

LoadRunner®
Analysis User's Guide
Version 7.5



MERCURY INTERACTIVE

LoadRunner Analysis User's Guide, Version 7.5

© Copyright 1994 - 2002 by Mercury Interactive Corporation

All rights reserved. All text and figures included in this publication are the exclusive property of Mercury Interactive Corporation or its licensors, and may not be copied, reproduced, or used in any way without the express permission in writing of Mercury Interactive. Information in this document is subject to change without notice and does not represent a commitment on the part of Mercury Interactive.

Mercury Interactive may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents except as expressly provided in any written license agreement from Mercury Interactive.

Mercury Interactive and Design, M and Design, WinRunner, XRunner, LoadRunner, TestDirector, TestSuite, WebTest, Astra, Astra SiteManager, Astra SiteTest, SiteRunner, FreshWater Software, SiteScope, and SiteSeer are registered trademarks of Mercury Interactive Corporation in the United States and/or other countries. Astra QuickTest, Astra LoadTest, Astra FastTrack, RapidTest, QuickTest, Visual Testing, Action Tracker, Link Doctor, Change Viewer, Dynamic Scan, Fast Scan, Visual Web Display, ActiveTest, ActiveTest SecureCheck, ActiveWatch, POPs on Demand, Topaz, Topaz ActiveAgent, Topaz Observer, Topaz Prism, Topaz Delta, Topaz Rent-a-POP, Topaz Open DataSource, Topaz AIMS, Topaz Console, Topaz Diagnostics, Topaz WeatherMap, Twinlook, TurboLoad, LoadRunner TestCenter, SiteReliance and Global SiteReliance are trademarks of Mercury Interactive Corporation in the United States and/or other countries.

This document may also contain registered trademarks, trademarks, service marks and/or trade names that are owned by their respective companies or organizations. Mercury Interactive Corporation disclaims any responsibility for specifying which marks are owned by which companies or organizations.

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

Mercury Interactive Corporation
1325 Borregas Avenue
Sunnyvale, CA 94089 USA

Table of Contents

Welcome to LoadRunner	ix
Online Resources	ix
LoadRunner Documentation Set.....	x
Using the LoadRunner Documentation Set	xi
Typographical Conventions.....	xiii
Chapter 1: Introducing LoadRunner Analysis	1
About the Analysis	2
Creating Analysis Sessions	3
Starting the Analysis.....	3
Collating Execution Results	4
Viewing Summary Data	4
Aggregating Analysis Data.....	5
Setting the Analysis Time Filter	7
Setting General Options	9
Setting Database Options	10
Working with Templates.....	14
Viewing Session Information	16
Viewing the Scenario Run-Time Settings.....	17
Analysis Graphs	17
Opening Analysis Graphs.....	20

Chapter 2: Working with Analysis Graphs	21
About Working with Analysis Graphs	22
Determining a Point's Coordinates.....	22
Performing a Drill Down on a Graph	23
Enlarging a Section of a Graph	26
Changing the Granularity of the X-Axis.....	26
Applying a Filter to a Graph	29
Grouping and Sorting Results	33
Configuring Display Options	34
Viewing the Legend	38
Viewing the Data as a Spreadsheet and as Raw Data	40
Viewing Measurement Trends.....	42
Auto Correlating Measurements	43
Chapter 3: Vuser Graphs	47
About Vuser Graphs	47
Running Vusers Graph	48
Vuser Summary Graph	49
Rendezvous Graph	50
Chapter 4: Error Graphs	51
About Error Graphs	51
Error Statistics Graph	52
Errors per Second Graph	53
Chapter 5: Transaction Graphs	55
About Transaction Graphs	55
Average Transaction Response Time Graph	56
Transactions per Second Graph	58
Total Transactions per Second	59
Transaction Summary Graph	60
Transaction Performance Summary Graph	61
Transaction Response Time (Under Load) Graph	62
Transaction Response Time (Percentile) Graph	63
Transaction Response Time (Distribution) Graph	64

Chapter 6: Web Resource Graphs	67
About Web Resource Graphs.....	67
Hits per Second Graph	68
Throughput Graph	69
HTTP Status Code Summary Graph	70
HTTP Responses per Second Graph	71
Pages Downloaded per Second Graph	74
Retries per Second Graph	76
Retries Summary Graph	77
Chapter 7: Web Page Breakdown Graphs	79
About Web Page Breakdown Graphs	80
Activating the Web Page Breakdown Graphs	81
Page Component Breakdown Graph	83
Page Component Breakdown (Over Time) Graph	85
Page Download Time Breakdown Graph	87
Page Download Time Breakdown (Over Time) Graph	91
Time to First Buffer Breakdown Graph	93
Time to First Buffer Breakdown (Over Time) Graph	95
Downloaded Component Size Graph	97
Chapter 8: User-Defined Data Point Graphs	99
About User-Defined Data Point Graphs.....	99
Data Points (Sum) Graph	100
Data Points (Average) Graph	101
Chapter 9: System Resource Graphs	103
About System Resource Graphs.....	103
Windows Resources Graph	104
UNIX Resources Graph.....	108
SNMP Resources Graph	110
TUXEDO Resources Graph	111
Chapter 10: Network Monitor Graphs	113
About Network Monitoring	113
Understanding Network Monitoring.....	114
Network Delay Time Graph	115
Network Sub-Path Time Graph	116
Network Segment Delay Graph	117
Verifying the Network as a Bottleneck.....	118
Chapter 11: Firewall Server Monitor Graphs	119
About Firewall Server Monitor Graphs	119
Check Point FireWall-1 Server Graph	120

Chapter 12: Web Server Resource Graphs	121
About Web Server Resource Graphs.....	121
Apache Server Graph	122
Microsoft Information Internet Server (IIS) Graph	124
iPlanet/Netscape Server Graph	126
Chapter 13: Web Application Server Resource Graphs	129
About Web Application Server Resource Graphs.....	130
Ariba Graph	131
ATG Dynamo Graph	133
BroadVision Graph	136
Brokat Twister Graph	143
ColdFusion Graph	147
Fujitsu INTERSTAGE Graph	148
Microsoft Active Server Pages (ASP) Graph	149
Oracle9iAS HTTP Server Graph	150
SilverStream Graph	154
WebLogic (SNMP) Graph	155
WebLogic (JMX) Graph	157
WebSphere Graph	160
Chapter 14: Database Server Resource Graphs	167
About Database Server Resource Graphs.....	167
DB2 Graph	168
Oracle Graph	182
SQL Server Graph	184
Sybase Graph	186
Chapter 15: Streaming Media Graphs	191
About Streaming Media Graphs.....	191
RealPlayer Client Graph	192
RealPlayer Server Graph	194
Windows Media Server Graph	195
Media Player Client Graph.....	197
Chapter 16: ERP Server Resource Graphs	199
About ERP Server Resource Graphs	199
SAP Graph	200
Chapter 17: Java Performance Graphs	203
About Java Performance Graphs	203
EJB Graph	204
JProbe Graph	205
TowerJ Graph	205

Chapter 18: Cross Result and Merged Graphs	209
About Cross Result and Merged Graphs	209
Cross Result Graphs.....	210
Generating Cross Result Graphs	212
Merging Graphs	213
Creating a Merged Graph	215
Chapter 19: Understanding Analysis Reports	217
About Analysis Reports.....	218
Viewing Summary Reports	219
Creating HTML Reports	220
Working with Transaction Reports	221
Scenario Execution Report	223
Failed Transaction Report	224
Failed Vuser Report	225
Data Point Report	226
Detailed Transaction Report	227
Transaction Performance by Vuser Report	228
Chapter 20: Managing Results Using TestDirector	229
About Managing Results Using TestDirector	229
Connecting to and Disconnecting from TestDirector	230
Creating a New Session Using TestDirector	233
Opening an Existing Session Using TestDirector	235
Saving Sessions to a TestDirector Project	236
Chapter 21: Interpreting Analysis Graphs	237
Analyzing Transaction Performance	238
Using the Web Page Breakdown Graphs.....	240
Using Auto Correlation	242
Identifying Server Problems	247
Identifying Network Problems	248
Comparing Scenario Results	249
Index	251

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, 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, refer to the *LoadRunner Installation Guide*. The installation guide explains how to install:

- ▶ the LoadRunner Controller—on a Windows-based machine
- ▶ Virtual User components—for both Windows and UNIX platforms

Controller User's Guide

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

The *LoadRunner Controller User's Guide (Windows)* 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 *LoadRunner Creating Vuser Scripts User's 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	<i>LoadRunner Installation Guide</i>
The LoadRunner testing process	<i>LoadRunner Controller User's Guide (Windows)</i>
Creating Vuser scripts	<i>LoadRunner Creating Vuser Scripts User's Guide</i>
Creating and running scenarios	<i>LoadRunner Controller User's Guide (Windows)</i>
Analyzing test results	<i>LoadRunner Analysis User's Guide</i>

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
<i>Italics</i>	<i>Italic</i> text indicates method or function arguments, file names or paths, and book titles.
Helvetica	The Helvetica 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, < <i>Product installation folder</i> >\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 LoadRunner 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:

- Creating Analysis Sessions
- Starting the Analysis
- Collating Execution Results
- Viewing Summary Data
- Aggregating Analysis Data
- Setting the Analysis Time Filter
- Setting General Options
- Setting Database Options
- Working with Templates
- Viewing Session Information
- Viewing the Scenario Run-Time Settings
- Analysis Graphs
- Opening Analysis Graphs

About the 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 (Windows)*. 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 *LoadRunner Creating Vuser Scripts User's 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 (Windows)*.
- ▶ 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.

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

Creating Analysis Sessions

When you run a scenario, data is stored in a result file with an *.lrr* extension. The 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). The Analysis stores the display information and layout settings for the active graphs in a file with an *.lra* extension.

Starting the Analysis

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

To open the Analysis directly from the Controller, select **Results > Analyze Results**. This option is only available after running a scenario. The 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 the Analysis after it completes scenario execution by selecting **Results > Auto Load Analysis**.

When creating a new session, the Analysis prompts you for the result file (*.lrr* extension) that you wish to include in the session. To open an existing 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, the Analysis will automatically collate the results before generating the analysis data. For more information about collating results, see the *LoadRunner Controller User's Guide (Windows)*.

Viewing Summary Data

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

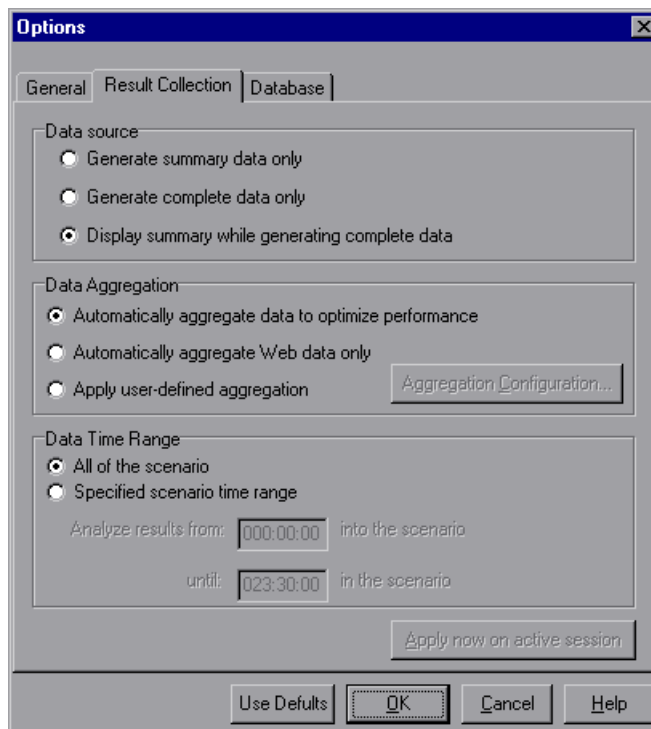
Note: Some fields are not available for filtering when you are working with summary graphs.

Aggregating Analysis Data

If you choose to generate the complete Analysis data, the 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.

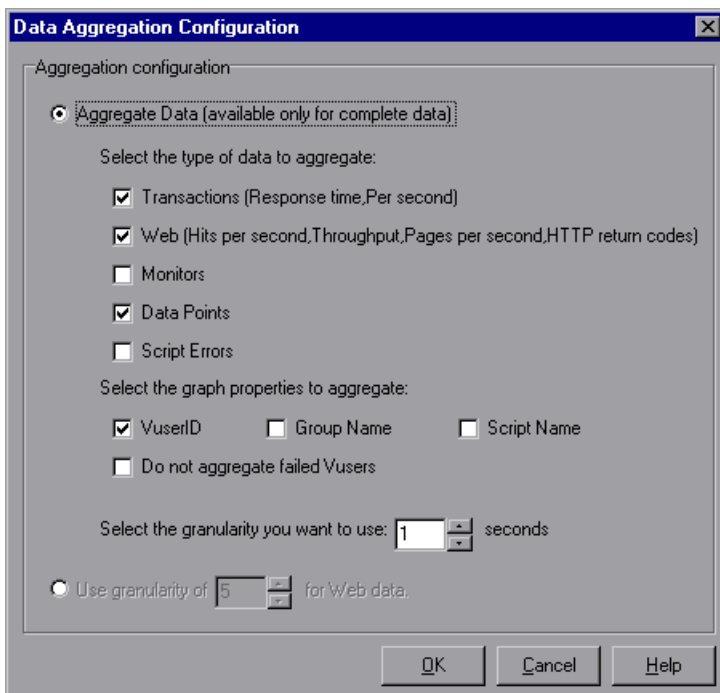
To aggregate Analysis data:

- 1 Select **Tools > Options**, and click the **Result Collection** tab in the Options dialog box.



2 Select one of the following three options:

- **Automatically aggregate data to optimize performance:** Instructs the Analysis to aggregate data using its own, built-in data aggregation formulas.
- **Automatically aggregate Web data only:** Instructs the Analysis to aggregate Web data using its own, built-in data aggregation formulas.
- **Apply user-defined aggregation:** Applies the aggregation settings you define. To define aggregation settings, click the **Aggregation Configuration** button. The Data Aggregation Configuration dialog box opens.



Select **Aggregate Data** if you want to specify the data to aggregate. Specify the type(s) of graphs for which you want to aggregate data, and 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**.

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

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.

Select **Use granularity of xxx seconds for Web data** to specify a custom granularity for Web data. By default, the 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.

Click **OK** to close the Data Aggregation Configuration dialog box.

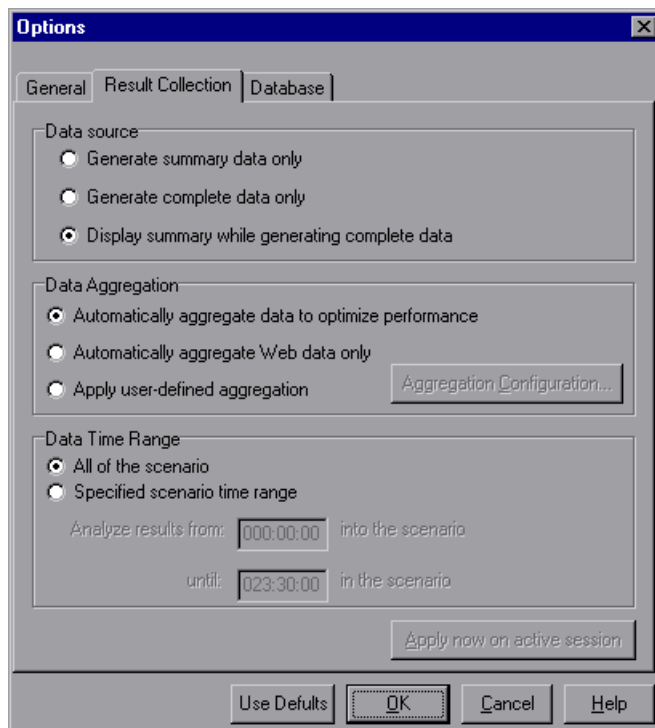
- 3 Click **OK** in the Result Collection tab to save your settings and close the Options dialog box.

Setting the Analysis Time Filter

You can filter the time range of scenario data stored in the database and displayed in the Analysis graphs. This decreases the size of the database and thereby decreases processing time.

To filter the time range of scenario data:

- 1 Select **Tools > Options**, and click the **Result Collection** tab in the Options dialog box.



- 2 In the Data Time Range box, select **Specified scenario time range**.
- 3 Enter the beginning and end of the scenario time range you want the Analysis to display.
- 4 Click **OK** to save your settings and close the Options dialog box.

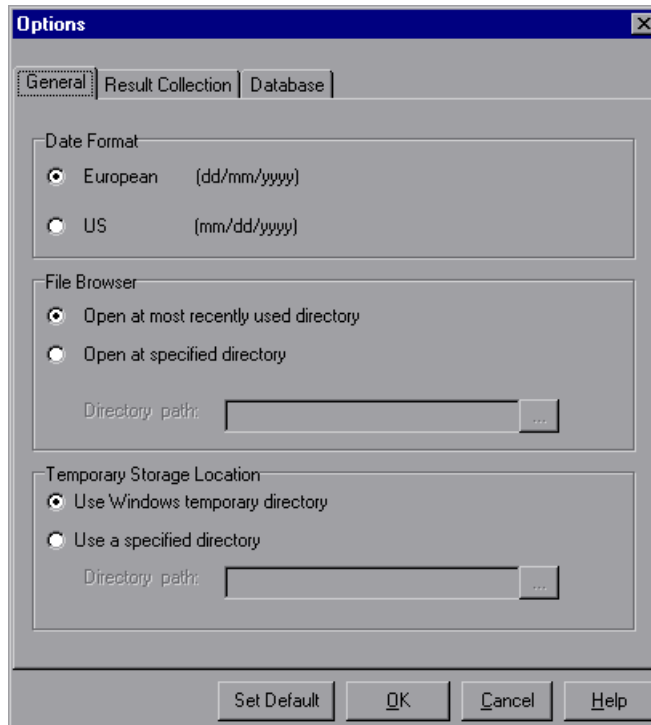
Note: All graphs except the Running Vusers graph will be affected by the time range settings, in both the Summary and Complete Data Analysis modes.

