enter the name of the server that hosts the TestDirector project.

   Connect: Click this button to connect to the specified server.

Project Connection: After connecting, you select a project, and enter the user name and password for the project. The project stores the Analysis session information.

   Domain: Enter the domain name.

   Project: Select the project to which you want to connect. The list includes all projects that are registered with the selected server.

   User Name: Enter your user name.

   Password: Enter your user password.

   Connect: Click this button to connect to the selected project.

Reconnect on startup: Select this option to automatically open the connection to the TestDirector server and the specified project when you start Analysis.

Save password for reconnection on startup: Select this option to save the specified password to the registry to automate the login process.

Opening an Existing Session Using TestDirector

When LoadRunner is connected to a TestDirector project, you can open an existing Analysis session from TestDirector. You locate sessions according to their position in the test plan tree, rather than by their actual location in the file system.

You use the Open Existing Analysis Session File dialog box to open an existing session file with an .lra extension. The session file contains a result set and your configuration settings as well as merged graphs, Cross Result graphs, etc.

To open a session from a TestDirector project:

1 Connect to the TestDirector server. For more information, see "Connecting LoadRunner to TestDirector" on page 350.

2 In the Controller, choose **File** > **Open** or click **File Open**. The Open Existing Analysis Session File from TestDirector Project dialog box opens and displays the test plan tree.
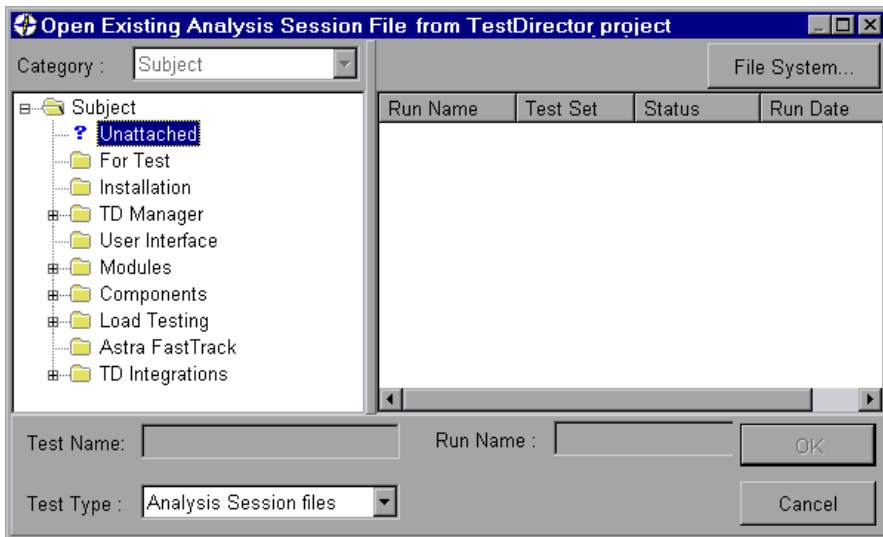
3 To open a scenario directly from the file system, click **File System**. The Open Existing Analysis Session File dialog box opens. (From the Open Existing Analysis Session File dialog box, you can return to the Open Existing Analysis Session File from TestDirector Project dialog box by clicking the **TestDirector** button.)

4 Select the relevant subject in the test plan tree. To expand the tree and view sublevels, double-click closed folders. To collapse the tree, double-click open folders.

Note that when you select a subject, the sessions that belong to the subject are displayed in the **Run Name** list.

5 Select a session from the **Run Name** list. The session is displayed in the **Test Name** box.

6 Click **OK** to open the session. LoadRunner loads the session. The name of the session appears in the Analysis title bar.

Note: You can also open sessions from the recent sessions list in the File menu. If you select a session located in a TestDirector project, but LoadRunneris currently not connected to that project, the TestDirector Connection dialog box opens. Enter your user name and password to log in to the project, and click **OK**.

Creating a New Session Using TestDirector

When LoadRunner is connected to a TestDirector project, you can create a new Analysis session using a result file (.lrr extension) stored in TestDirector. You locate result files according to their position in the test plan tree, rather than by their actual location in the file system.

You use the Open Result File for New Analysis Session dialog box to choose a result file (.lrr extension) for a new Analysis session. A session must contain at least one set of results and when you create a new session, you are prompted for a result set. When you save a session file, it retains all of the settings you applied to the graphs to a file with an .lra extension.

To create a new session using results from a TestDirector project:

1 Connect to the TestDirector server. For more information, see "Connecting LoadRunner to TestDirector" on page 350.

2 In Analysis, choose **File** > **New** or click **Create New Analysis Session**. The Open Result File for New Analysis Session from TestDirector Project dialog box opens and displays the test plan tree.
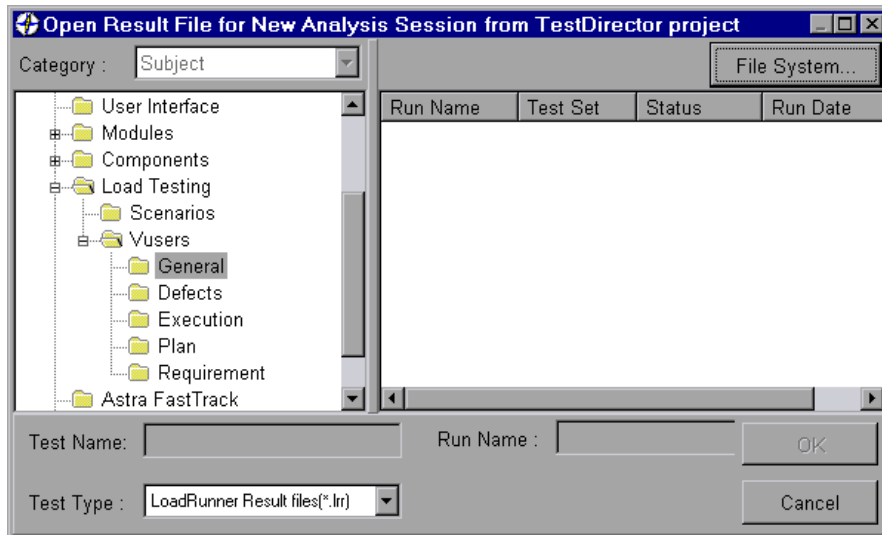
3 To open a result file directly from the file system, click **File System**. The Open Result File for New Analysis Session dialog box opens. (From the Open Result File for New Analysis Session dialog box, you can return to the Open Result File for New Analysis Session from TestDirector Project dialog box by clicking the **TestDirector** button.)

4 Select the relevant subject in the test plan tree. To expand the tree and view sublevels, double-click closed folders. To collapse the tree, double-click open folders.

Note that when you select a subject, the sessions that belong to the subject appear in the **Run Name** list.

5 Select an Analysis session from the **Run Name** list. The scenario is displayed in the Test Name box.

6 Click **OK** to open the session. LoadRunner loads the session. The name of the session appears in the Analysis title bar.

Note: You can also open Analysis sessions from the recent session list in the File menu. If you select a session located in a TestDirector project, but LoadRunneris currently not connected to that project, the TestDirector Connection dialog box opens. Enter your user name and password to log in to the project, and click **OK**.

Saving Sessions to a TestDirector Project

When LoadRunner is connected to a TestDirector project, you can create new sessions in LoadRunner and save them directly to your project. To save a session, you give it a descriptive name and associate it with the relevant subject in the test plan tree. This helps you to keep track of the sessions created for each subject and to quickly view the progress of test planning and creation.

To save a session to a TestDirector project:

1 Connect to the TestDirector server. For more information, see "Connecting LoadRunner to TestDirector" on page 350.

2 Select **File** > **Save** and save the session in the TestDirector data directory.

23

Creating a Microsoft Word Report

You can create a report as a Microsoft Word document. This chapter describes:

➤ Creating Microsoft Word Reports

About the Microsoft Word Report

You can use the Word Report generation tool to automatically summarize and display the test's significant data in graphical and tabular format and to display and describe all graphs in the current Analysis session.