Setting General Options

In the General tab of the Options dialog box, you can choose whether you want dates stored and displayed in a European or U.S. format. In addition, you can select the directory location at which you want the file browser to open, and in which you want to store temporary files.

To configure General options:

- 1 Select **Tools > Options**. The Options dialog box opens, displaying the General tab.



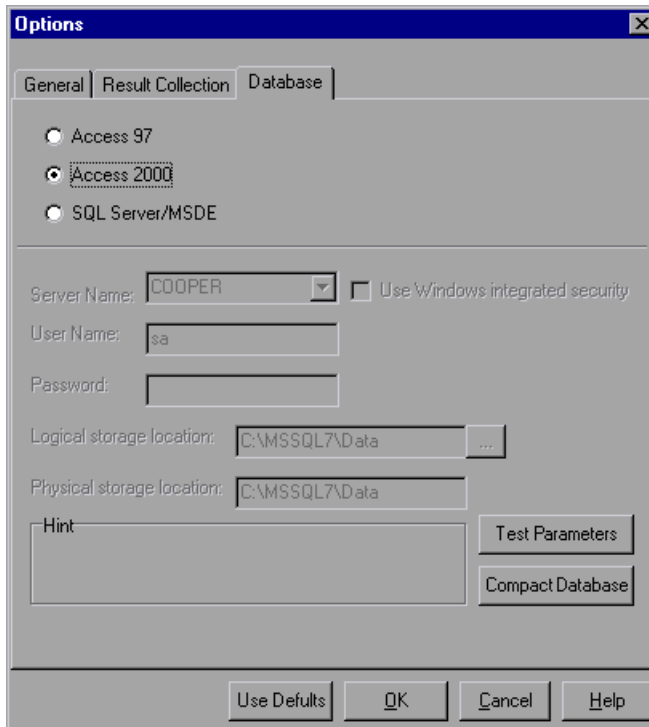
- 2** In the Date Format box, select **European** to store and display dates in the European format, or **US** to store and display dates in the American format.
- 3** In the File Browser box, select **Open at most recently used directory** if you want the Analysis to open the file browser at the previously used directory location. Select **Open at specified directory** and enter the directory location at which you want the file browser to open, if you want the Analysis to open the file browser at a specified directory.
- 4** In the Temporary Storage Location box, select Use Windows temporary directory if you want the Analysis to store temporary files in your Windows temp directory. Select **Use a specified directory** and enter the directory location in which you want to save temporary files, if you want the Analysis to save temporary files in a specified directory.

Setting Database Options

The Database Options tab lets you specify the database in which to store Analysis session result data. By default, LoadRunner stores Analysis result data in an Access 97 database. If your Analysis result data exceeds two gigabytes, it is recommended that you store it on an SQL server or MSDE machine.

To configure Database options:

- 1 Select **Tools > Options** to open the Options dialog box. Select the **Database** tab.



2 Choose one of the following three options:

- ▶ To instruct LoadRunner to save Analysis result data in an Access 97 database format, select **Access 97**.
- ▶ To instruct LoadRunner to save Analysis result data in an Access 2000 database format, select **Access 2000**.

Click the **Test Parameters** button to 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.

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. Click the **Compact Database** button to repair and compress your results and optimize your Access database.

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

- ▶ To instruct LoadRunner to save Analysis result data on an SQL Server or MSDE machine, select **SQL Server/MSDE**. Select or enter the name of the machine on which the SQL server or MSDE is running, and enter the master database user name and password. Select **Use Windows-integrated security** in order to use your Windows login, instead of specifying a user name and password.

Note: By default, the user name “sa” and no password are used for the SQL server.

In the Logical storage location box, 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>. Note that Analysis results stored on an SQL server/MSDE machine can only be viewed on the machine's local LAN.

In the Physical storage location box, 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>.

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

Click **Test Parameters** to connect to the SQL server/MSDE machine and verify that the shared directory you specified exists on the server, and that you have write permissions on the shared server directory. If so, the Analysis synchronizes the shared and physical server directories.

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**.

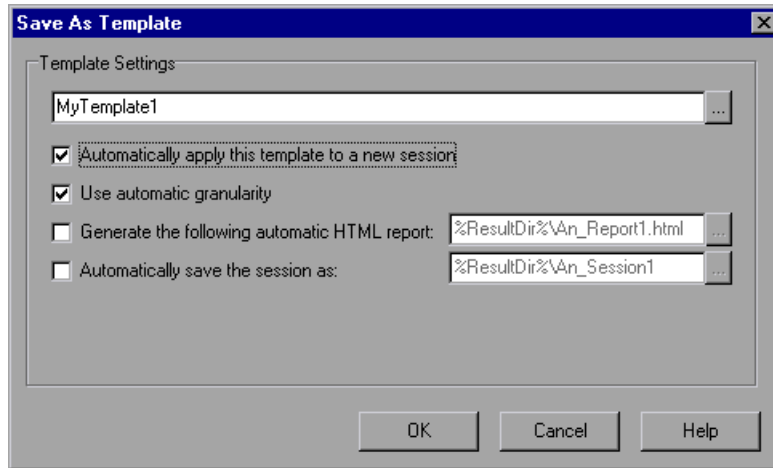
To open a session stored on an SQL Server/MSDE machine, the machine must be running and the directory you defined must exist as a shared directory.

Working with Templates

You can save the filter and display options that you specified for an Analysis session in order to use them later in another session.

To save the Analysis template that you created:

- 1 Select **Tools > Templates > Save As Template**. The Save As Template dialog box opens.



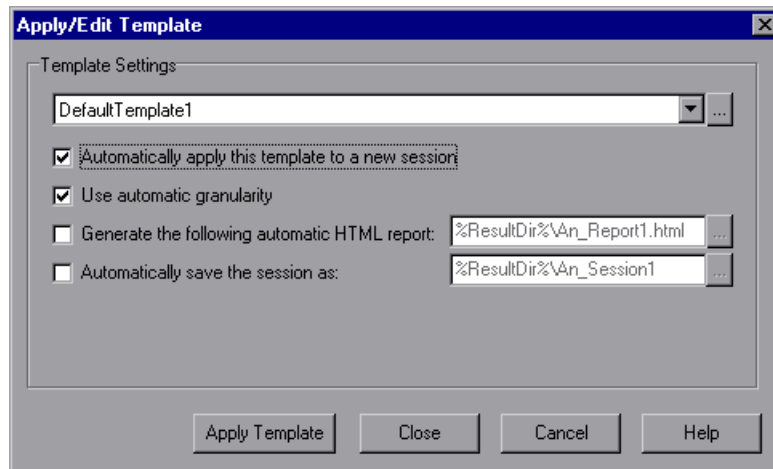
- 2 Enter the name of the template you want to create, or select the template name using the browse button to the right of text box.
- 3 To apply the template to any new session you open, select **Automatically apply this template to a new session**.
- 4 To apply the default Analysis granularity (one second) to the template, select **Use automatic granularity**. For information about setting Analysis granularity, see “Changing the Granularity of the X-Axis” on page 26.
- 5 To generate an HTML report using the template, select **Generate the following automatic HTML report**, and specify or select a report name. For information about generating HTML reports, see “Creating HTML Reports” on page 220.
- 6 To instruct the Analysis to automatically save the session using the template you specify, select **Automatically save the session as**, and specify or select a file name.

- 7 Click **OK** to close the dialog box and save the template.

Once you have saved a template, you can apply it to other Analysis sessions.

To use a saved template in another Analysis session:

- 1 Select **Tools > Templates > Apply/Edit Template**. The Apply/Edit Template dialog box opens.



- 2 Select the template you want to use from the template selector, or click the browse button to choose a template from a different location.
- 3 To apply the template to any new session you open, select **Automatically apply this template to a new session**.
- 4 To apply the default Analysis granularity (one second) to the template, select **Use automatic granularity**. For information about setting Analysis granularity, see “Changing the Granularity of the X-Axis” on page 26.
- 5 To generate an HTML report using the template, select **Generate the following automatic HTML report**, and specify or select a report name. For information about generating HTML reports, see “Creating HTML Reports” on page 220.
- 6 To instruct the Analysis to automatically save the session using the template you specify, select **Automatically save the session as**, and specify or select a file name.

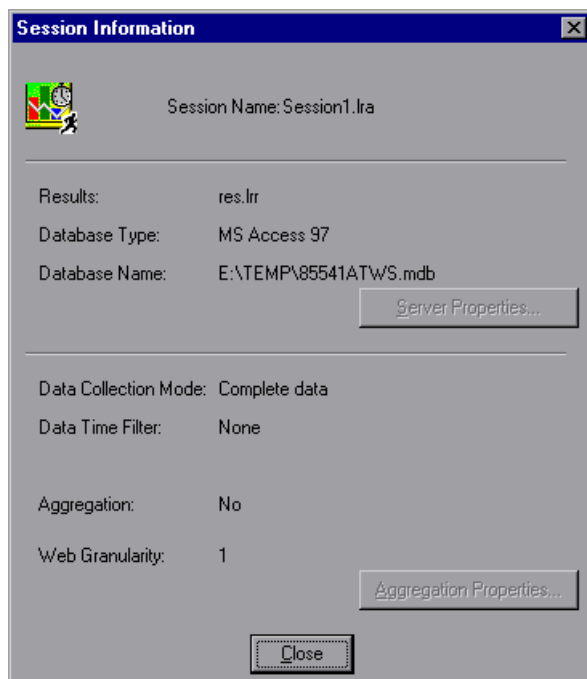
Click **OK** to close the dialog box and load the template you selected into the current Analysis session.

Viewing Session Information

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

To view the properties of the current Analysis session:

Select **File > Session Information**. The Session Information dialog box opens, enabling you to view the session name, result file name, and database name, directory path, and type. In addition, you can view whether the session contains complete or summary data, whether a time filter has been applied to the session, and whether the data was aggregated. The Web granularity used in the session is also displayed.



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 *LoadRunner Creating Vuser Scripts User's Guide*.

To view the scenario run-time settings:



Select **File > Scenario Run-Time Settings**, or click the **Run-Time Settings** button 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.

Analysis Graphs

The Analysis graphs are divided into the following categories:

- Vusers
- Errors
- Transactions
- Web Resources
- Web Page Breakdown
- User-Defined Data Points
- System Resources
- Network Monitor
- Firewalls

- Web Server Resources
- Web Application Server Resources
- Database Server Resources
- Streaming Media
- ERP Server Resources
- Java Performance

Vuser graphs display 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. For more information, see Chapter 4, “Error Graphs.”

Transaction graphs and reports 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 display 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 show statistics relating to the system resources that were monitored during the scenario using the online monitor. This category also includes graphs for SNMP and TUXEDO 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 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, 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 such as Ariba, ATG Dynamo, BroadVision, Brokat Twister, ColdFusion, Fujitsu INTERSTAGE, Microsoft ASP, Oracle9iAS HTTP, SilverStream, WebLogic (SNMP), WebLogic (JMX), WebSphere, and WebSphere (EPM). For more information see Chapter 13, “Web Application Server Resource Graphs.”

Database Server Resource graphs provide information about DB2, Oracle, SQL Server, and Sybase database resources. For more information, see Chapter 14, “Database Server Resource Graphs.”

Streaming Media graphs provide information about RealPlayer Client, RealPlayer Server, and Windows Media Server resource usage. For more information, see Chapter 15, “Streaming Media Graphs.”

ERP Server Resource graphs provide information about SAP R/3 system server resource usage. For more information, see Chapter 16, “ERP Server Resource Graphs.”

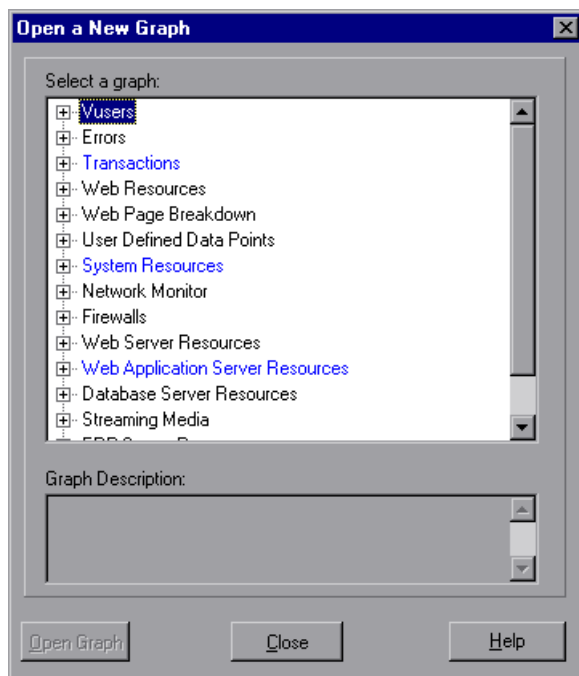
Java Performance graphs provide information about resource usage of Enterprise Java Bean (EJB) objects, Java-based applications, and the TowerJ Java virtual machine. For more information, see Chapter 17, “Java Performance Graphs.”

Opening Analysis Graphs

By default, LoadRunner displays only the Summary Report in the graph tree view. You can open the Analysis graphs 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.



- 2 Click the "+" to 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**. The Analysis generates the graph you selected and adds it to the graph tree view. The graph is displayed in the right pane of the Analysis.

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

2

Working with Analysis Graphs

The Analysis contains several utilities that allow you to customize result data and view this data in various different ways.

This chapter describes:

- About Working with Analysis Graphs
- Determining a Point's Coordinates
- Performing a Drill Down on a Graph
- Enlarging a Section of a Graph
- Changing the Granularity of the X-Axis
- Applying a Filter to a Graph
- Grouping and Sorting Results
- Configuring Display Options
- Viewing the Legend
- Viewing the Data as a Spreadsheet and as Raw Data
- Viewing Measurement Trends
- Auto Correlating Measurements

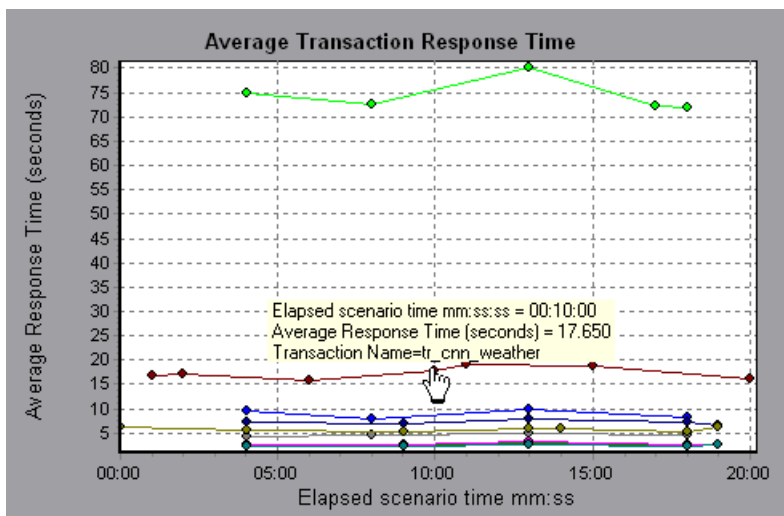
About Working with Analysis Graphs

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

Using the Analysis utilities, you can determine a point's coordinates, drill down on a graph, enlarge a section of a graph, change the granularity of a graph's x-axis, apply a filter to a graph, group and sort result data, configure a graph's display options, view a graph's data as a spreadsheet or as raw data, view measurement trends, or auto correlate between measurements in a graph.

Determining a Point's Coordinates

You can determine the coordinates and values at specific points in a graph. Place the cursor over the point you want to evaluate. The Analysis displays the axis values and other grouping information.

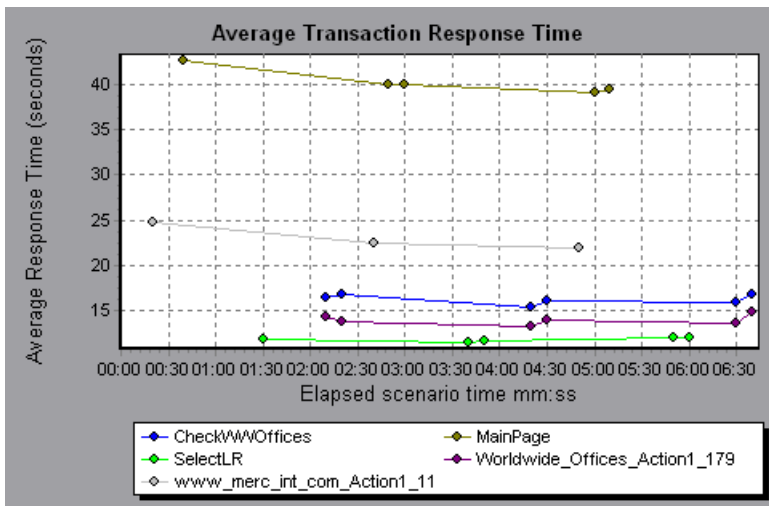


Performing a Drill Down on a Graph

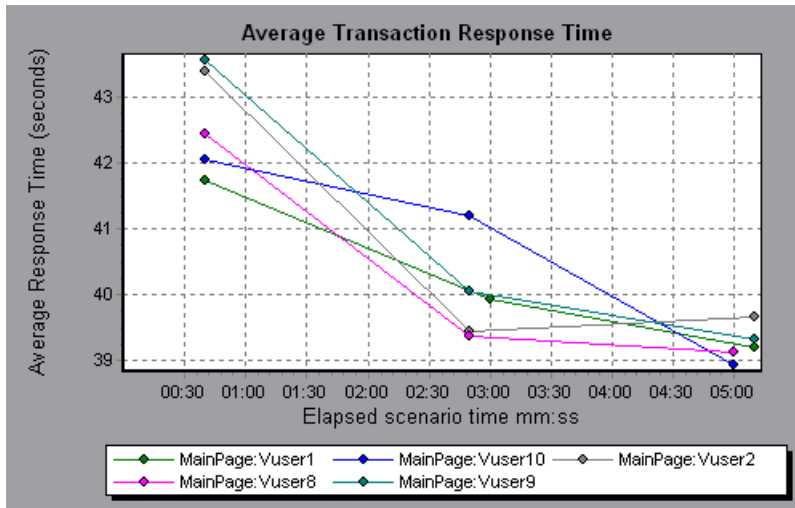
The Analysis provides another filtering mechanism called *drill down*. Drill down lets you focus on a specific measurement within your graph. You select one measurement within the graph and display it according to desired grouping. The available groupings are Vuser ID, Group Name, Vuser End Status, etc. depending 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.

In the following example, the graph shows a line for each of the five transactions.



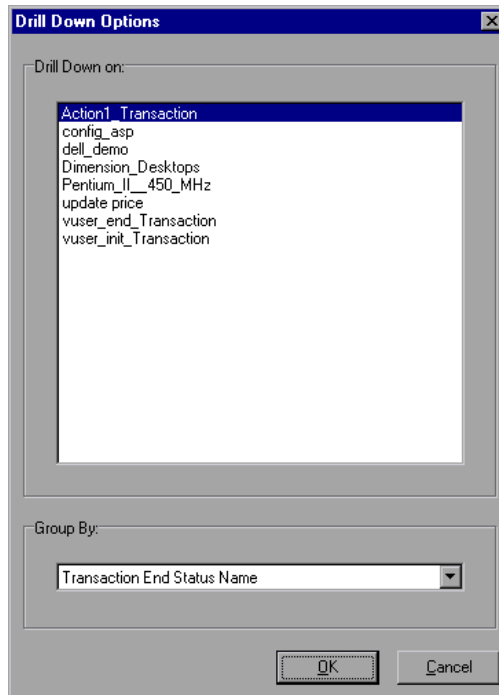
In a drill down for the MainPage transaction per Vuser ID, the graph displays the response time only for the MainPage transaction, one line per Vuser.



You can see from the graph that the response time was longer for some Vusers than for others.

To 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**. The Analysis drills down and displays the new graph.
- 5 To undo the last drill down settings, choose **Undo Set Filter/Group By** from the right-click menu.
- 6 To perform additional drill downs, repeat steps 1 to 4.
- 7 To clear all filter and drill down settings, choose **Clear Filter/Group By** from the right-click menu.

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:

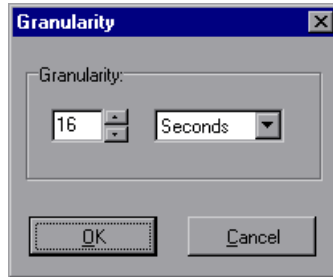
- 1** Click inside a graph.
- 2** Move the mouse pointer to the beginning of the section you want to enlarge, but not over a line of the graph.
- 3** Hold down the left mouse button and draw a box around the section you want to enlarge.
- 4** Release the left mouse button. The section is enlarged.
- 5** To restore the original view, choose **Clear Display Option** from the right-click menu.

Changing the Granularity of the X-Axis

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, the Analysis automatically adjusts the minimum granularity of graphs with ranges of 500 seconds or more.

To change the granularity of a graph:

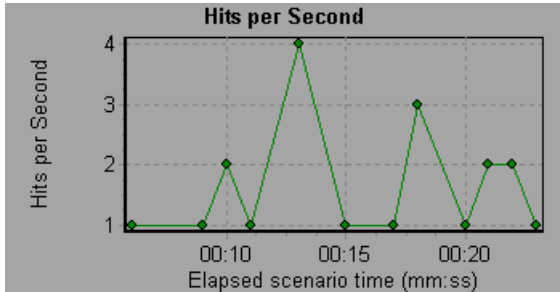
- 1 Click inside a graph.
- 2 Select **View > Set Granularity**, or click the **Set Granularity** button. The Granularity dialog box opens.



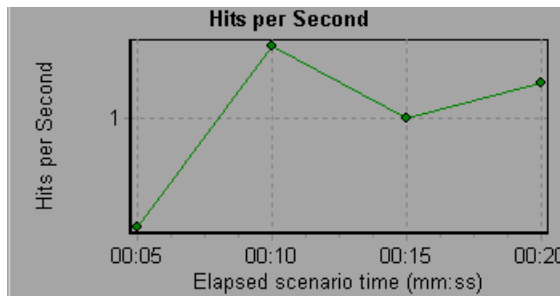
- 3 Enter a new granularity value in milliseconds, minutes, seconds, or hours.
- 4 Click **OK**.

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

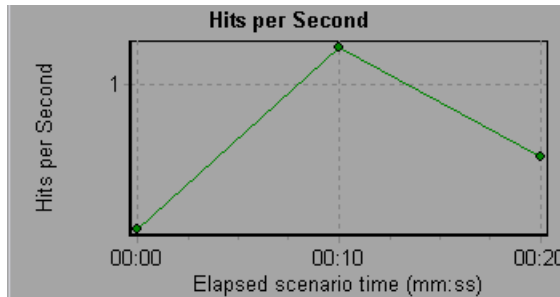
granularity of 5, the y-axis shows the number of hits per second for every five-second period of the scenario.



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.

Applying a Filter to a Graph

When viewing graphs in an Analysis session, you can specify to display only the desired information. If, by default, all transactions are displayed for the entire length of the scenario, you can filter a graph to show fewer transactions for a specific segment of the scenario. For example, you can display four transactions beginning from five minutes into the scenario and ending three minutes before the end of the scenario.

The 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.

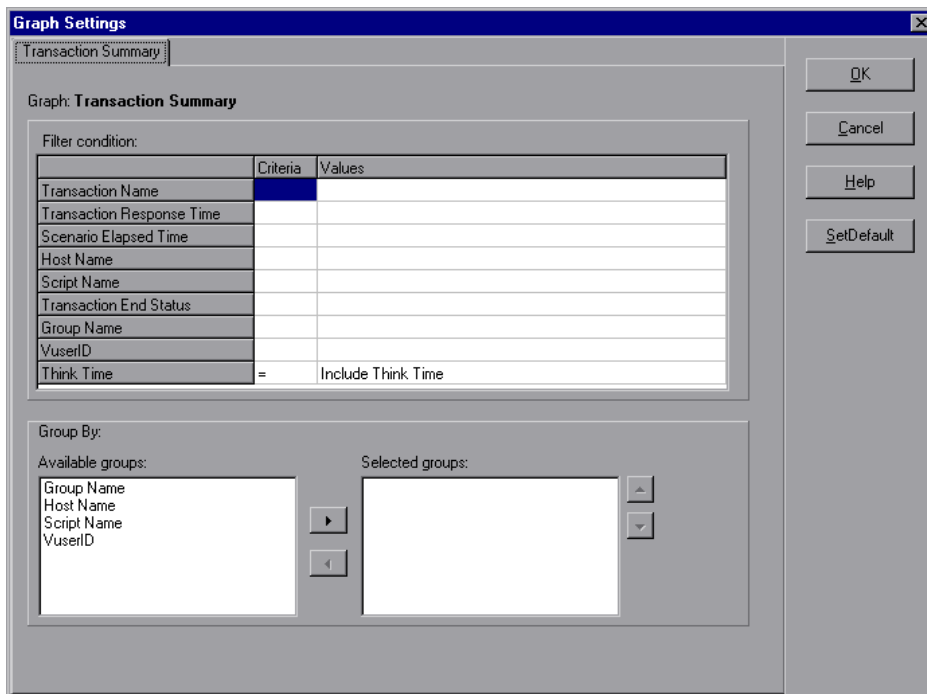
You can also activate global filter conditions that will apply to all graphs in the session. The available global filter conditions are a combination of the filter conditions that apply to the individual graphs.

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

To set a filter condition for an individual graph:

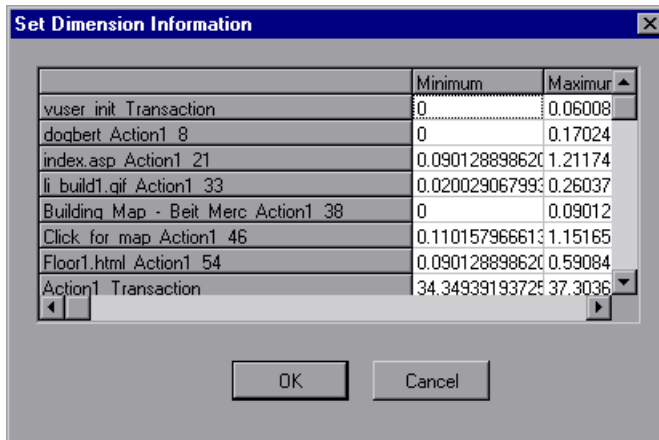
- 1 Select the graph you want to filter by clicking the graph tab or clicking the graph name in the tree view.

- 2 Choose **View > Set Filter/Group By**. The Graph Settings dialog box opens.

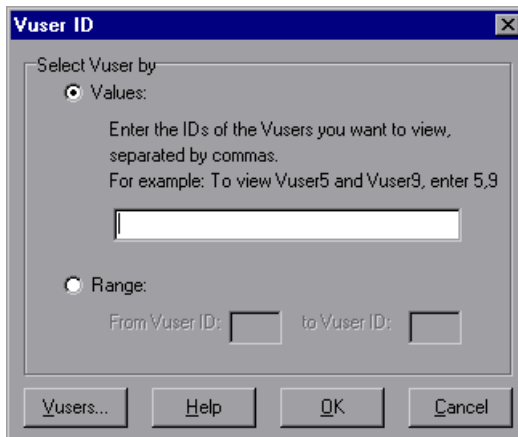


- 3 Click the Criteria box of the condition(s) you want to set, and select either "=" or "<>" (does not equal) from the drop down list.
- 4 Click the Values box of the filter condition you want to set, and select a value from the drop down box. For several filter conditions, a new dialog box opens.

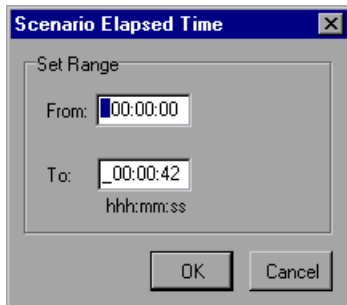
- 5 For the Transaction Response Time filter condition, the Set Dimension Information dialog box opens. Specify a minimum and maximum transaction response time for each transaction.



- 6 For the Vuser ID filter condition, the Vuser ID dialog box opens. Specify the Vuser IDs, or the range of Vusers, you want the graph to display.



- 7 For the Scenario Elapsed Time filter condition, the Scenario Elapsed Time dialog box opens. Specify the start and end time for the graph in hours:minutes:seconds format. The time is relative to the start of the scenario.



- 8 For Rendezvous graphs, while setting the *Number of Released Vusers* condition, the Set Dimension Information dialog box opens. Specify a minimum and maximum number of released Vusers.
- 9 For all graphs that measure resources (Web Server, Database Server, etc.), when you set the *Resource Value* condition, the Set Dimension Information dialog box opens displaying a full range of values for each resource. Specify a minimum and maximum value for the resource.
- 10 Click **OK** to accept the settings and close the Graph Settings dialog box.

To apply a global filter condition for all graphs in the session (both those displayed and those that have not yet been opened), choose **File > Set Global Filter** or click the **Set Global Filter** button, and set the desired filters.

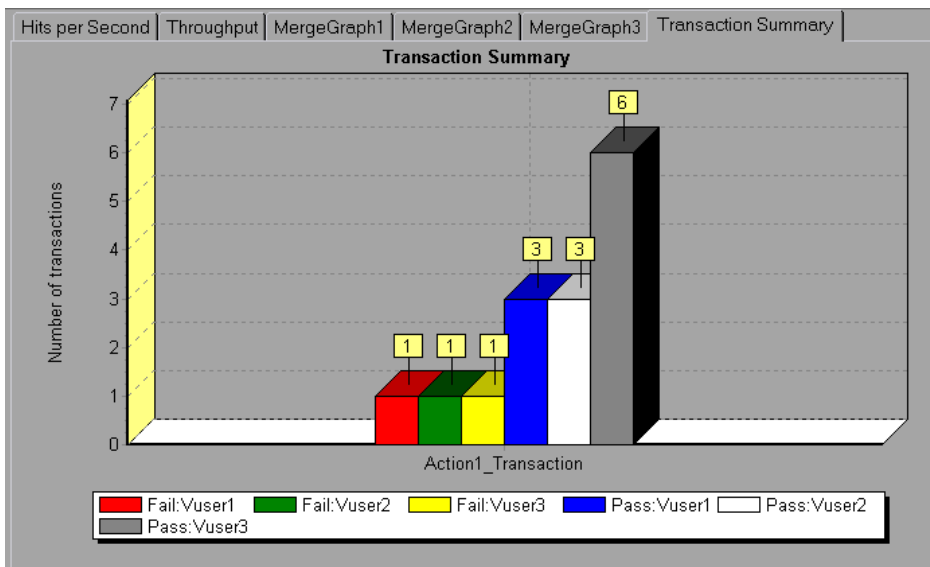


Note: You can set the same filter conditions described above for the Summary Report. To set filter conditions for the Summary Report, select **Summary Report** in the graph tree view, and choose **View > Summary Filter**. Select the filter conditions you want to apply to the Summary Report in the Analysis Summary Filter dialog box.

Grouping and Sorting Results

When viewing a graph, you can group the data in several ways. For example, Transaction graphs can be grouped by the Transaction End Status. The Vuser graphs can be grouped by the Scenario Elapsed Time, Vuser End Status, Vuser Status, and VuserID.

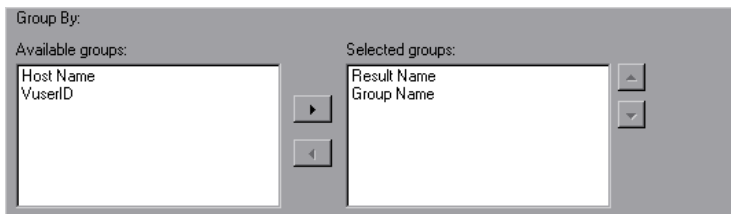
You can also sort by more than one group—for example by Vuser ID and then Vuser status. The results are displayed in the order in which the groups are listed. You can change the order in which things are grouped by rearranging the list. The following graph shows the Transaction Summary grouped according to Vusers.



To sort the graph data according to groups:

- 1** Select the graph you want to sort by clicking the graph tab or clicking the graph name in the tree view.
- 2** Choose **View > Set Filter/Group By**. The Graph Settings dialog box opens.

- 3 In the Available Groups box, select the group by which you want to sort the results.



- 4 Click the right-facing arrow to move your selection to the Selected Groups box.
- 5 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.
- 6 To remove an existing grouping, select an item in the Selected Groups box and click the left-facing arrow to move it to the Available Groups box.
- 7 Click **OK** to close the dialog box.

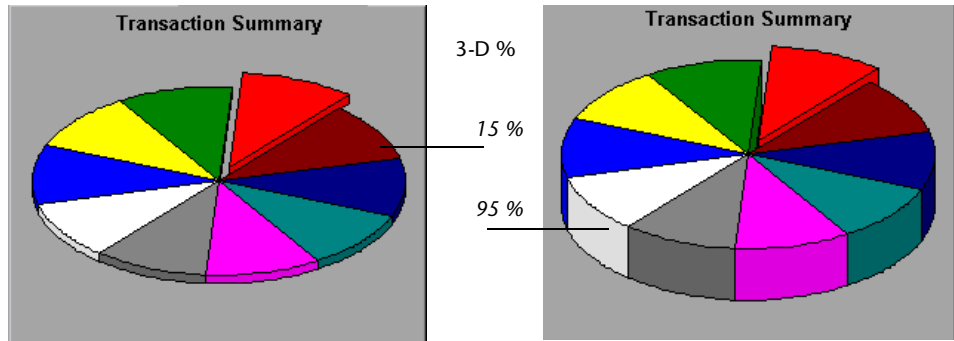
Configuring Display Options

You can configure both standard and advanced display options for each graph. The standard options let you select the type of graph and the time setting. The advanced options allow you to modify the scale and format of each graph.

Standard Display Options

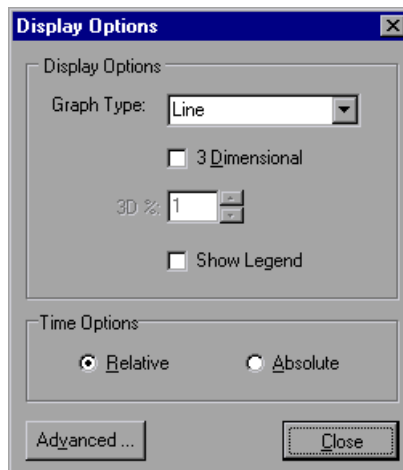
The standard display options let you choose the type of graph to display: line, point, bar, or pie graph. Not all options are available for all graphs.

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 standard 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.

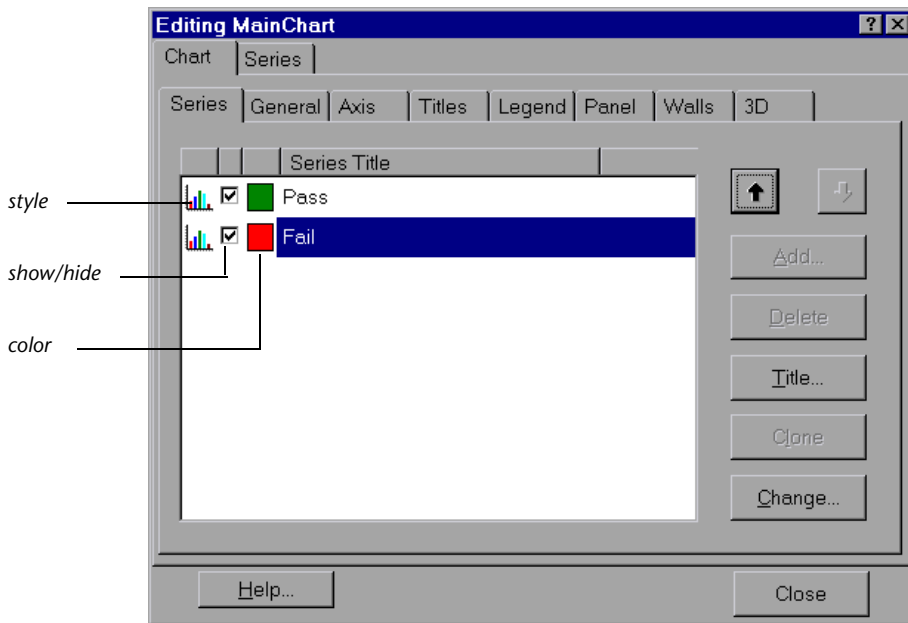
To open the Display Options dialog box, choose **View > Display Options** or click the **Display Options** button.