Other features of the report include the automatic inclusion of an overview of the LoadRunner Scenario configuration, and an executive summary.

The report is structured into logical and intuitive sections with a table of contents and various appendices.

Creating Microsoft Word Reports

You create Microsoft Word reports using the Microsoft Word Report dialog box.

To create Microsoft Word reports:

1 Choose **Reports** > **Microsoft Word Reports...** The Microsoft Word Report dialog box opens
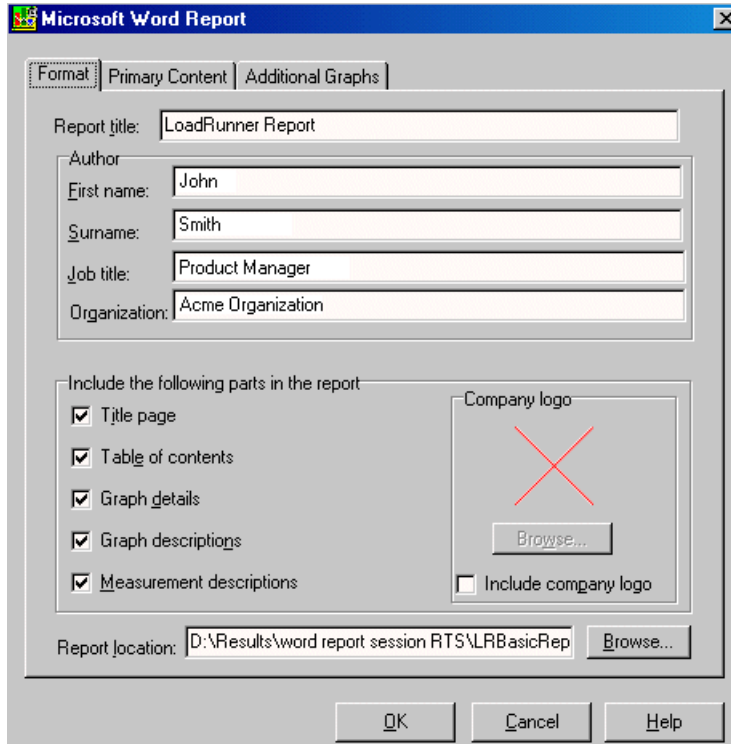
2 Enter the Microsoft Word report options as described in the following sections:

➤ Understanding the Microsoft Word Report Dialog Box—Format Tab

➤ Understanding the Microsoft Word Report Dialog Box—Primary Content Tab

➤ Understanding the Microsoft Word Report Dialog Box—Additional Graphs Tab

3 Click **OK**. The report generation process may take a few minutes.

When completed, Analysis launches the Microsoft Word application containing the report. The file is saved to the location specified in the **Report Location** box of the **Format** tab.

Understanding the Microsoft Word Report Dialog Box—Format Tab

You use the Format tab to add custom title and author information to the Word report, as well as to include additional pages and descriptive comments.

Report title: Enter a name for the report. This will be displayed on the report's title page.

Author: Enter the author details. This will be displayed on the report's title page.

Title Page: Attaches a cover page to the report. For example:

Table of contents: Attaches a table of contents to the report, placed after the cover page.

Graph details: Displays details such as graph filters and granularity settings. These details also appear in the **Description** tab below the graph. For example:

Graph descriptions: Displays a short description of the graph. The description is identical to the one which appears in the **Description** tab in the Analysis window. For example:

Measurement descriptions: Attaches descriptions of each type of monitor measurement in the report appendix.

Report location: Enter the path to the Microsoft Word report.

Include Company logo: Includes a logo (.bmp format) in the report.

Understanding the Microsoft Word Report Dialog Box— Primary Content Tab

You use the Primary Content tab of the Microsoft Word Report dialog box to include graphs and tables of the most significant performance data. You can also include an executive summary and Scenario information that assists the reader in gaining a general view of the test.
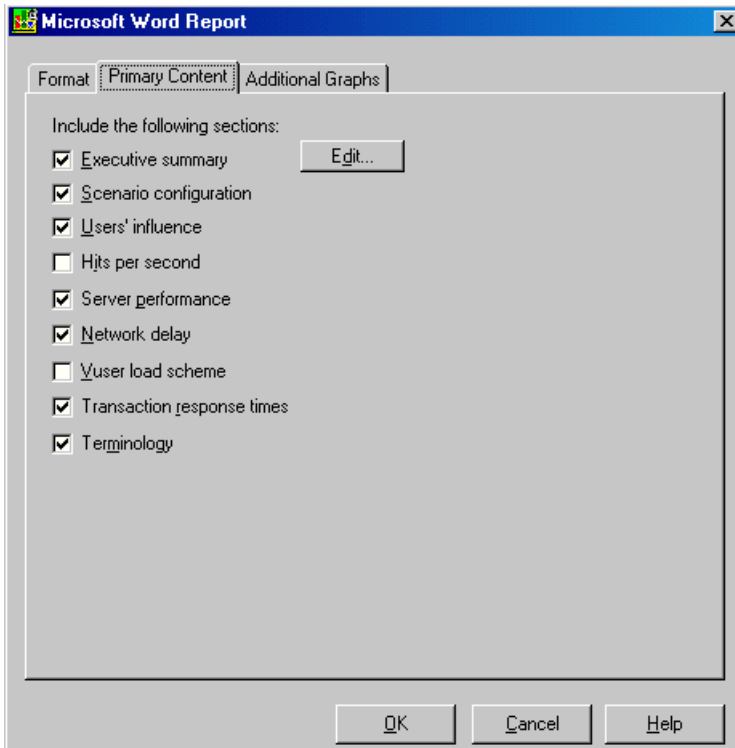
Executive summary: Includes your own high-level summary or abstract of the LoadRunner test, suitable for senior management. An executive summary typically compares performance data with business goals, states significant findings and conclusions in non-technical language, and suggests recommendations. To create the executive summary, click **Edit** and enter the text in the Executive Summary dialog box.

The executive summary also includes two other sub-sections, Scenario Summary and Top Time-Consuming Transactions:

1.2 Scenario Summary

Duration:	43 seconds.
Maximum Running Vusers:	7
Total Throughput (bytes):	65,073
Total Hits:	29

1.3 Top Time-Consuming Transactions

Transaction Name	Ave	Max	90%
vuser_init_Transaction	19.134	23.864	23.854
Actions_Transaction	2.708	7.14	4.084
create	0.499	0.691	0.685

Scenario configuration: Gives the basic schema of the test, including the name of result files, Controller scheduler information, scripts, and Run Time Settings.

Users' influence: A graph that helps you view the general impact of Vuser load on performance time. It is most useful when analyzing a load test which is run with a gradual load.

Hits per second: Applicable to Web tests. Displays the number of hits made on the Web server by Vusers during each second of the load test. It helps you evaluate the amount of load Vusers generate, in terms of the number of hits.

Server performance: A graph that displays a summary of resources utilized on the servers.

Network delay: Displays the delays for the complete network path between machines.

Vuser load scheme: Displays the number of Vusers that executed Vuser scripts, and their status, during each second of a load test. This graph is useful for determining the Vuser load on your server at any given moment.

Transaction response times: Displays the average time taken to perform transactions during each second of the load test. This graph helps you determine whether the performance of the server is within acceptable minimum and maximum transaction performance time ranges defined for your system.

Terminology: Displays an explanation of special terms used in the report.