Choose a graph type, specify a three-dimensional percent, select whether you want a legend to be displayed, and/or choose a time option. Click **Close** to accept the settings and close the dialog box.

Advanced Display Options

The Advanced options let you configure the look and feel of your graph as well as its title and the format of the data. To set the Advanced options, you must first open the Display Options dialog box. Choose **View > Display Options** or click the **Display Options** button. To access the Advanced options, click **Advanced**. The Editing MainChart dialog box opens.



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

Chart Settings

The Chart settings control the look and feel of the entire graph—not the individual points. You set Chart preferences using the following tabs: Series, General, Axis, Titles, Legend, Panel, Walls, and 3D.

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

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

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

Titles: allows you to set the title of the graph, its font, background color, border, and alignment.

Legend: includes 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: lets you set colors for the walls of three-dimensional graphs.

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

Series Settings

The Series settings control the appearance of the individual points plotted in the graph. You set Series preferences using the following tabs: Format, Point, General, and Marks.

Format: allows you to set the border color, line color, pattern and invert property for the lines or bars in your graph.

Point: displays the point properties. Points appear at various points within your line graph. You can set the size, color and shape of these points.

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

Marks: allows you to display the value for each point in the graph and configure the format of those marks.

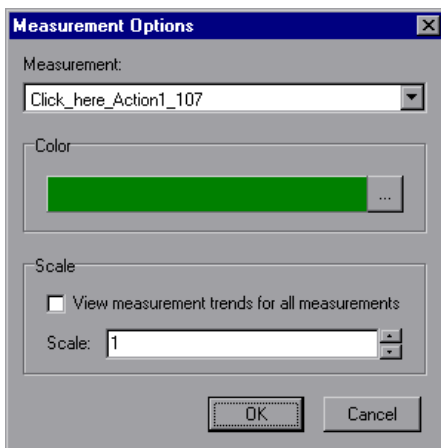
Viewing the Legend

The Legend tab displays the color, scale, minimum, maximum, average, median, and standard deviation of each measurement appearing in the graph.

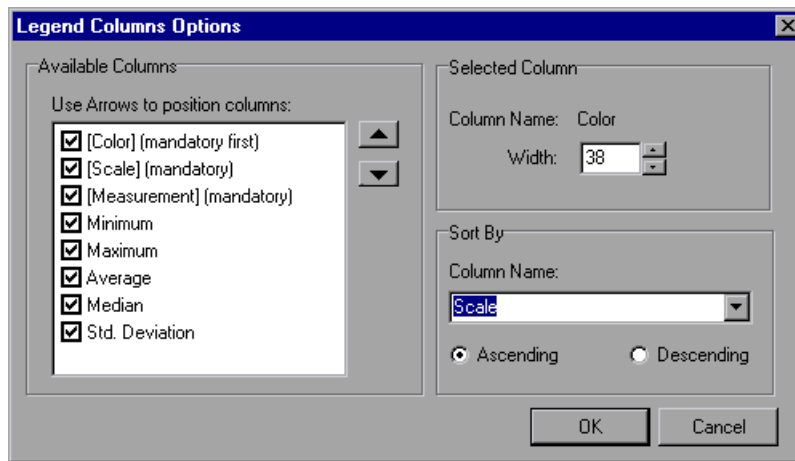
Legend Graph Details User Notes Graph Data Raw Data								
Color	Scale	Measurement	Minimum	Maximum	Average	Median	Std. Deviation	
<input checked="" type="checkbox"/>	1	air_head2_gif_Action1_159	0.176	0.657	0.247	0.19	0.155	
<input checked="" type="checkbox"/>	1	beige_gif_Action1_183	0.167	0.434	0.24	0.194	0.092	
<input checked="" type="checkbox"/>	1	book_gif_Action1_167	0.198	0.812	0.291	0.2	0.2	
<input checked="" type="checkbox"/>	1	bullet1_gif_Action1_234	0.11	0.245	0.164	0.17	0.04	
<input checked="" type="checkbox"/>	1	butt_ast_gif_Action1_207	0.13	0.238	0.177	0.176	0.027	
<input checked="" type="checkbox"/>	1	Click_here_Action1_107	0.187	0.827	0.288	0.201	0.22	

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

- **Show/Hide:** Displays or 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 in which you can set the color and scale of the measurement you selected, and choose whether you want to view measurement trends for all the measurements in the graph.



- **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.
- **Auto Correlate:** Opens a dialog box enabling you to correlate the selected measurement with other monitor measurements in order to view similar measurement trends in the scenario. For more information on the auto correlation feature, see page 43.
- **Sort by this column:** Sorts the measurements according to the selected column, in ascending or descending order.
- **Configure columns:** Opens the Legend Columns Options dialog box in which you can choose the columns you want to view, the width of each column, and the way you want to sort the columns.



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

Viewing the Data as a Spreadsheet and as Raw Data

The Analysis allows you to view graph data in two ways:

- ▶ **Spreadsheet:** The graph values displayed in the Graph Data tab.
- ▶ **Raw Data:** The actual raw data collected during the scenario, displayed in the Raw Data tab.

Spreadsheet View

You can view the graph displayed by the Analysis in spreadsheet format using the **Graph Data** tab below the graph.

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:** You can copy the spreadsheet to the clipboard in order to paste it into an external spreadsheet program.
- ▶ **Save As:** You can save the spreadsheet data to an Excel file. Once you have the data in Excel, you can generate your own customized graphs.

Raw Data View

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.

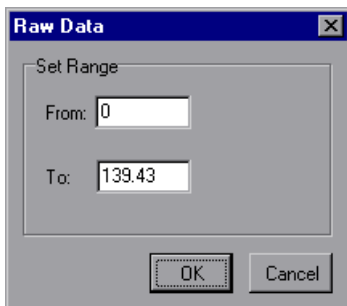
Viewing the raw data can be especially useful for:

- ▶ determining specific details about a peak—for example, which Vuser was running the transaction that caused the peak value(s).
- ▶ performing 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 the **Raw Data** button. The Raw Data dialog box opens.



- 2** Specify a time range—the entire graph (default) or a specific range of time—and click **OK**.

- 3 Click the **Raw Data** tab below the graph. The Analysis displays the raw data in a grid directly below the active graph.

Legend	Graph Details	User Notes	Graph Data	Raw Data	
	Transaction Name	Transaction Response Time	Scenario Elapsed Time	Transaction End Status Name	UserID
	Dimension Desktops	0.25	18	Pass	Vuser1
	Pentium_II_450_MHz	2.274	15	Pass	Vuser1
	Action1_Transaction	6.5	13	Pass	Vuser1
	dell_demo	1.502	18	Pass	Vuser1
	Action1_Transaction	4.106	17	Pass	Vuser1
	dell_demo	0.17	13	Pass	Vuser1

- 4 To show a different range, repeat the above procedure.

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.

The 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.

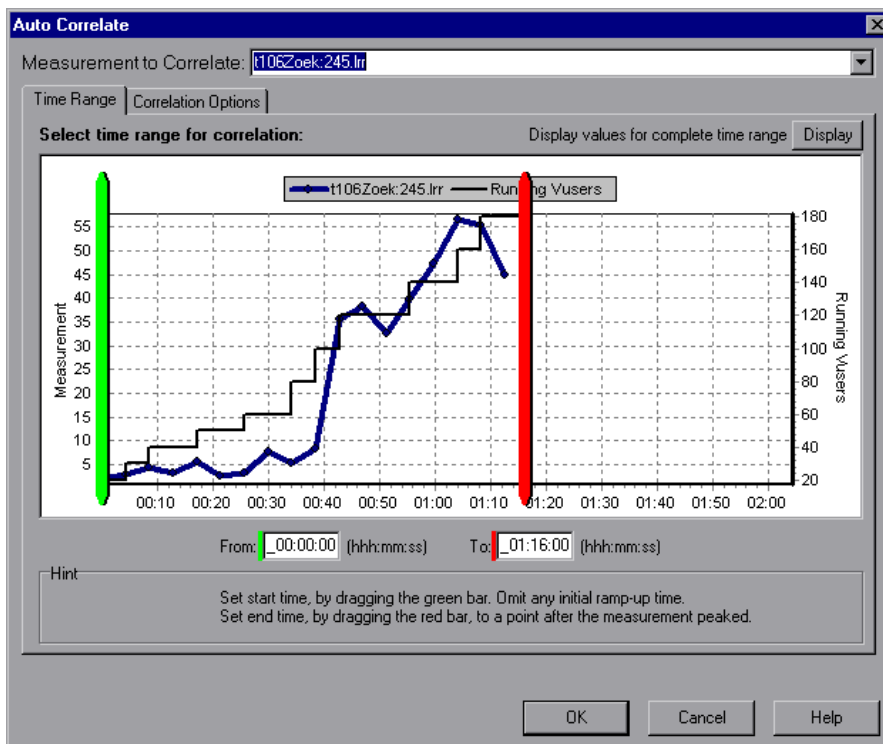
Note: If you standardize two line graphs, the two y-axes merge into one y-axis.

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.

To correlate a measurement in a graph with other measurements:

- 1 Right-click the measurement you want to correlate in the graph or legend, and select **Auto Correlate**. The Auto Correlate dialog box opens.



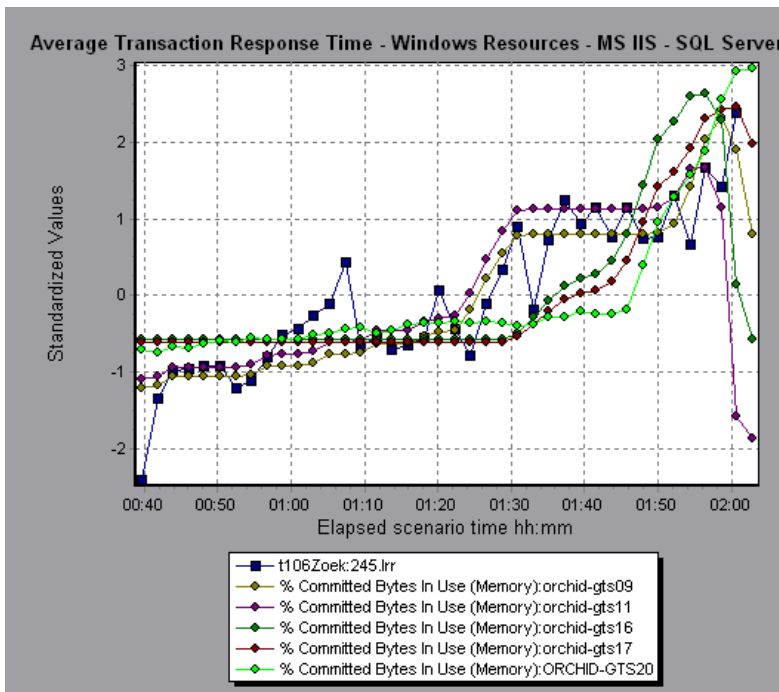
- 2 In the Time Range tab, specify a start and an end value (in hhh:mm:ss format) for the desired scenario time range. Alternatively, you can use the green and red vertical drag bars to specify the start and end values for the scenario time range.

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.

- 3** To specify the graphs you want to correlate with a selected measurement and the type of graph output to be displayed, click the **Correlation Options** tab.
- 4** In the select Graphs for Correlation section, choose the graphs whose measurements you want to correlate with your selected measurement.
- 5** In the Data Interval section, select one of the following two options:
 - **Automatic:** Instructs the Analysis to use an automatic value, determined by the time range, in order to calculate the interval between correlation measurement polls.
 - **Correlate data based on X second intervals:** Enter the number of seconds you want the Analysis to wait between correlation measurement polls.
- 6** In the Output section, select one of the following two options:
 - **Show the X most closely correlated measurements:** Specify the number of most closely correlated measurements you want the Analysis to display.
 - **Show measurements with an influence factor of at least X%:** Specify the minimum influence factor for the measurements displayed in the correlated graph.
- 7** Click **OK**. The 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.

In the following example, the t106Zoek:245.Irr 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.Irr are displayed in the graph below.



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

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

For more information on auto correlation, see Chapter 21, “Interpreting Analysis Graphs.”

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

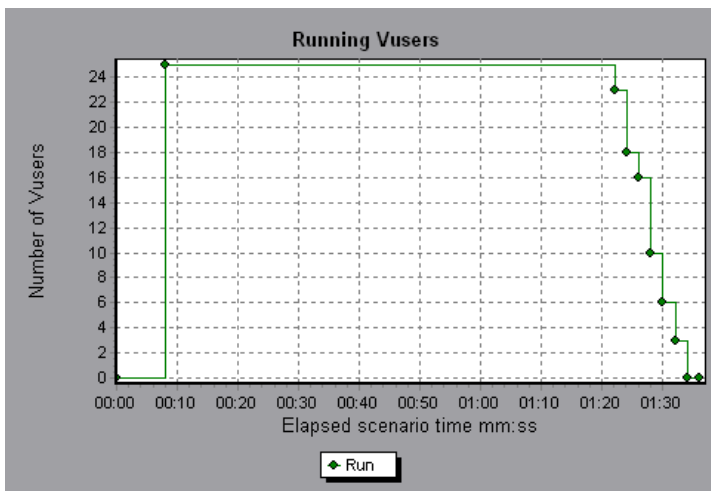
About Vuser Graphs

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."

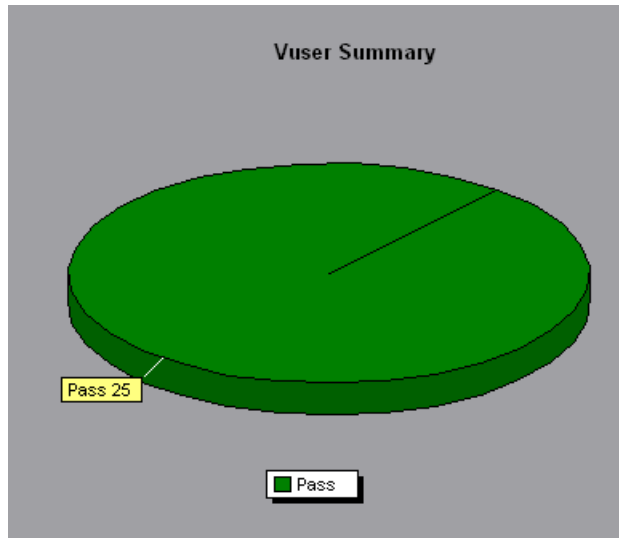
The x-axis represents the elapsed time from the beginning of the scenario run. The y-axis represents the number of Vusers in the scenario.



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.

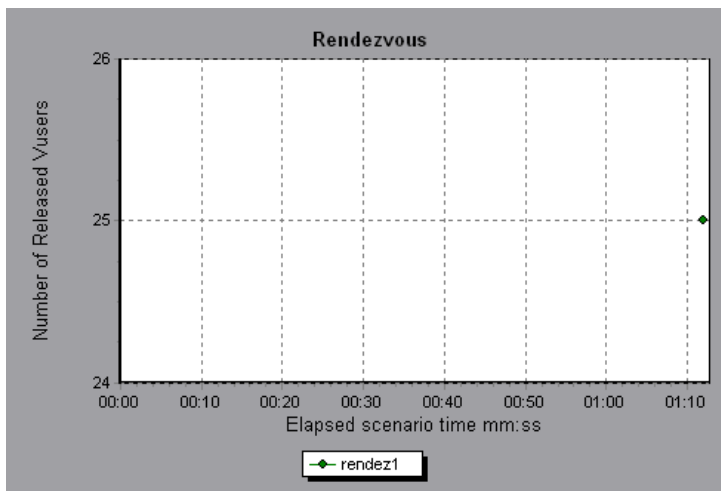


Rendezvous Graph

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

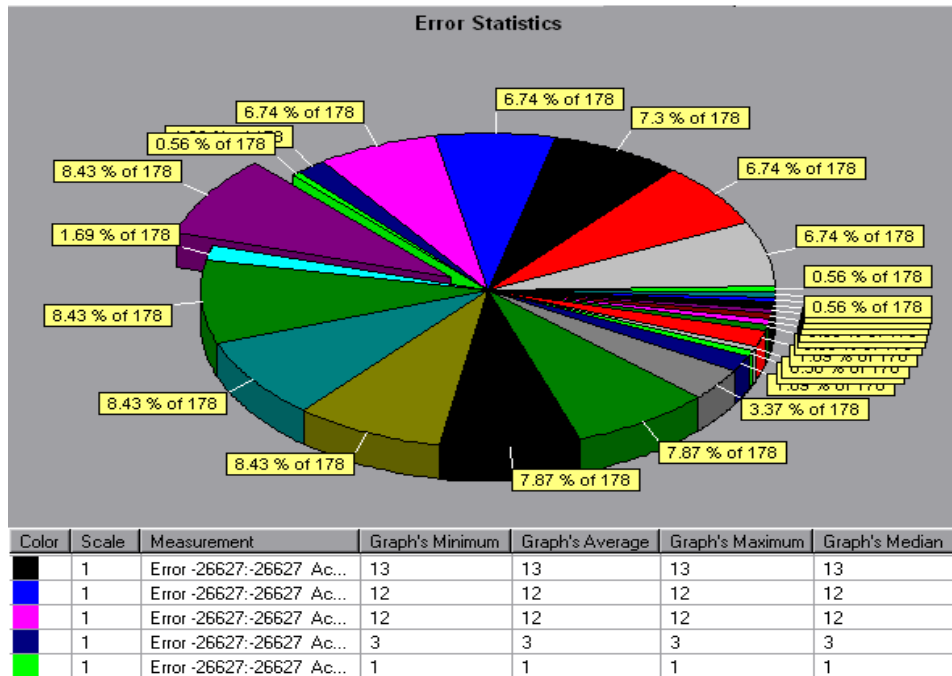
During scenario execution, Vusers may not complete all transactions successfully. The Error graphs let you view information about the transactions that failed, stopped, or ended in errors. Using the Error graphs, you can view a summary of errors that occurred during the scenario and the average number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

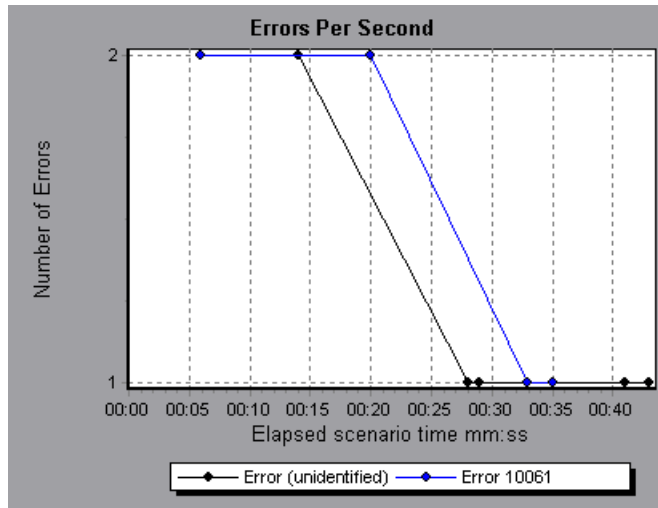
This graph may only be viewed as a pie.