Understanding the Microsoft Word Report Dialog Box— Additional Graphs Tab

You use the Additional Graphs tab of the Microsoft Word Report dialog box you to include graphs into the Word report. The graphs that were generated in the current Analysis session are listed in the dialog. You can also add other LoadRunner graphs.
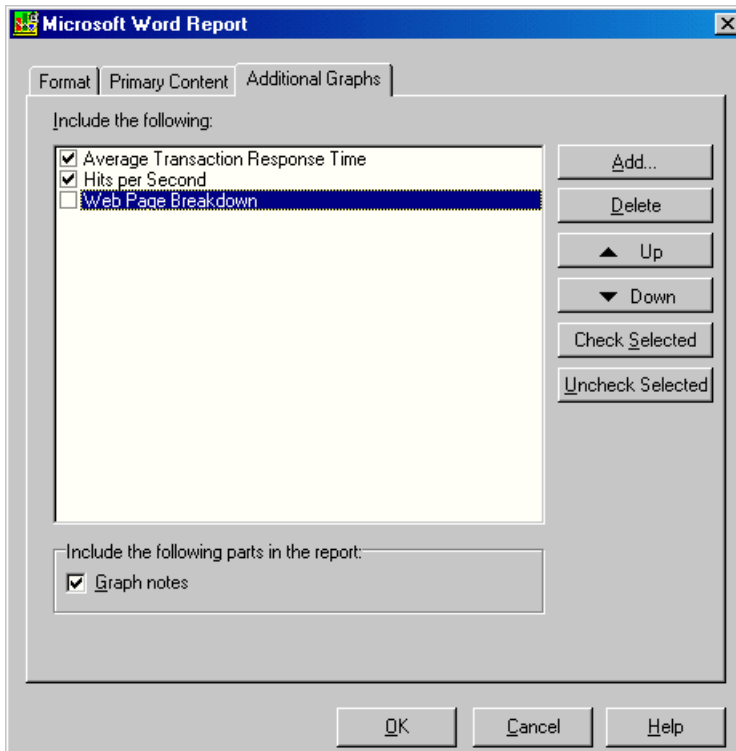
Include the following: Select graphs to include in the Microsoft Word report.

Graph notes: Select this to include text you entered for the graph in the **User Notes** tab of the Analysis window.

Add: Opens the Open a New Graph dialog box to enable you to add other graphs not yet generated in the Analysis session. After selecting the graph, it will be generated and added to the Word Report. For more information, see "Opening Analysis Graphs," on page 26.

Delete: Deletes a selected graph from the list.

Up/Down: Changes the order in which the graph is displayed in the Microsoft Word report.

Check Selected: Selects the current report in the **Include the following** box.

Uncheck Selected: Clears the selection of the current report in the **Include the following** box.

24

Importing External Data

The LoadRunner Analysis Import Data tool enables you to import and integrate non-Mercury Interactive data into a LoadRunner Analysis session. After the import procedure, you can view the data files as graphs within the session, using all the capabilities of the Analysis tool.

Suppose an NT Performance Monitor runs on a server and measures its behavior. Following a LoadRunner scenario on the server, you can retrieve the results of the Performance Monitor, and integrate the data into LoadRunner's results. This enables you to correlate trends and relationships between the two sets of data: LoadRunner's and the Performance Monitor's.

In this case, the results of the NT Performance Monitor are saved as a .csv file. You launch the Import Data tool, direct it to the .csv file, and specify its format. LoadRunner reads the file and integrates the results into its own Analysis session.

For a list of data formats that are supported, see "Supported File Types" on page 374. To define your own custom data files, see "Defining Custom File Formats" on page 377.

This chapter describes:

➤ Using the Import Data Tool

➤ Supported File Types

➤ Defining Custom File Formats

➤ Defining Custom Monitor Types for Import

Using the Import Data Tool

To use the Import Data tool:

1 Choose **Tools** > **External Monitors** > **Import Data**. The Import Data dialog box opens
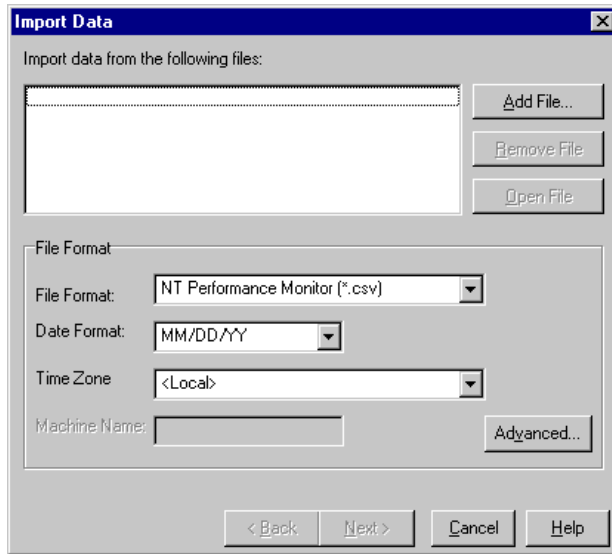
2 Select the format of the external data file from the **File format** list box.

3 Click **Add File**. In the **Select File to Import** dialog box that opens, the **Files of type** list box shows the type chosen in step 2.

4 Set other file format options, as described in "Understanding the Import Data Dialog Box," on page 371. You must enter a machine name.

5 To specify character separators and symbols, click **Advanced**. For more information, see "Understanding the Advanced Settings Dialog Box," on page 373.

6 Click **Next**. The Import Data dialog box opens.

7 Select the type of monitor that generated the external data file. If your monitor type does not exist, you can add it, as described in "Defining Custom Monitor Types for Import," on page 380.

When opening a new graph, you will see your monitor added to the list of available graphs under this particular category (See Opening Analysis Graphs).

8 Click **Finish**. LoadRunner Analysis imports the data file or files, and refreshes all graphs currently displayed in the session.

Note: When importing data into a scenario with two or more cross results, the imported data will be integrated into the last set of results listed in the **File > Cross with Result** dialog box. For more information, see "Generating Cross Result Graphs" on page 330.

Understanding the Import Data Dialog Box

You use the Import Data dialog box to import and integrate non-Mercury Interactive data files into a Analysis session.

Import data from the following files: Displays the files that you selected for import.

Add File: Select an external data file to import. A dialog box opens to enable you to select files.

Remove File: Delete an external data file from the list.

Open File: Open an external data file using the associated application.

File Format: Set the file format options.

File Format: Choose the format of the external data file. For an explanation of available formats, see "Supported File Types," on page 374.

Date Format: Specify the format of the date in the imported data file. For example, for European dates with a 4 digit year, choose *DD/MM/YYYY*

Time Zone: Select the time zone where the external data file was recorded. LoadRunner Analysis compensates for the various international time zones and aligns the times in the file with local time zone settings in order to match LoadRunner results. If the times in the imported file are erroneous by a constant offset, you can synchronize the time, as described in "Synchronizing Erroneous Time," on page 372.

Time Zone also contains the option <*Synchronize with scenario start time*>. Choose this to align the earliest measurement found in the data file to the start time of the LoadRunner scenario.

Machine Name: Specify the machine name the monitor runs on. This associates the machine name with the measurement. For example, a file IO rate on the machine *fender* will be named *File IO Rate:fender*. This enables you to apply Graph settings by the machine name. For more information , see "Filtering and Sorting Graph Data," on page 43.

Synchronizing Erroneous Time

If the times within the imported file are erroneous by a constant offset, select the **Time Zone** option <*User Defined*> to correct the error and synchronize with LoadRunner's results. The **Alter File Times** dialog box opens, where you specify the amount of time to add or subtract from all time measurements in the imported file:
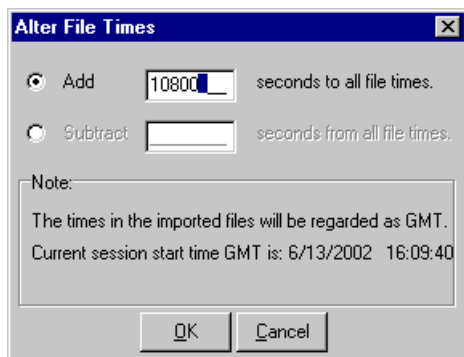
The example above adds 3 hours (10,800 seconds) to all times taken from the imported data file.