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

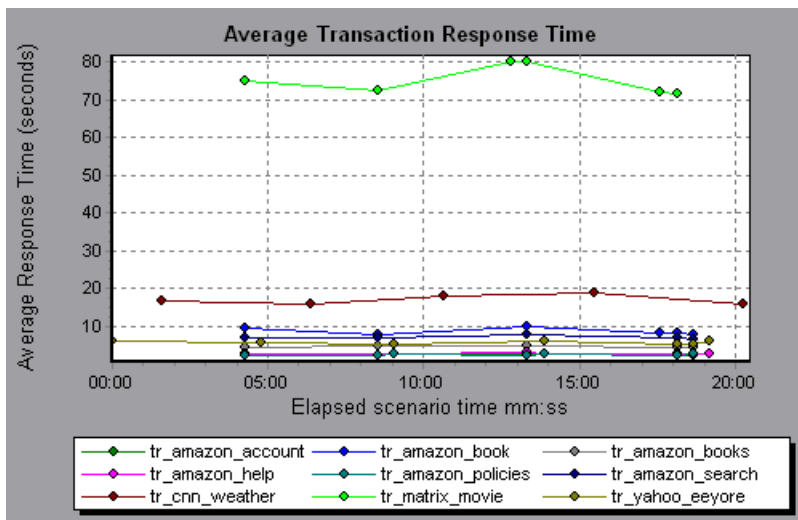
During scenario execution, Vusers generate data as they perform transactions. The Analysis enables you to generate graphs that show the transaction performance and status throughout script execution.

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 the 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.

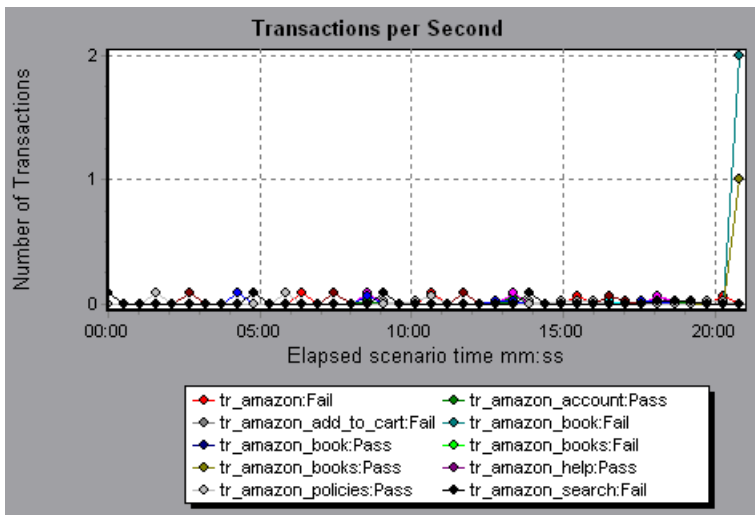
For example, if the Average Transaction Response Time graph shows that performance time gradually improved, you can compare it to the Running Vusers graph to see whether the performance time improved due to a decrease in the Vuser load.

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.

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.

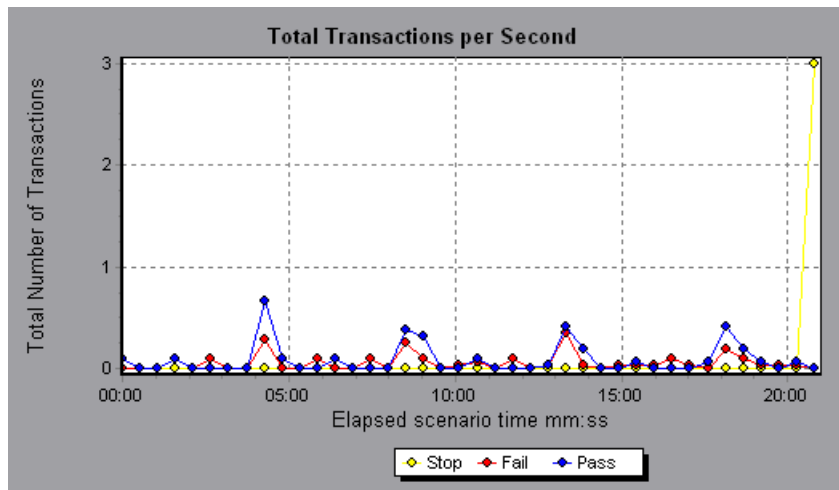
The x-axis represents the elapsed time from the beginning of the scenario run. The y-axis represents the number of transactions performed during the scenario run.



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.

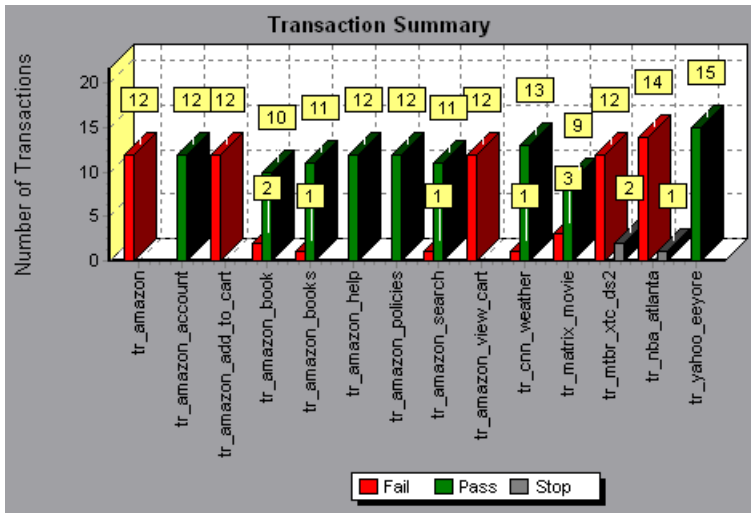
The x-axis represents the elapsed time (in seconds) since the start of the scenario run. The y-axis represents the total number of transactions performed during the scenario run.



Transaction Summary Graph

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

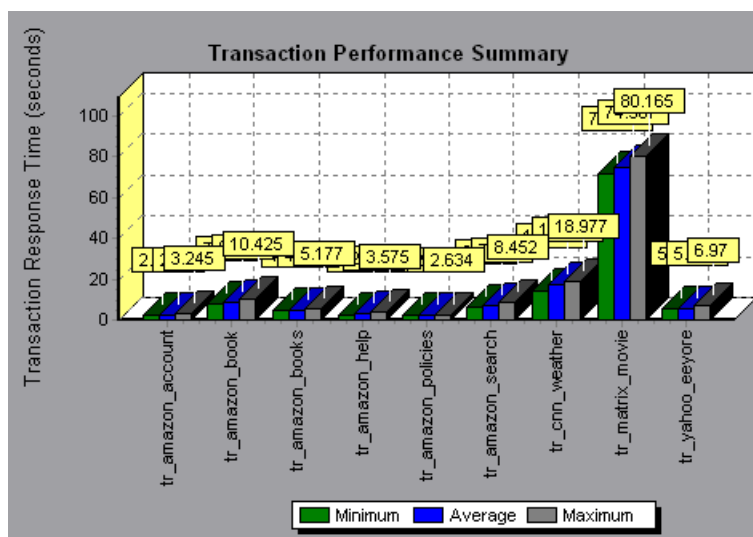
The x-axis specifies the name of the transaction. The y-axis shows the number of transactions performed during the scenario run.



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.



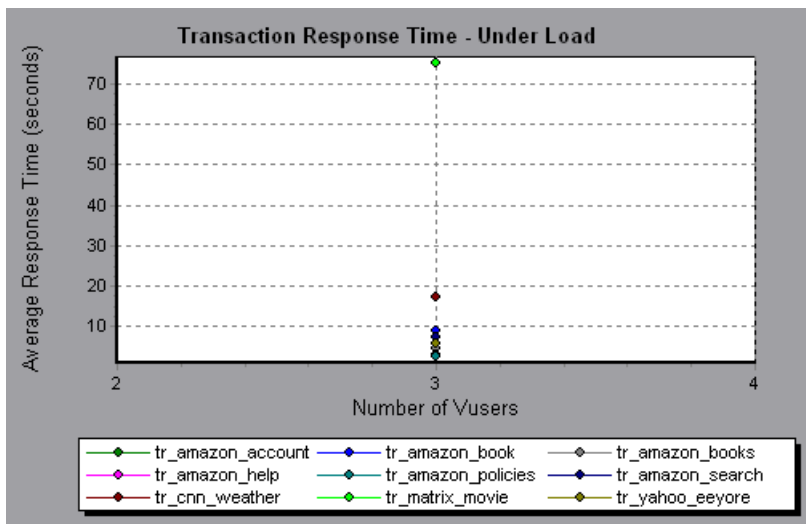
You can view a breakdown of a transaction in the Transaction Performance Summary graph by selecting **View > Show Transaction Breakdown Tree**, or right-clicking the transaction and selecting **Show Transaction Breakdown Tree**. In the Transaction Breakdown Tree, right-click the transaction you want 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.



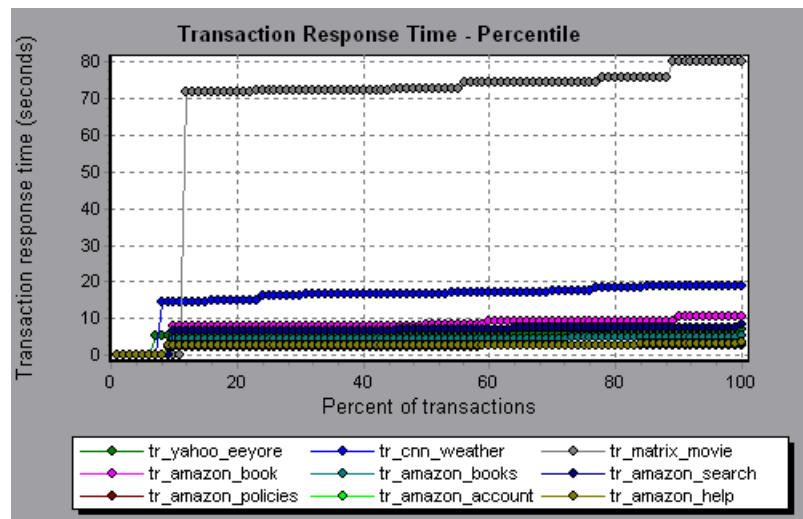
Transaction Response Time (Percentile) Graph

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: The 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.



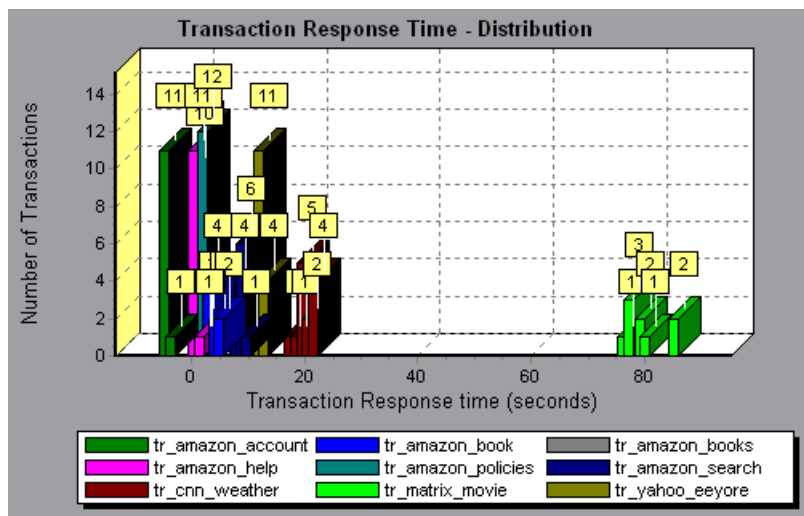
It is recommended to compare the Percentile graph to a graph indicating average response time such as the Average Transaction Response Time graph. A high response time for several transactions may raise the overall average. However, if the transactions with a high response time occurred less than five 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. 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:

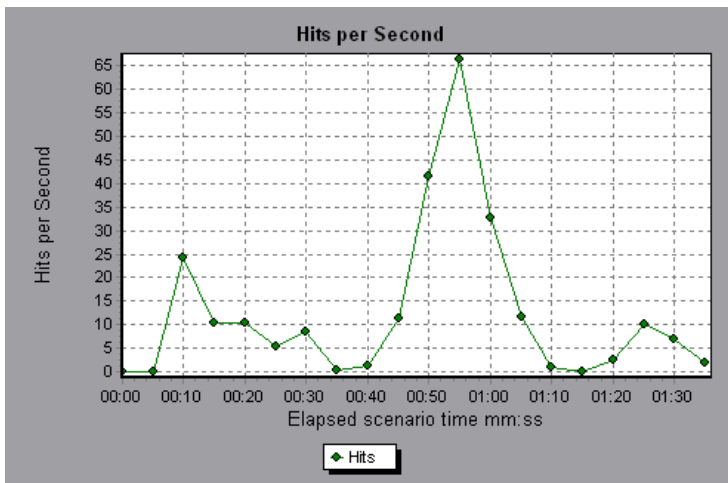
- Hits per Second Graph
- Throughput Graph
- HTTP Status Code Summary Graph
- HTTP Responses per Second Graph
- Pages Downloaded per Second Graph
- Retries per Second Graph
- Retries Summary Graph

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 that occurred during the scenario, the number of HTTP responses per second, the 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, the number of downloaded pages per second, the number of server retries per second, and a summary of the server retries during the scenario.

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.

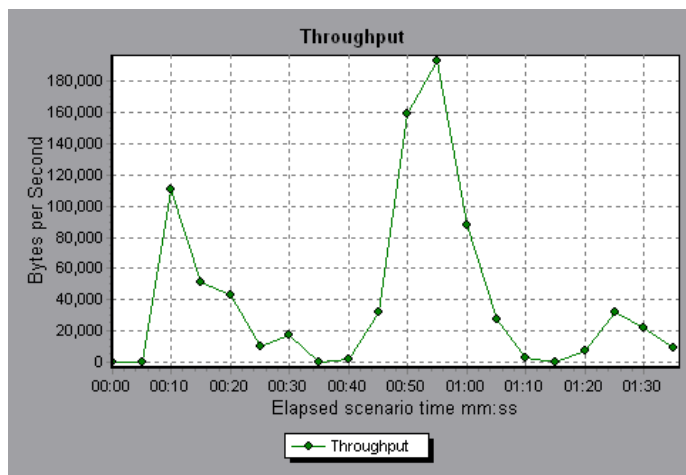
Note: You cannot change the granularity of the x-axis to a value that is less than the Web granularity you defined in the General tab of the Options dialog box.

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.



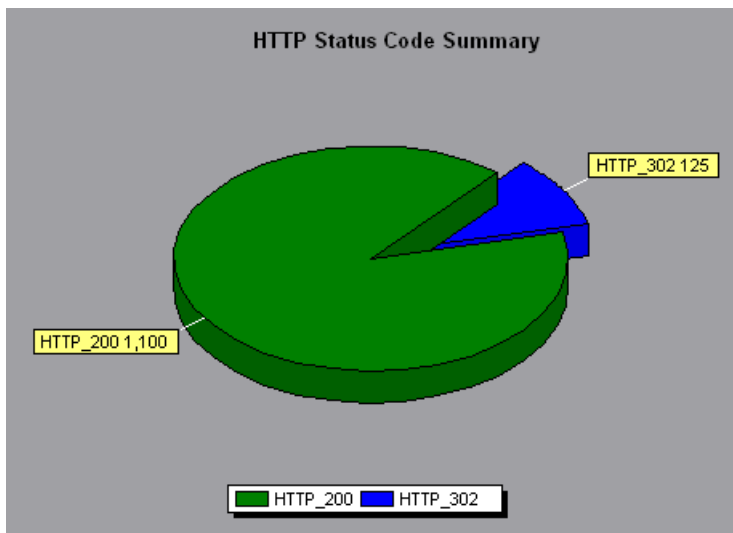
Note: You cannot change the granularity of the x-axis to a value that is less than the Web granularity you defined in the General tab of the Options dialog box.

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 may 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.

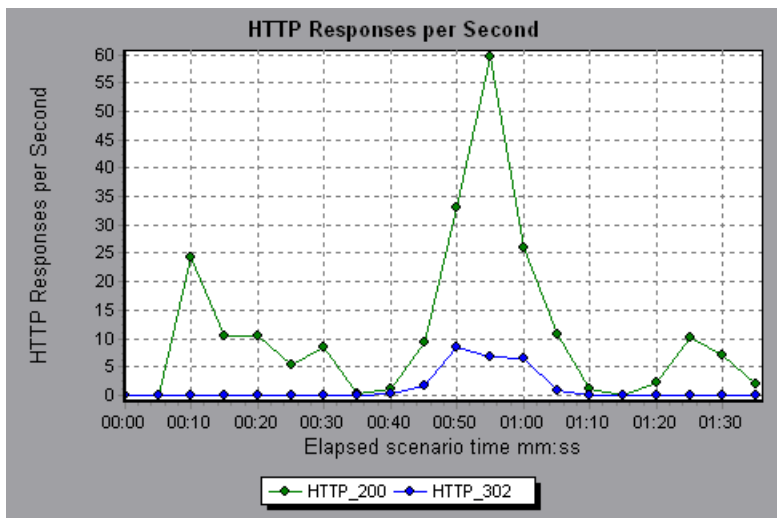


HTTP Responses per Second Graph

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

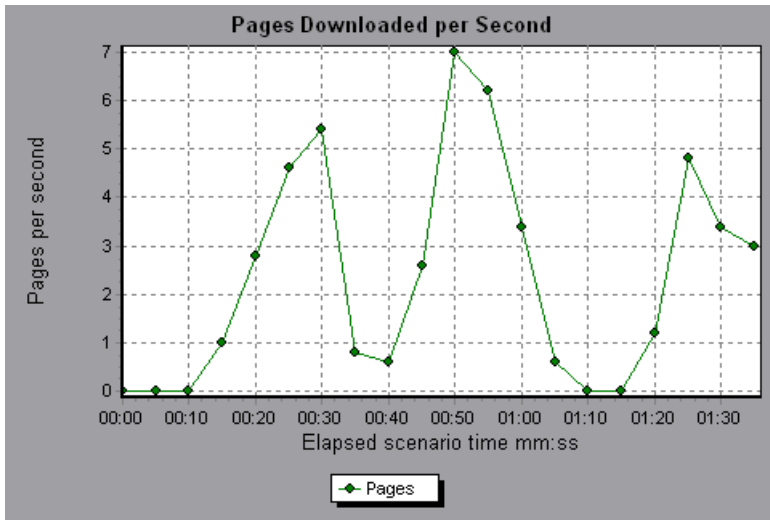
Code	Description
410	Gone
411	Length Required
412	Precondition Failed
413	Request Entity Too Large
414	Request - URI Too Large
415	Unsupported Media Type
416	Requested range not satisfiable
417	Expectation Failed
500	Internal Server Error
501	Not Implemented
502	Bad Gateway
503	Service Unavailable
504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

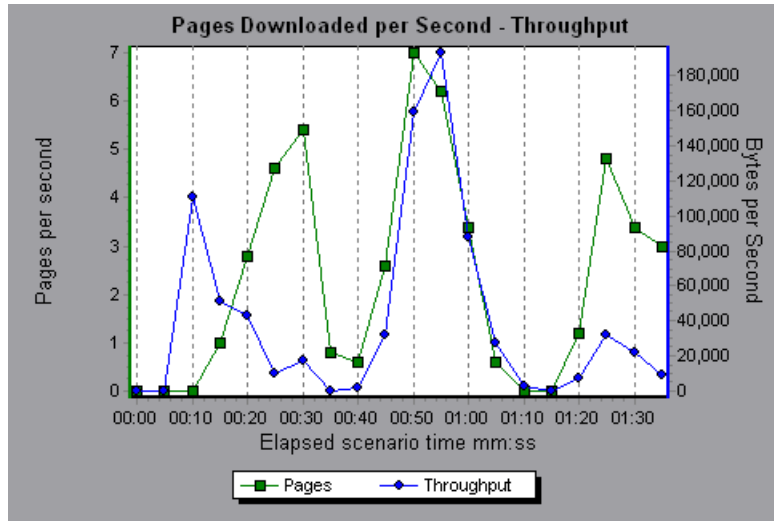
The Pages Downloaded per Second graph shows the number of Web pages (y-axis) downloaded from the server during each second of the scenario run (x-axis). This graph helps you evaluate the amount of load Vusers generate, in terms of the number of pages downloaded.

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.

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.

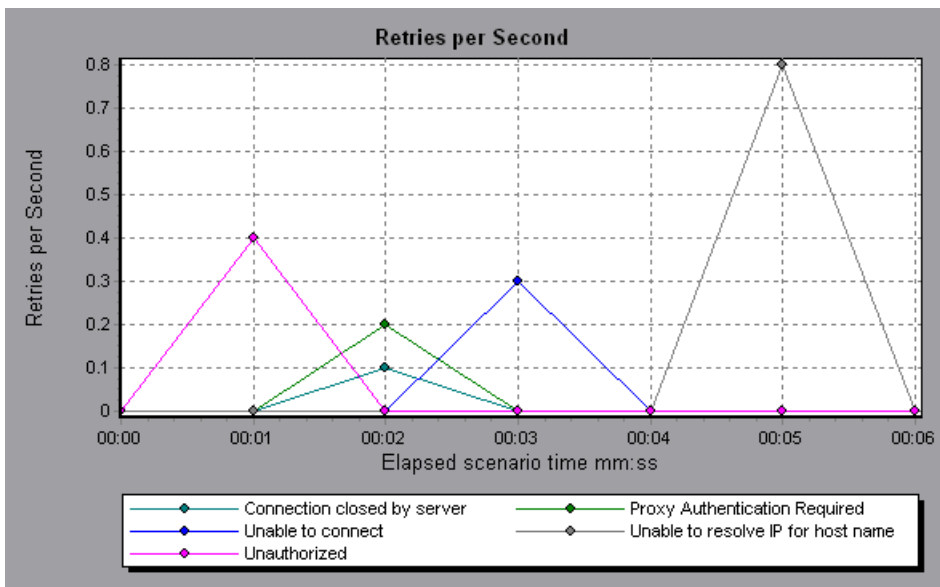


Retries per Second Graph

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.

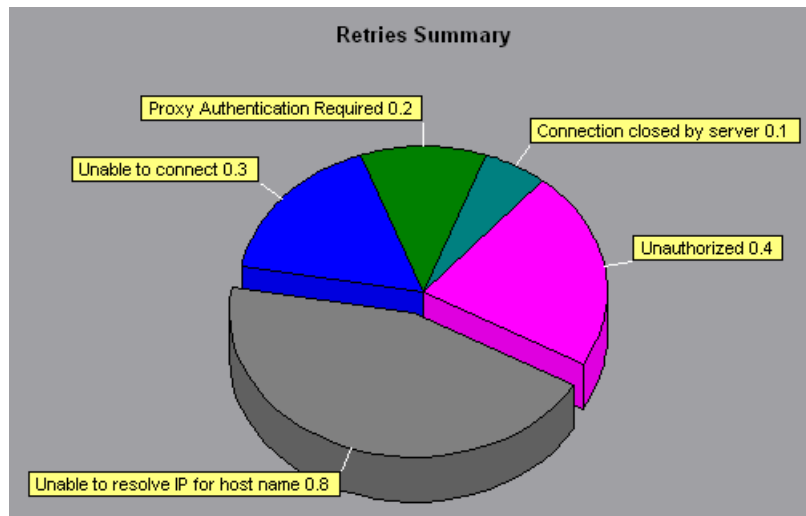


Retries Summary Graph

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.



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. The 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.

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

To enable the component breakdown feature:

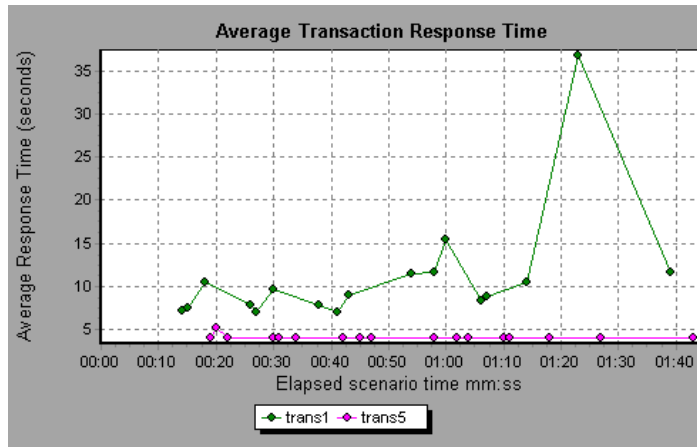
- 1** From the Controller menu, choose **Tools > Options**.
- 2** Select the **Web Page Breakdown** tab. Select the **Enable Web Page Breakdown** check box.
- 3** From the VuGen menu, choose **Vuser > Run-Time Settings**. Select the **General** tab, and select the **Define each step as a transaction** 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 *LoadRunner Creating Vuser Scripts User's 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.
- 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 four 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.

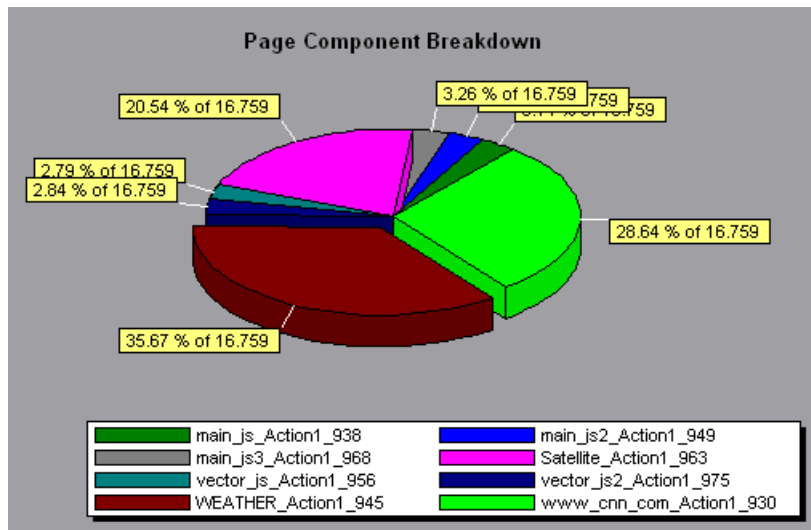
See the following sections for an explanation of each of these four graphs.



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.

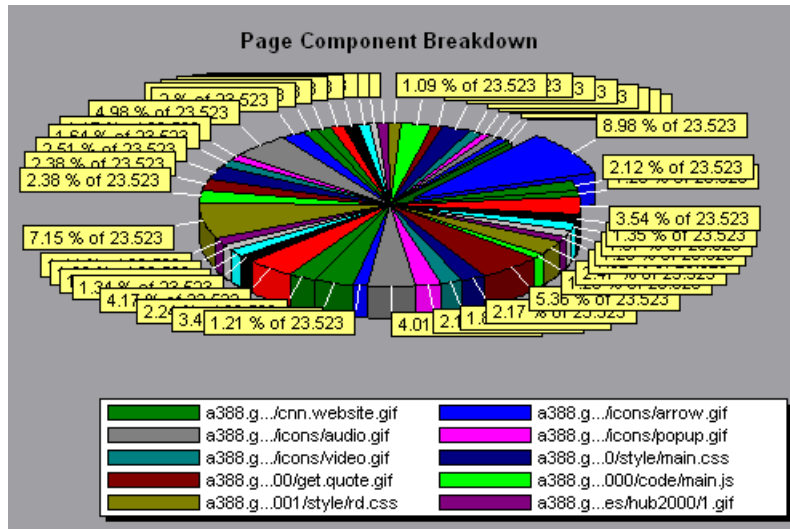
Page Component Breakdown Graph

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



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.

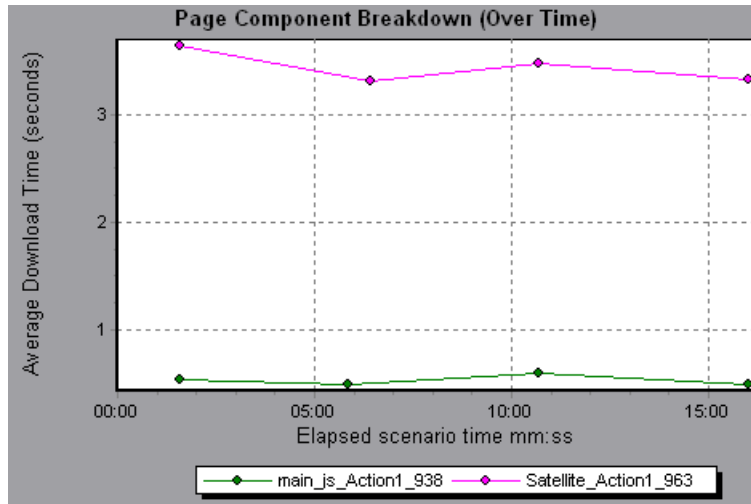


The above graph shows that the main cnn.com/WEATHER component took the longest time to download (8.98% of the total download time). To isolate other problematic components of the URL, it may be helpful to sort the legend according to the average number of seconds taken to download a component. To sort the legend by average, click the **Graph's Average** column heading.

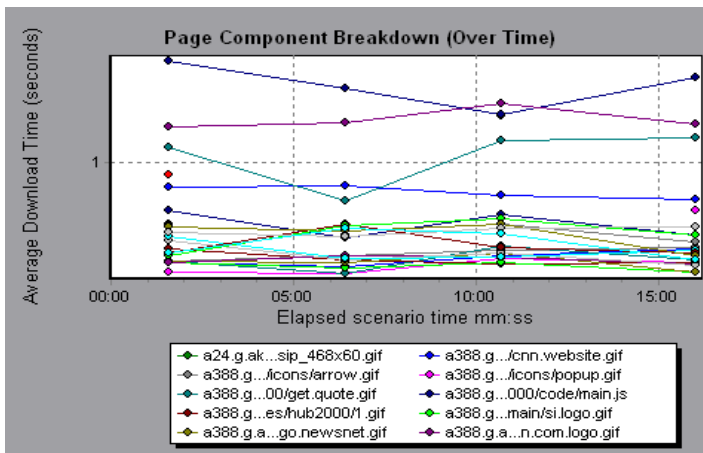
Note: The Page Component Breakdown graph can only be viewed as a pie.

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. 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.

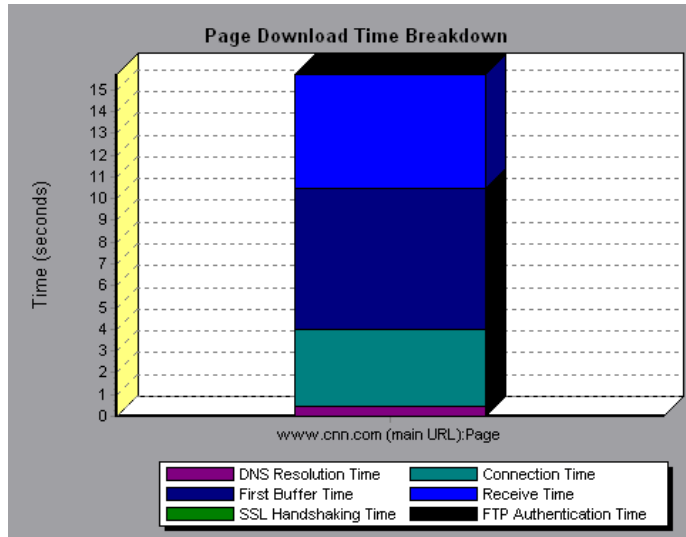


Using the above graph, you can track which components of the main component were most problematic, and at which point(s) during the scenario the problem(s) occurred. To isolate the most problematic components, it may be helpful to sort the legend tab according to the average number of seconds taken to download a component. To sort the legend by average, double-click the **Average** column heading.

To identify a component in the graph, you can click 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:

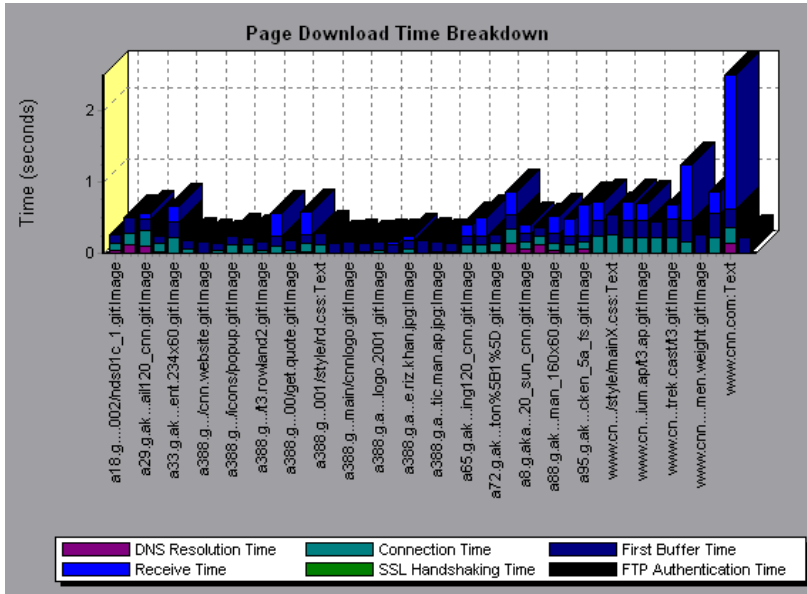
Name	Description
DNS Resolution	Displays the amount of time needed to resolve the DNS name to an IP address, using the closest DNS server. The DNS Lookup measurement is a good indicator of problems in DNS resolution, or problems with the DNS server.
Connection	Displays the amount of time needed to establish 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	<p>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.</p> <p>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.</p>
SSL Handshaking	<p>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.</p> <p>The SSL Handshaking measurement is only applicable for HTTPS communications.</p>

Name	Description
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).
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](http://www.cnn.com) URL. If you break the [cnn.com](http://www.cnn.com) URL down further, you can

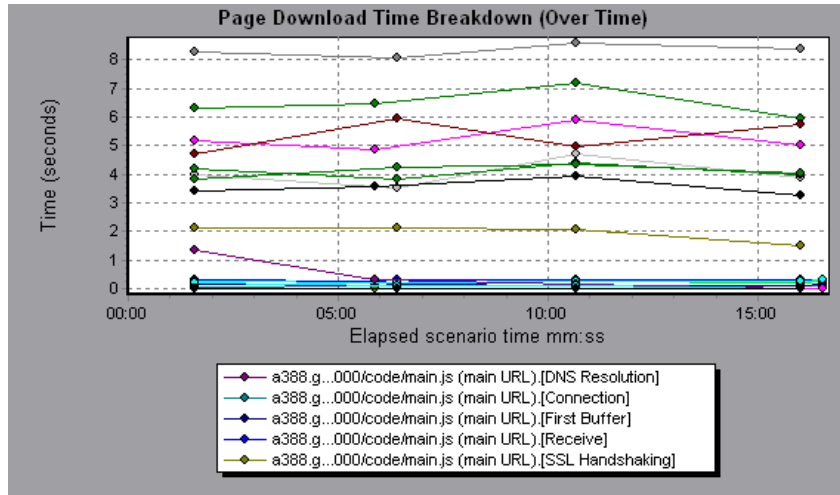
isolate the components with the longest download time, and analyze the network or server problems that contributed to the delay in response time.