Note: When doing this, you should synchronize the time to GMT (and not to Local Time). To help you with this alignment, the dialog displays the scenario start time in GMT.

In the example above, the start time is 16:09:40. Since the clock on the server machine was running slow and produced measurements in the data file beginning at 13:09, 3 hours are added to all time measurements in the file.

Note that when importing data into a scenario with two or more cross results, the imported data will be integrated into the last set of results listed in the **File** > **Cross with Result** dialog box.

Understanding the Advanced Settings Dialog Box

You can use the Advanced Settings Dialog Box to define the data format of the imported file to settings other than of the regional configuration.
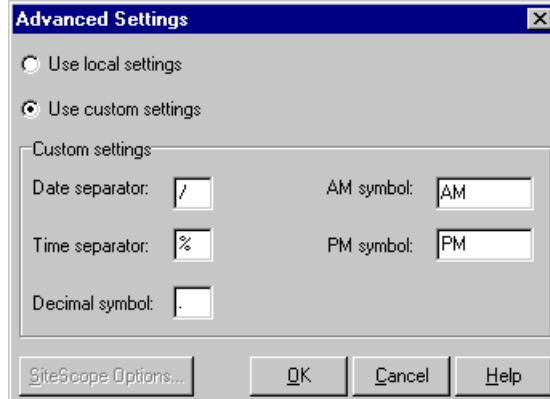
The above example shows a non-standard time separator, the character '%', substituted for the standard ':' separator.

Use local settings: Keep default settings of the regional configuration. Disables the *Custom Settings* area of the dialog box.

Use custom settings: Define your own settings. Enables the *Custom Settings* area of the dialog box.

Date Separator: Enter a custom symbol, for example, the slash ('/') character in 11/10/02

Time Separator: Enter a custom symbol, for example, the colon ':' character in 9:54:19

Decimal symbol: Enter a custom symbol, for example, the '.' character in the number 2.5

AM symbol: Enter a custom symbol for the hours between midnight and noon.

PM symbol: Enter a custom symbol for the hours between noon and midnight.

Supported File Types

The following file types are supported:

➤ NT Performance Monitor (.csv)

➤ Windows 2000 Performance Monitor (.csv)

➤ Standard Comma Separated File (.csv)

➤ Master-Detail Comma Separated File (.csv)

➤ Microsoft Excel File (.xls)

➤ Master-Detail Microsoft Excel file (.xls)

NT Performance Monitor (.csv)

Default file type of NT Performance monitor, in comma separated value (CSV) format. For example:

```
Reported on \\WINTER
Date: 10/23/01
Time: 10:08:39 AM
Data: Current Activity
Interval:   1.000 seconds

,,% Privileged Time,% Processor Time,% User Time,
,,0,0,0,
,,,,,
,,Processor,Processor,Processor,
Date,Time,\\WINTER,\\WINTER,\\WINTER,
10/23/01,10:07:00 AM ,     0.998,     1.174,     0.000,
10/23/01,10:07:01 AM ,     0.000,     0.275,     0.000,
```

Windows 2000 Performance Monitor (.csv)

Default file type of Windows 2000 Performance monitor, but incompatible with NT Performance monitor. In comma separated value (CSV) format. For example:

```
"(PDH-CSV 4.0)","\\MACRON\Processor(_Total)\% Processor Time","\\MACRON\Processor(_Total)\% User
Time","\\MACRON\Processor(_Total)\Interrupts/sec","\\MACRON\System\File Control Bytes/sec"
"10/29/2001
13:09:33.746","99.999148401465547","0.0021716772078191897","997.21487008127474","488.5347931889
"10/29/2001
13:09:48.747","18.157543391188248","8.4112149532710276","1116.5859176246415","9843.2933303122791
"10/29/2001
13:10:03.749","5.941255006675572","1.5353805073431241","1100.9651204860379","623.18277489319848"
```

Standard Comma Separated File (.csv)

This file type has the following format:

Date,Time,Measurement_1,Measurement_2, ...

where fields are comma separated and first row contains column titles

The following example from a standard CSV file shows 3 measurements: an interrupt rate, a file IO rate and a CPU usage. The first row shows an interrupt rate of 1122.19 and an IO rate of 4.18:

```
date, time, interrupt rate, File IO rate, CPU bust percent
25/05/01,10:09:01,1122.19,4.18,1.59
25/05/01,10:10:01,1123.7,6.43,1.42
```

Master-Detail Comma Separated File (.csv)

This file type is identical to Standard Comma Separated Files except for an additional *Master* column which specifies that row's particular breakdown of a more general measurement. For example, a Standard CSV file may contain data points of a machine's total CPU usage at a given moment:

Date,Time,CPU_Usage

However, if the total CPU usage can be further broken up into CPU time per-process, then a Master-Detail CSV file can be created with an extra column *ProcessName*, containing the name of a process.

Each row contains the measurement of a specific process's CPU usage only. The format will be the following:

Date,Time,ProcessName,CPU_Usage

as in the following example:

```
date, time,process name, CPU used, elapsed time used
25/05/01,10:06:01,edaSend,0.1,47981.36
25/05/01,10:06:01,PDS,0,47981.17
```

Microsoft Excel File (.xls)

Created by the Microsoft Excel application. The first row contains column titles.

	A	B	C	D	E
1	date	time	interrupt rate	File IO rate	CPU bust percent
2	25/05/01	10:09:01	1122.19	4.18	1.59
3	25/05/01	10:10:01	1123.7	6.43	1.42
4	25/05/01	10:11:01	1103.62	5.33	1.17
5	25/05/01	10:12:01	1118.89	12.18	2.37
6	25/05/01	10:13:01	1116.89	19.85	3.87
7	25/05/01	10:14:01	1128.12	19.9	4.15
8	25/05/01	10:15:01	1151.98	20.82	4.25
9	25/05/01	10:16:01	1110.1	4.83	1.34

Master-Detail Microsoft Excel file (.xls)

Created by Microsoft's Excel application. The first row contains column titles. It contains an extra *Master* column. For an explanation of this column, see "Master-Detail Comma Separated File (.csv)" on page 376.

	A	B	C	D	E
1	date	time	process name	CPU used	elapsed time used
2	25/05/01	10:06:01	edaSend	0.1	47981.36
3	25/05/01	10:06:01	PDS	0	47981.17
4					

Defining Custom File Formats

If the file format of your import file is not supported, you can define a custom format.

To define a data format of an import file:

1 Choose **Tools** > **External Monitors** > **Import Data**. The Import Data dialog box opens.

2 From the **File Format**s list, select <**Custom File Format**>. The Enter New Format Name dialog box opens.

3 Enter a name for the new format (in this case, *my_monitor_format*):
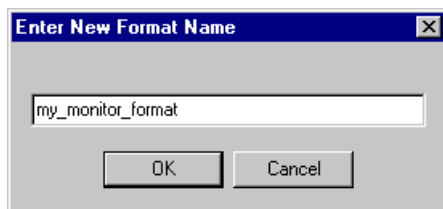
4 Click **OK**. The Define External Format dialog box opens.
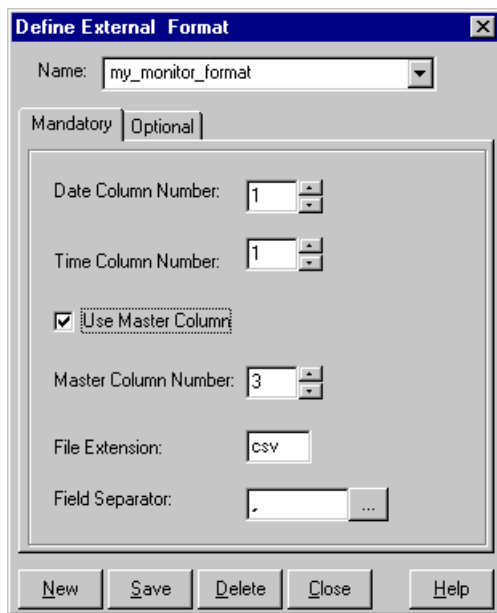
5 Specify the mandatory and optional data, as described in "Understanding the Define External Format Dialog Box," on page 379.

6 Click **Save**.

Understanding the Define External Format Dialog Box

You use the Define External Format dialog box to define a new file format for external data files not supported by Analysis.

The Define External Format dialog box is divided into mandatory and optional information.

The mandatory fields are on the **Mandatory** tab:

Date Column Number: Enter the column that contains the date. If there is a master column (see "Master-Detail Comma Separated File (.csv)" on page 376), specify its number.

Time Column Number: Enter the column that contains the time.

Use Master Column: Select this if the data file contains a master column. A master column specifies the row's particular breakdown of a more general measurement.

File Extension: Enter the file suffix.

Field Separator: Enter the character that separates a field in a row from its neighbor. To select a field separator character, click **Browse** and select a character from the define Field Separator dialog box.

The optional fields are on the **Optional** tab:

Date Format: Specify the format of the date in the imported data file. For example, for European dates with a 4 digit year, choose *DD/MM/YYYY*

Time Zone: Select the time zone where the external data file was recorded. LoadRunner Analysis aligns the times in the file with local time zone settings to match LoadRunner results. (LoadRunner does not alter the file itself).

Machine Name: Specify the machine name the monitor runs on. This associates the machine name with the measurement.

Exclude Columns: Indicate which columns are to be excluded from the data import, such as columns containing descriptive comments. When there is more than one column to be excluded, specify the columns in a comma-separated list e.g. 1.3.8

Convert file from UNIX to DOS format: Monitors often run on UNIX machines. Check this option to convert data files to Windows format. A carriage return (Ascii character 13) is appended to all line feed characters (Ascii character 10) in the UNIX file.

Skip the first [ ] lines: Specify the number of lines at the start of the file to ignore before reading in data. Typically, the first few lines in a file contain headings and sub-headings.

Defining Custom Monitor Types for Import

If your monitor is not included in any of the categories found in the **Monitor Type** list, you can define a new monitor type.

To define a new monitor type:

1 Choose **Tools** > **External Monitors** > **Import Data**. The Import Data dialog box opens. Follow step 2 through 6, as described in "Understanding the Import Data Dialog Box," on page 371.

2 In the **Import Data** dialog box, select **External Monitors** > **Add Custom Monitor**. The Add Custom Monitor dialog box opens.
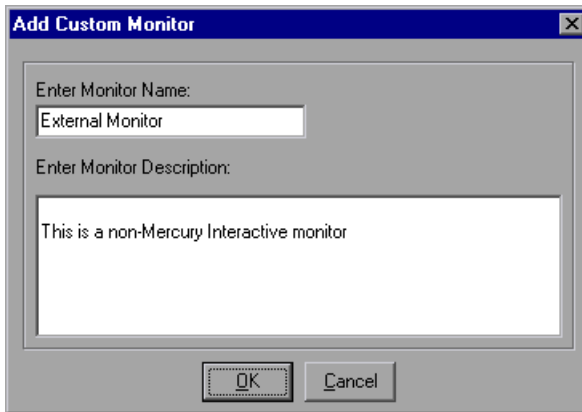
3 Enter a monitor name and description and click **OK**.

The new monitor can now be selected in the list of available monitors.

25

Interpreting Analysis Graphs

LoadRunner Analysis graphs present important information about the performance of your scenario. Using these graphs, you can identify and pinpoint bottlenecks in your application and determine what changes are needed to improve its performance.

This chapter presents examples of:

➤ Analyzing Transaction Performance

➤ Using the Web Page Breakdown Graphs

➤ Using Auto Correlation

➤ Identifying Server Problems

➤ Identifying Network Problems

➤ Comparing Scenario Results

Note: The chapter presents examples from Web load tests.

Analyzing Transaction Performance

The **Average Transaction Response Time** and **Transaction Performance Summary** graphs should be your starting point in analyzing a scenario run. Using the Transaction Performance Summary graph, you can determine which transaction(s) had a particularly high response time during scenario execution. Using the Average Transaction Response Time graph, you can view the behavior of the problematic transaction(s) during each second of the scenario run.

Question 1: Which transactions had the highest response time? Was the response time for these transactions high throughout the scenario, or only at certain points during scenario execution?

Answer: The Transaction Performance Summary graph demonstrates a summary of the minimum, average, and maximum response time for each transaction during scenario execution. In the example below, the response time of the Reservation transaction averaged 44.4 seconds during the course of the scenario.
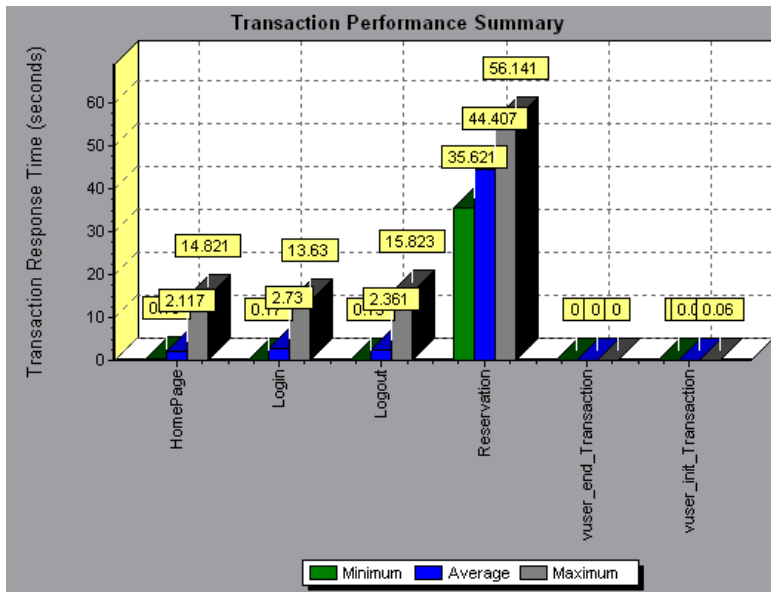
The Average Transaction Response Time graph demonstrates that response time was high for the Reservation transaction throughout the scenario. Response time for this transaction was especially high—approximately 55 seconds—during the sixth and thirteenth minutes of the scenario.
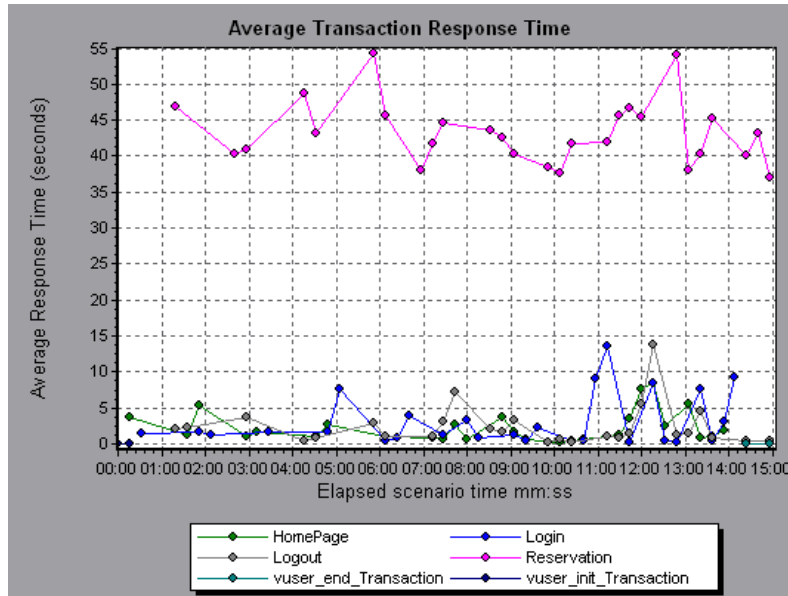
In order to pinpoint the problem and understand why response time was high for the Reservation transaction during this scenario, it is necessary to break down the transactions and analyze the performance of each page component. To break down a transaction, right-click it in the Average Transaction Response Time or Transaction Performance Summary graph, and select **Web Page Breakdown for <transaction name>**.