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. This graph enables you to determine at what point during scenario execution network or server problems occurred.



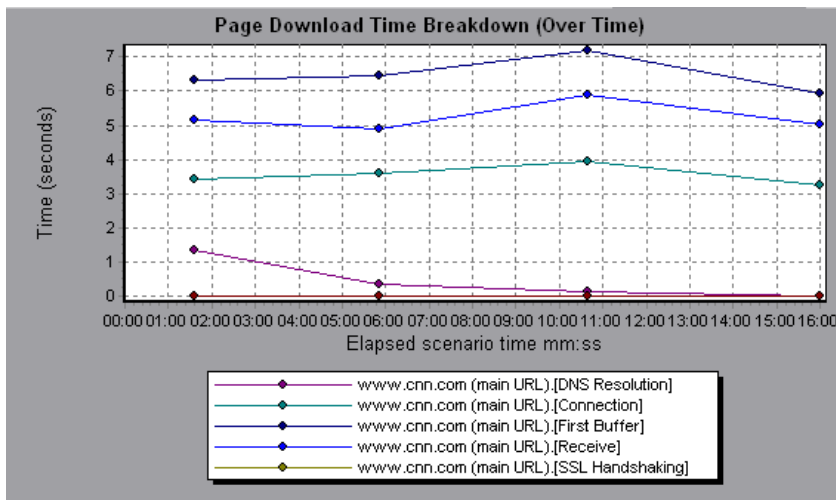
Note: Each measurement displayed on the page level is the sum of that measurement recorded for each page component. For example, the Connection Time for www.cnn.com is the sum of the Connection Time for each of the page's components.

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, click 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

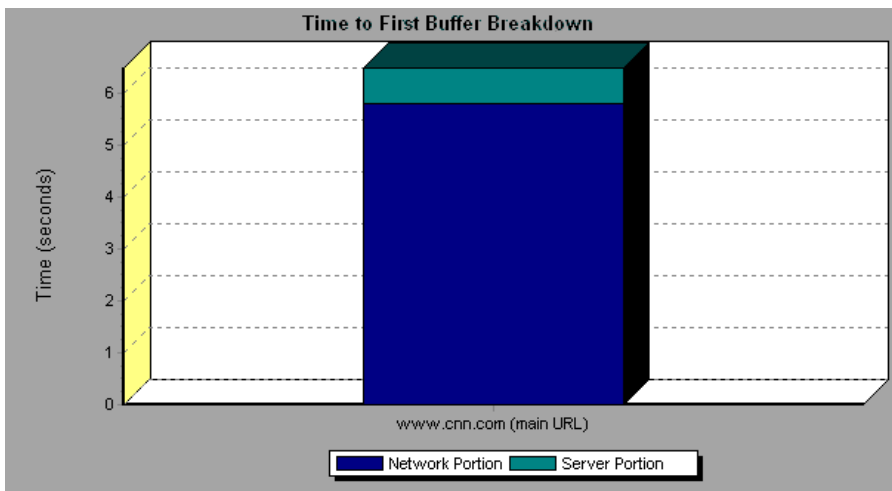
Page Download Time Breakdown (Over Time) graph demonstrates that first buffer and receive time remained high throughout the scenario, and that DNS Resolution time decreased during the scenario.



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.



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.

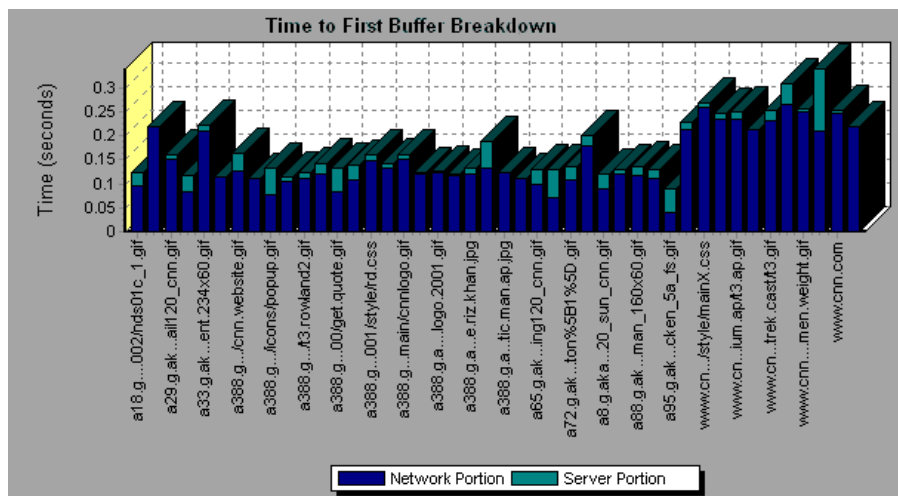
Server time is defined as the average amount of time that passes from the receipt of ACK of the initial HTTP request (usually GET) until the first buffer is successfully received back from the Web server.

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.

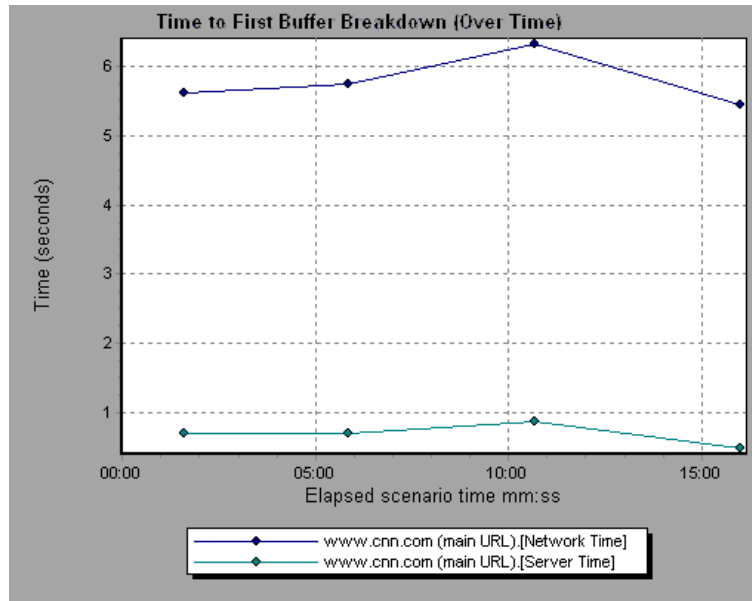
You can break the main cnn.com URL down further to view the time to first buffer breakdown for each of its components.



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.



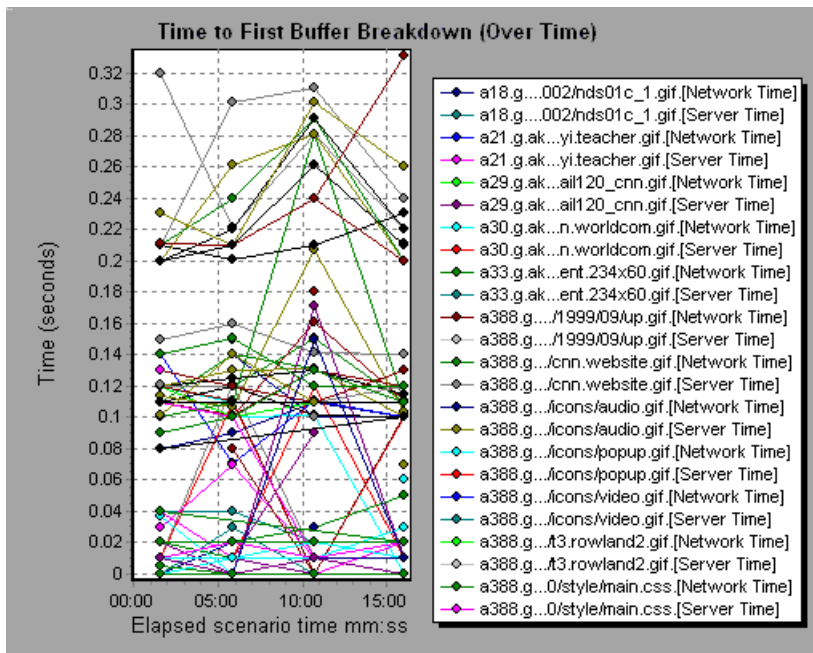
Network time is defined as the average amount of time that passes from the moment the first HTTP request is sent until receipt of ACK.

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.

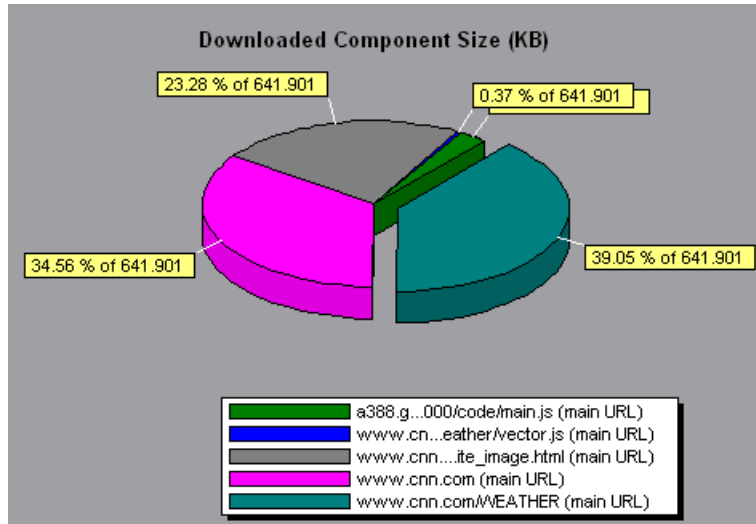
You can break the main cnn.com URL down further to view the time to first buffer breakdown for each of its components.



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.