Using the Web Page Breakdown Graphs

Using the **Web Page Breakdown graphs**, you can drill down on the Average Transaction Response Time or Transaction Performance Summary graphs in order to view the download time for each page component in a transaction. Note that this is possible only if you enabled the Web Page Breakdown feature before running your scenario.

Question 2: Which page components were responsible for the high transaction response time? Were the problems that occurred network- or server-related?

Answer: The Web Page Breakdown graph displays a breakdown of the download time for each page component in the Reservation transaction.
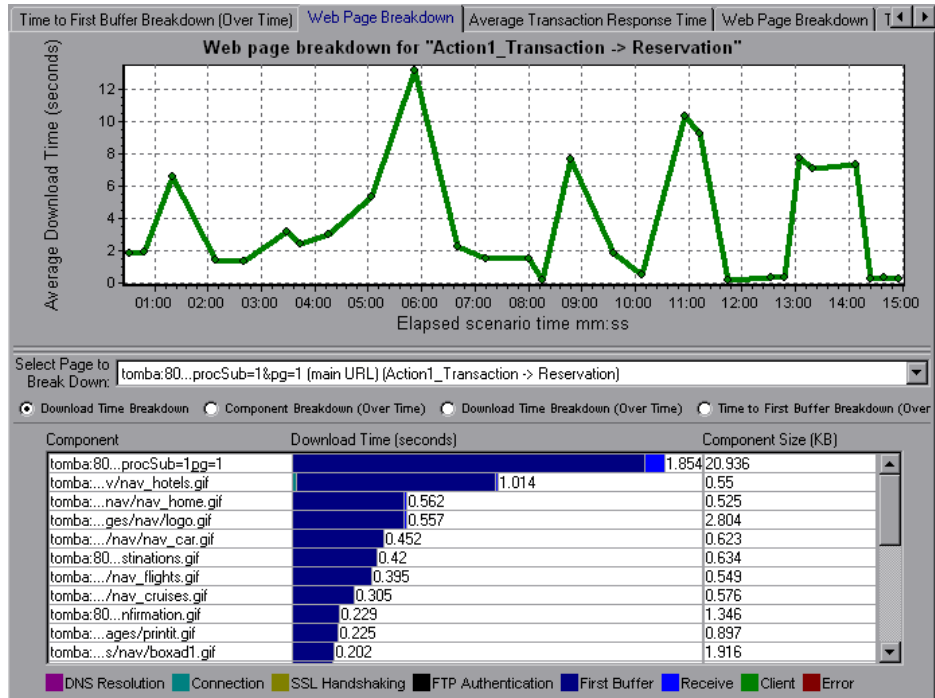
If the download time for a component was unacceptably long, note which measurements—DNS resolution time, connection time, time to first buffer, SSL handshaking time, receive time, and FTP authentication time—were responsible for the lengthy download. To view the point during the scenario at which the problem occurred, select the Page Download Breakdown (Over Time) graph. For more information regarding the measurements displayed, see "Page Download Time Breakdown Graph" on page 119.

To identify whether a problem is network- or server-related, select the Time to First Buffer Breakdown (Over Time).
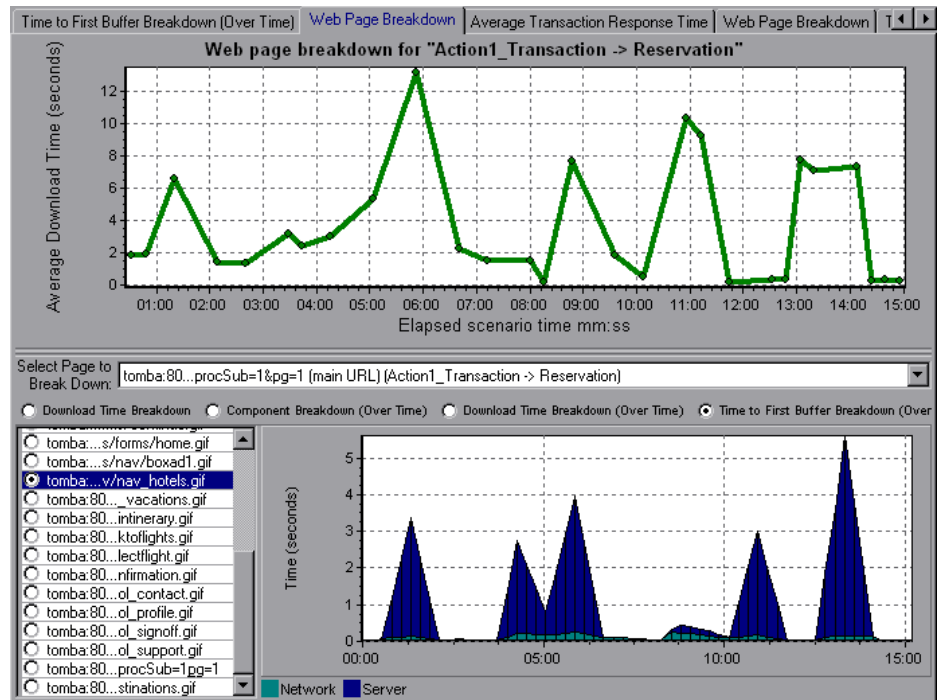
The above graph demonstrates that the server time was much higher than the network time. If the server time is unusually high, use the appropriate server graph to identify the problematic server measurements and isolate the cause of server degradation. If the network time is unusually high, use the Network Monitor graphs to determine what network problems caused the performance bottleneck.

Using Auto Correlation

You can identify the cause of a server or network bottleneck by analyzing the Web Page Breakdown graphs, or by using the auto correlation feature. The auto correlation feature applies sophisticated statistical algorithms to pinpoint the measurements that had the greatest impact on a transaction's response time.

Question 3: Did a bottleneck occur in the system? If so, what was the cause of the problem?

Answer: The Average Transaction Response Time graph displays the average response time during the course of the scenario for each transaction. Using this graph, you can determine which transaction(s) had a particularly high response time during scenario execution.
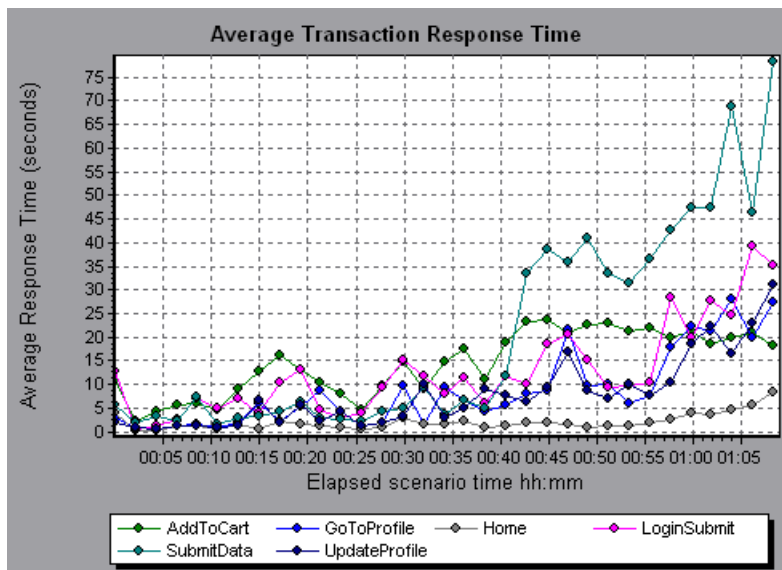
The above graph demonstrates that the response time for the SubmitData transaction was relatively high toward the end of the scenario. To correlate this transaction with all of the measurements collected during the scenario, right-click the SubmitData transaction and select **Auto Correlate**.

In the dialog box that opens, choose the time frame you want to examine.
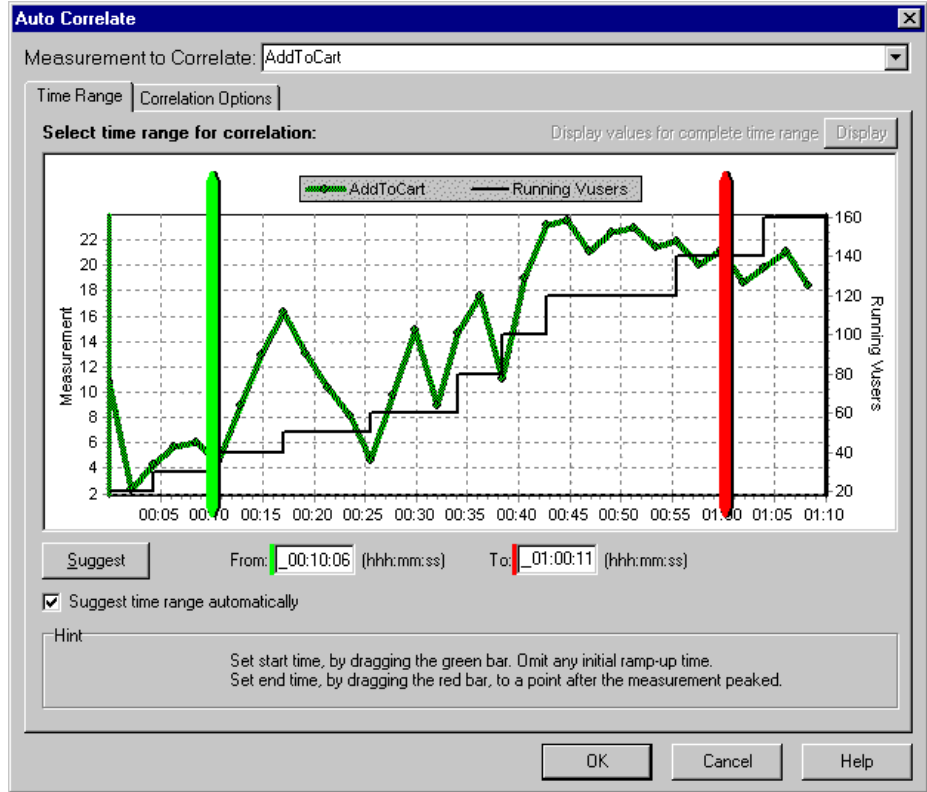
Click the **Correlation Options** tab, select the graphs whose data you want to correlate with the SubmitData transaction, and click **OK**.
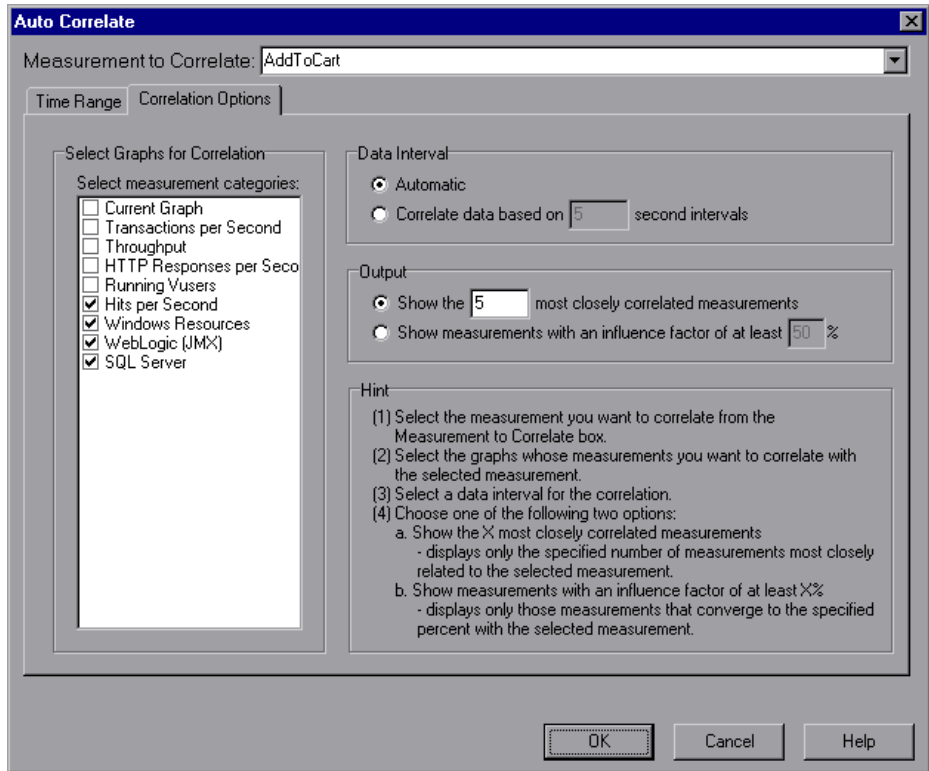
In the following graph, Analysis displays the five measurements most closely correlated with the SubmitData transaction.
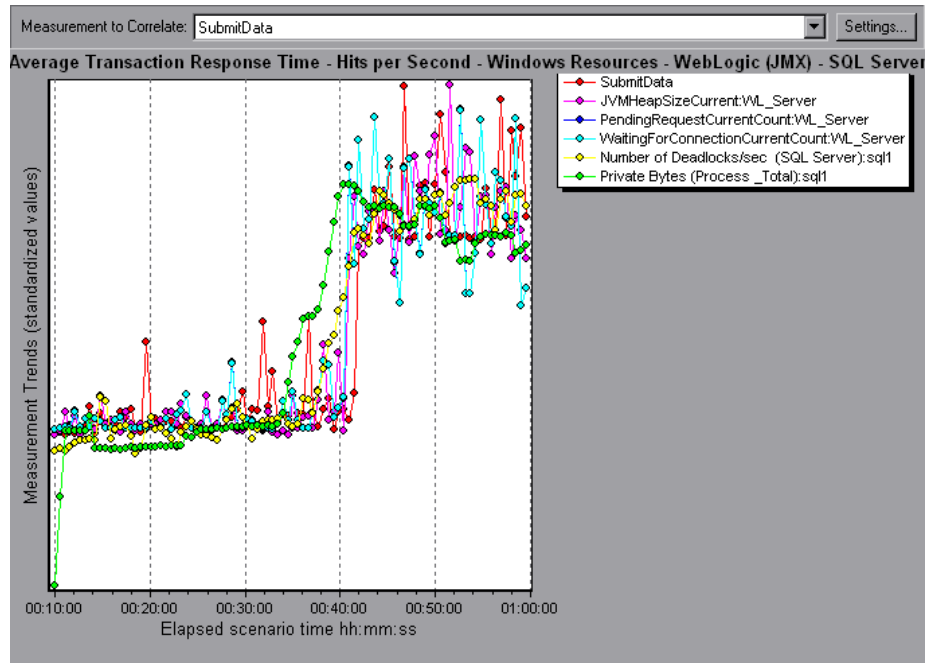
This correlation example demonstrates that the following database and Web server measurements had the greatest influence on the SubmitData transaction: *Number of Deadlocks/sec* (SQL server), *JVMHeapSizeCurrent* (WebLogic server), *PendingRequestCurrentCount* (WebLogic server), *WaitingForConnectionCurrentCount* (WebLogic server), and *Private Bytes (Process_Total)* (SQL server). Using the appropriate server graph, you can view data for each of the above server measurements and isolate the problem(s) that caused the bottleneck in the system.

For example, the graph below demonstrates that both the *JVMHeapSizeCurrent* and *Private Bytes (Process_Total)* WebLogic (JMX) application server measurements increase as the number of running Vusers increases.
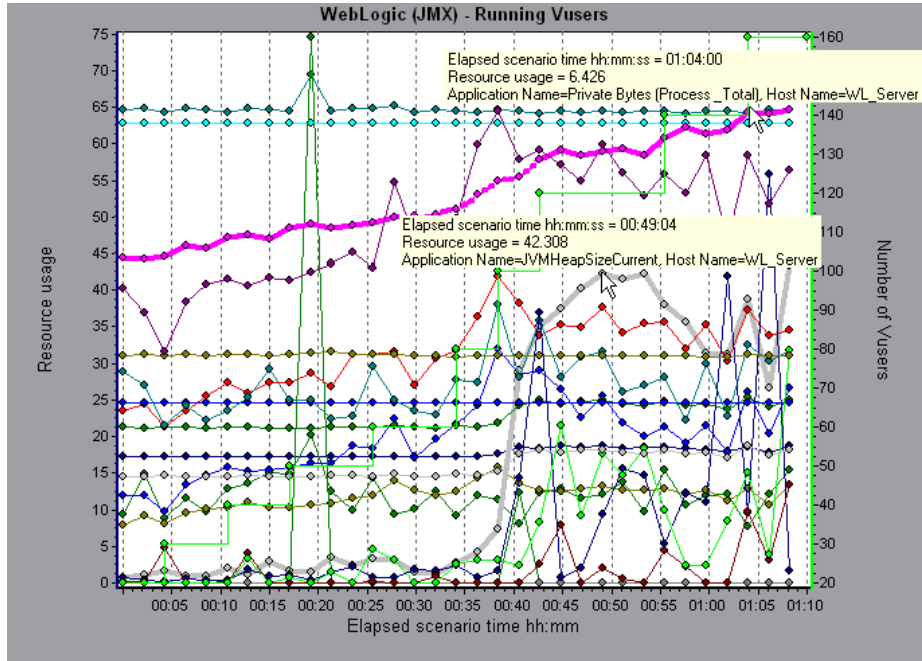
The above graph, therefore, indicates that these two measurements contributed to the slow performance of the WebLogic (JMX) application server, which affected the response time of the SubmitData transaction.

Identifying Server Problems

Web site performance problems can be a result of many factors. Nearly half of the performance problems, however, can be traced to Web, Web application, and database server malfunctioning. Dynamic Web sites that rely heavily on database operations are particularly at risk for performance problems.

The most common database problems are inefficient index design, fragmented databases, out-of-date statistics, and faulty application design. Database system performance can therefore be improved by using smaller result sets, automatically updating data, optimizing indexes, frequently compacting data, implementing a query or a lock on time-outs, using shorter transactions, and avoiding application deadlocks.

In twenty percent of load tests, Web and Web application servers are found to be the cause of performance bottlenecks. The bottlenecks are usually the result of poor server configuration and insufficient resources. For example, poorly written code and DLLs can use nearly all of a computer's processor time (CPU) and create a bottleneck on the server. Likewise, physical memory constraints and mismanagement of server memory can easily cause a server bottleneck. It is therefore advisable to check both the CPU and physical memory of your server before exploring other possible causes of poor Web or Web application server performance.

For information about other useful Web, Web application, and database server measurements, see the *LoadRunner Controller User's Guide*.

HTTPS Problems

Excessive use of HTTPS and other security measures can quickly exhaust server resources and contribute to system bottlenecks. For example, when HTTPS is implemented on the Web server during a load test, system resources are quickly exhausted by a relatively small load. This is caused by secured socket layer (SSL) resource-intensive operations.

Continuously open connections can also drain server resources. Unlike browsers, servers providing SSL services typically create numerous sessions with large numbers of clients. Caching the session identifiers from each transaction can quickly exhaust the server's resources. In addition, the "keep-alive" enhancement features of most Web browsers keep connections open until they are explicitly terminated by the client or server. As a result, server resources may be wasted as large numbers of idle browsers remain connected to the server.

The performance of secured Web sites can be improved by:

➤ Fine-tuning the SSL and HTTPS services according to the type of application

➤ Using SSL hardware accelerators, such as SSL accelerator appliances and cards

➤ Changing the level of security according to the level of sensitivity of the data (i.e., changing the key length used for public-key encryption from 1,024 to 512 bits)

➤ Avoiding the excessive use of SSL and redesigning those pages that have low levels of data sensitivity to use regular HTTPS

Identifying Network Problems

Network bottlenecks can usually be identified when the load increment is considerable and is not significantly affecting any of the server side components, as in the case of informational sites using many static Web pages. In twenty-five percent of such cases, the pipe to the Internet cannot sufficiently handle the desired load and causes delays in incoming and outgoing requests. In addition, bottlenecks are frequently uncovered between the Web site and the ISP.

Using the Network Monitor graphs, you can determine whether, in fact, the network is causing a bottleneck. If the problem is network-related, you can locate the problematic segment so that it can be fixed.

Comparing Scenario Results

Each time the system is fine tuned and another performance bottleneck is resolved, the same load test should be run again in order to verify that the problem has been fixed and that no new performance bottlenecks have been created. After performing the load test several times, you can compare the initial results to the final results.

The following graph displays a comparison of an initial load test and a final load test for a scenario.
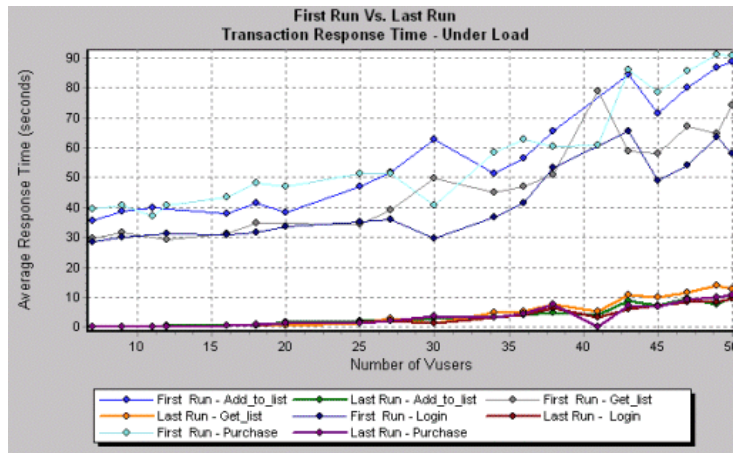
The first load test demonstrates the application's performance in its initial state, before any load testing was performed. From the graph, you can see that with approximately 50 Vusers, response time was almost 90 seconds, indicating that the application suffered from severe performance problems.

Using the analysis process, it was possible to determine what architectural changes were necessary to improve the transaction response time. As a result of these site architectural changes, the transaction response time for the same business process, with the same number of users, was less than 10 seconds in the last load test performed. Using Analysis, therefore, the customer was able to increase site performance tenfold.

Index

MERCURY INTERACTIVE

Mercury Interactive Corporation
1325 Borregas Avenue
Sunnyvale, CA 94089 USA

Main Telephone: (408) 822-5200
Sales & Information: (800) TEST-911, (866) TOPAZ-4U
Customer Support: (877) TEST-HLP
Fax: (408) 822-5300

Home Page: www.mercuryinteractive.com
Customer Support: support.mercuryinteractive.com

LRANUG7. 8/ 01