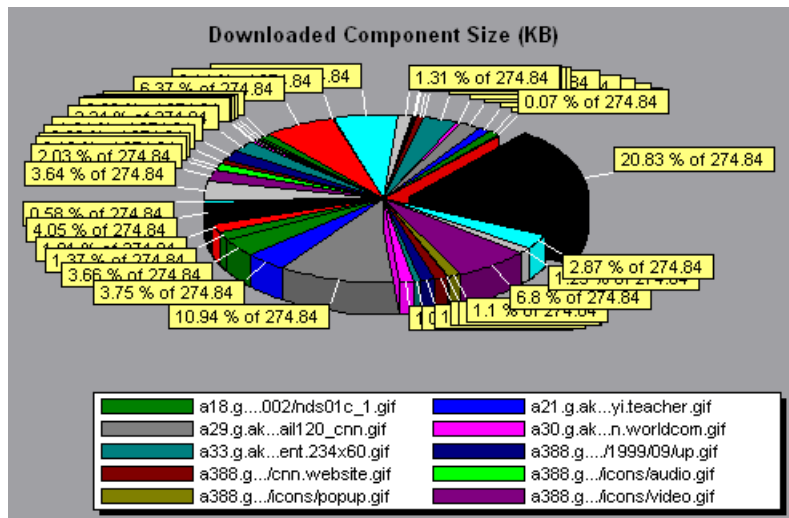
Downloaded Component Size Graph

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;
}
```

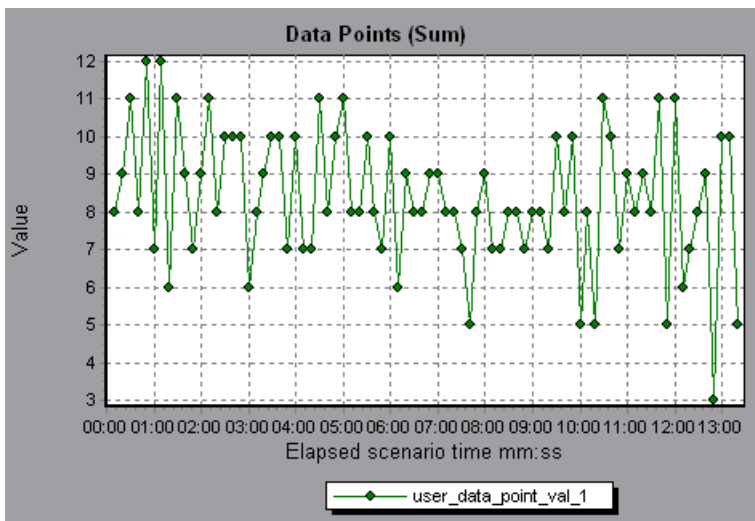
For Vuser protocols that support the graphical script representations such as Web and Oracle NCA, you insert a data point as a User Defined step. Data point information is gathered each time the script executes the function or step. For more information about data points, see the online *LoadRunner Function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

Data Points (Sum) Graph

The Data Points (Sum) graph shows the sum values 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 sum values of the recorded data point statements.

In the following example, the *user_data_point_val_1* data point is shown as a function of the elapsed scenario time.

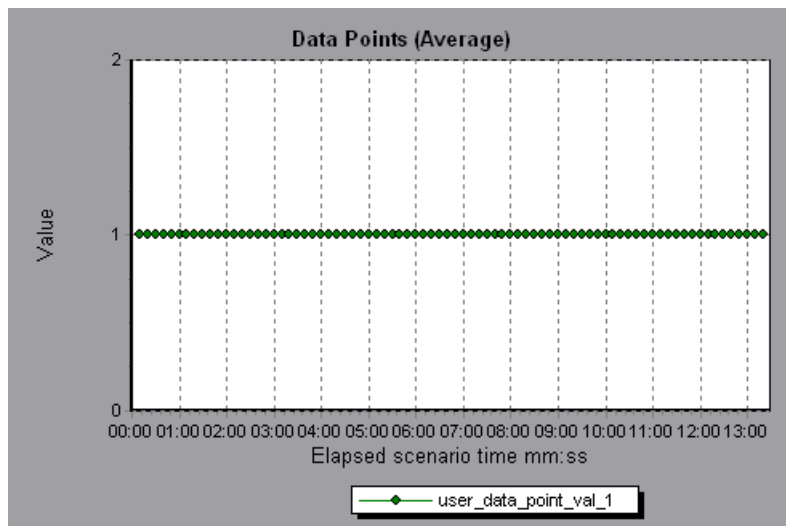


Data Points (Average) Graph

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.

In the following example, the *user_data_point_val_1* data point 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
- TUXEDO Resources Graph

About System Resource Graphs

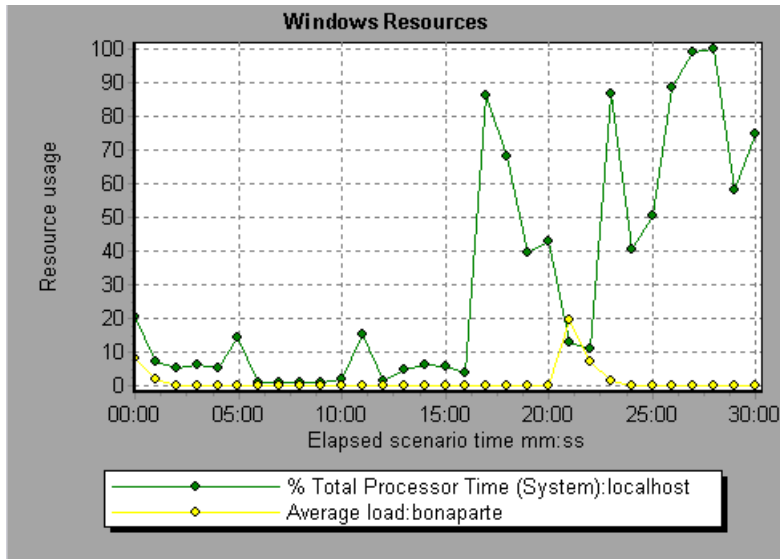
System Resource graphs display the system resource usage measured by the online monitors during the scenario run. These graphs require that you specify the resources you want to measure *before* running the scenario. For more information, see the section on online monitors in the *LoadRunner Controller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

System Resource graphs are available for:

- Windows Resources
- UNIX Resources
- SNMP Resources
- TUXEDO Resources

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.



The following default measurements are available for Windows Resources:

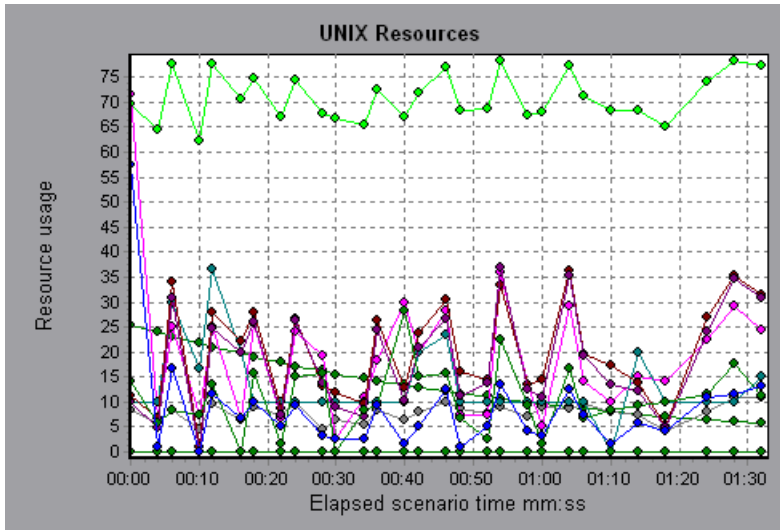
Object	Measurement	Description
System	% Total Processor Time	The average percentage of time that all the processors on the system are busy executing non-idle threads. On a multi-processor system, if all processors are always busy, this is 100%, if all processors are 50% busy this is 50% and if 1/4th 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.



The following default measurements are available for UNIX machines:

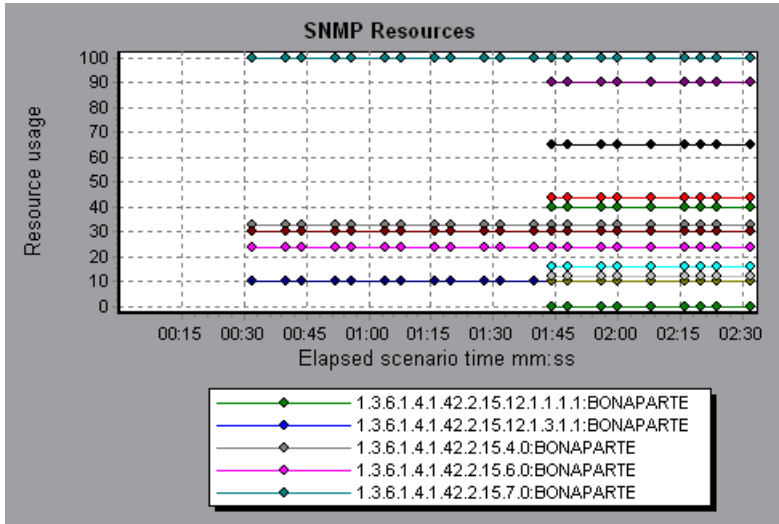
Measurement	Description
Average load	Average number of processes simultaneously in 'Ready' state during the last minute
Collision rate	Collisions per second detected on the Ethernet
Context switches rate	Number of switches between processes or threads, per second
CPU utilization	Percent of time that the CPU is utilized
Disk rate	Rate of disk transfers

Measurement	Description
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

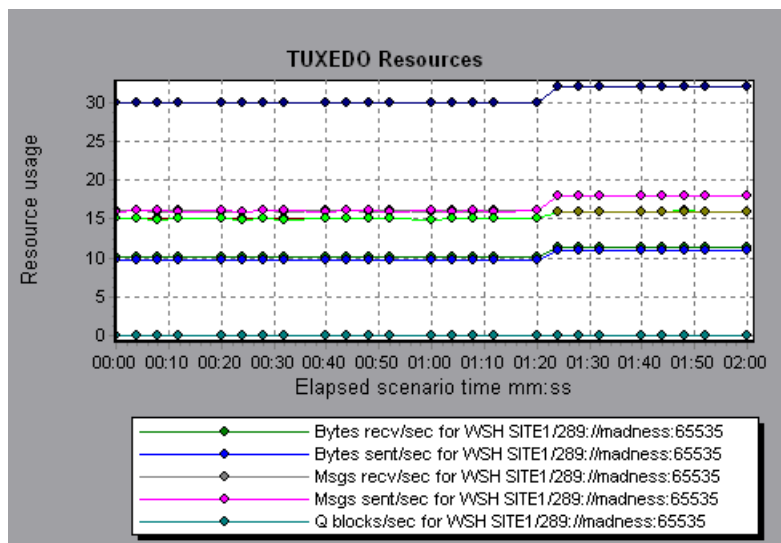
The SNMP Resources graph shows statistics for machines running an SNMP agent, using the Simple Network Management Protocol (SNMP).

The following graph displays SNMP measurements for a machine called *bonaparte*:



TUXEDO Resources Graph

The TUXEDO Resources graph provides information about the server, load generator machine, workstation handler, and queue in a TUXEDO system.



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

Monitor	Measurements
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.
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.

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

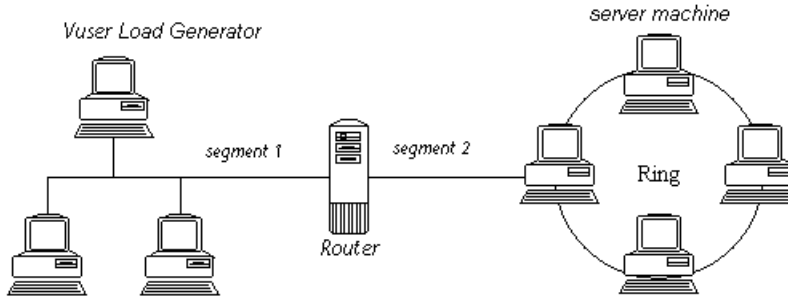
Network configuration is a primary factor in the performance of applications and Web systems. A poorly designed network can slow client activity to unacceptable levels.

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.

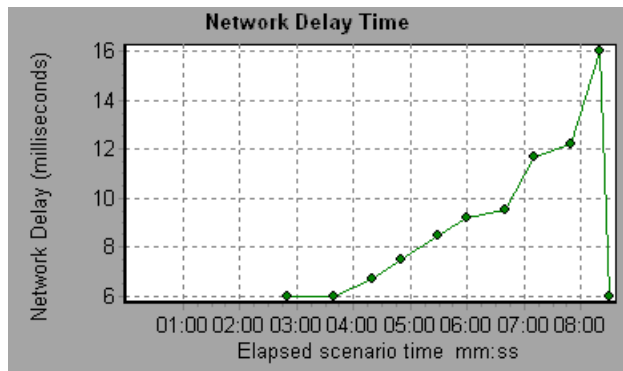
In order for the Analysis to generate Network monitor graphs, you must activate the Network monitor before executing the scenario. In the Network monitor settings, you specify the path you want to monitor. For information about setting up the Network monitor, see the *LoadRunner Controller User's Guide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

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.

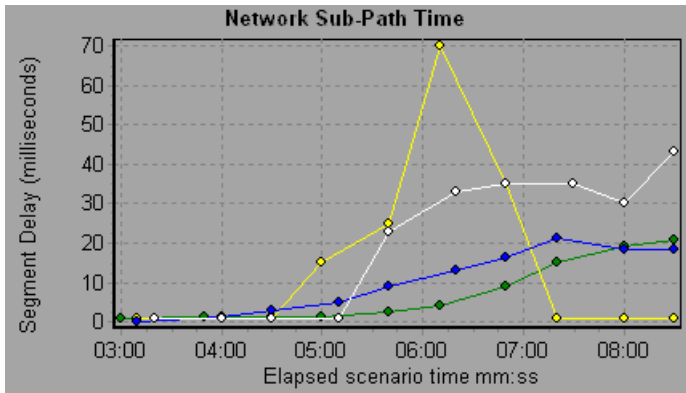
The following graph shows the network delay as a function of the elapsed scenario time. It shows a delay of 16 milliseconds in the 8th minute of the scenario.



Network Sub-Path Time Graph

The Network Sub-Path Time graph 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 milliseconds in the sixth minute.

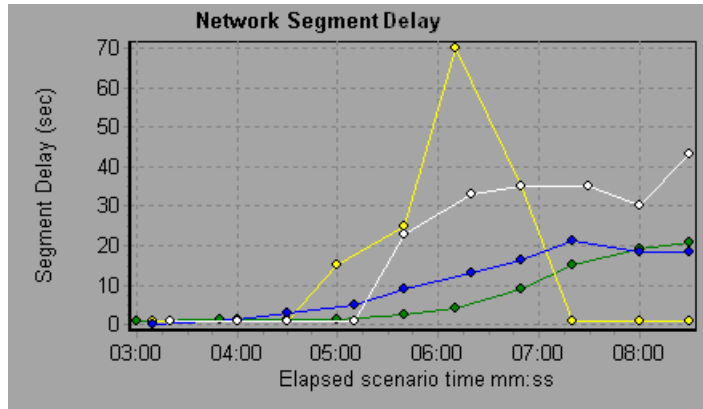


Note: The delays from the source machine to each of the nodes are measured concurrently, yet independently. It is therefore possible that the delay from the source machine to one of the nodes could be greater than the delay for the complete path between the source and destination machines.

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.

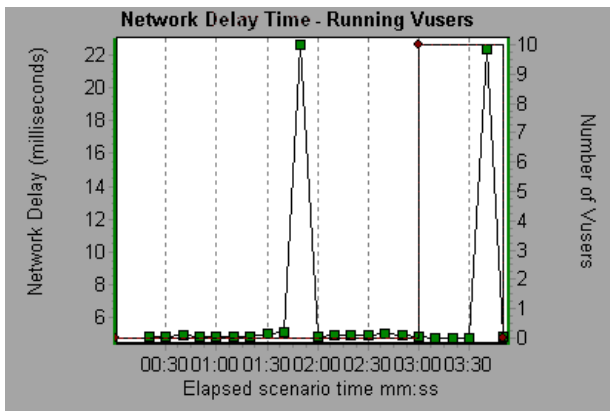


Note: The segment delays are measured approximately, and do not add up to the network path delay which is measured exactly. The delay for each segment of the path is estimated by calculating the delay from the source machine to one node and subtracting the delay from the source machine to another node. 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:

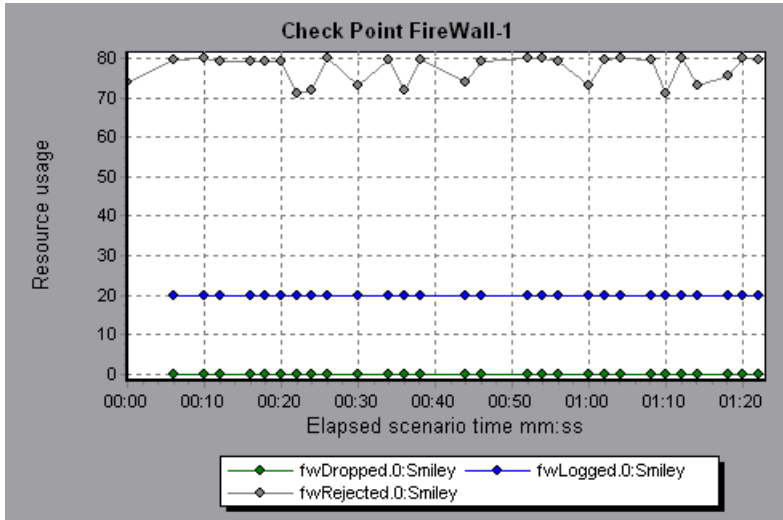
- ▶ Check Point FireWall-1 Server Graph

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 (Windows)*.

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.



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.

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, and iPlanet/Netscape server performance.

This chapter describes:

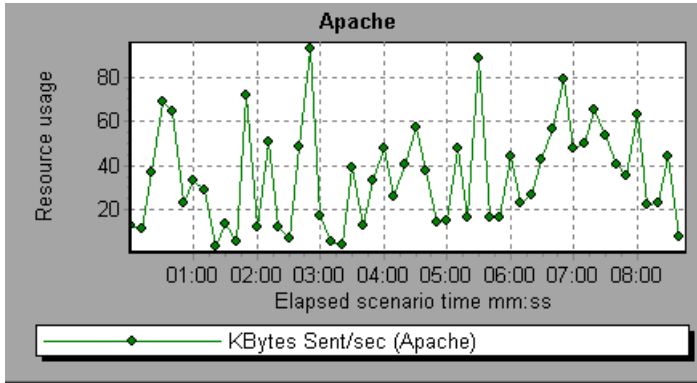
- ▶ Apache Server Graph
- ▶ Microsoft Information Internet Server (IIS) Graph
- ▶ iPlanet/Netscape Server Graph

About Web Server Resource Graphs

Web Server Resource graphs provide you with information about the resource usage of the Apache, Microsoft IIS, and iPlanet/Netscape 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 (Windows)*.

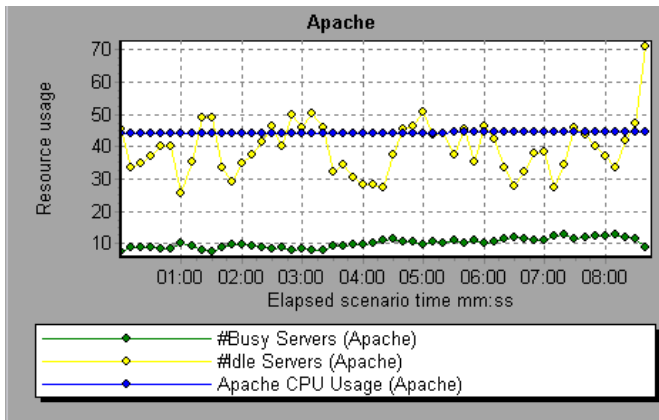
In order to display all the measurements on a single graph, the Analysis may scale them. The Legend tab indicates the scale factor for each resource. To obtain the true value, multiply the scale factor by the displayed value. For example, in the following graph the actual value of *KBytes Sent per second* 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.



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.

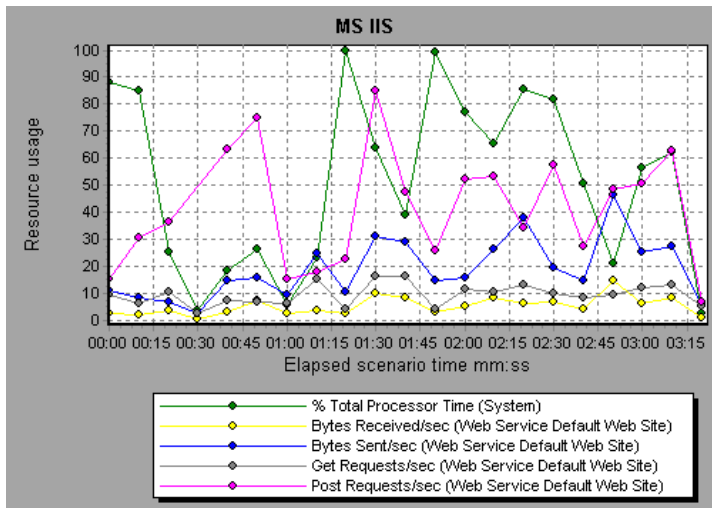
The following default measurements are available for the Apache server:

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.



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.

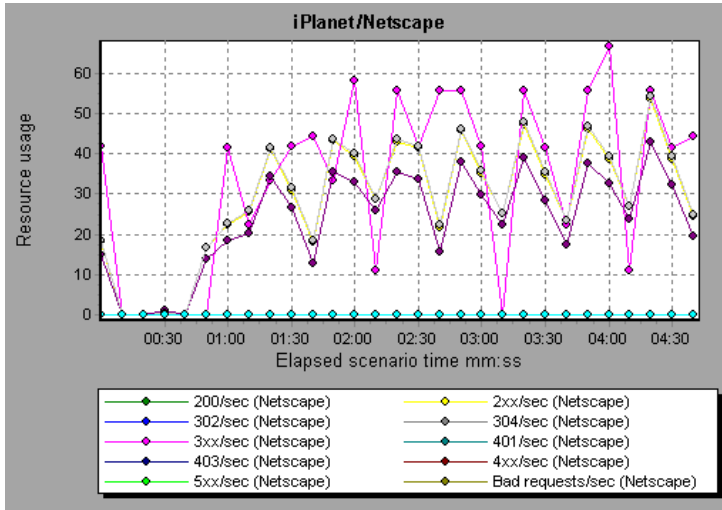
Note that the scale factor for the *Bytes Sent/sec* and *Bytes Received/sec* measurements is 1/100, and the scale factor for the *Post Requests/sec* measurement is 10.

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.



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.

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

Measurement	Description
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 not return an HTTP status code

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
- Brokat Twister Graph
- ColdFusion Graph
- Fujitsu INTERSTAGE Graph
- Microsoft Active Server Pages (ASP) Graph
- Oracle9iAS HTTP Server Graph
- SilverStream Graph
- WebLogic (SNMP) Graph
- WebLogic (JMX) Graph
- WebSphere Graph

About Web Application Server Resource Graphs

Web Application Server Resource graphs provide you with resource usage information about the Ariba, ATG Dynamo, BroadVision, Brokat Twister, ColdFusion, Fujitsu INTERSTAGE, Microsoft ASP, Oracle9iAS HTTP, SilverStream, WebLogic (SNMP), WebLogic (JMX) Web 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 (Windows)*.

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, the 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 121.

Ariba Graph

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
Free Memory	The instantaneous reading of the reserved memory (in KB) 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

Measurement	Description
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

Measurement	Description
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 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
ImIsManager	True if the Dynamo is running a load manager
ImManagerIndex	Returns the Dynamo's offset into the list of load managing entities
ImIsPrimaryManager	True if the load manager is an acting primary manager

d3LoadManagement Measurement	Description
ImServicingCMs	True if the load manager has serviced any connection module requests in the amount of time set as the connection module polling interval
ImCMLDRPPort	The port of the connection module agent
ImIndex	A unique value for each managed entity
ImSNMPPort	The port for the entry's SNMP agent
ImProbability	The probability that the entry will be given a new session
ImNewSessions	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 ImNewSessionOverride.
ImNewSessionOverride	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

d3DRP Server Measurement	Description
drpPort	The port of the DRP server
drpTotalReqsServed	Total number of DRP requests serviced

d3DRPServer Measurement	Description
drpTotalReqTime	Total service time in msec for all DRP requests
drpAvgReqTime	Average service time in msec 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
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 monitor supplies performance statistics for all the servers/services available within the application server.

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.
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.

Server	Multiple Instances	Description
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
- Rebind
- Rsvl
- Unbind

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.
- ▶ **IdIQ**
- ▶ **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**
- **JSPERR**
- **RELEASE**
- **STOP**
- **SUCC**
- **SYNTAX**

Brokat Twister Graph

The following default measurements are available for the Brokat Twister server:

Measurement	Description
nam-boundObjTable	A table that contains information about the Naming Service's bound objects
nam-boundObjEntry	A row in the nam-boundObjTable table
nam-boundObjId	The ID is the unique logical name by which the object is known in the Brokat Twister environment.
nam-boundObjAvail	The object may be available (true) or unavailable (false). One possible reason for an object not being available is that the service is not running.
nam-boundObjHost	The machine on which this object is running
nam-boundObjPort	The port where this object can be accessed
nam-boundObjKey	ObjectKey is the server object name
nam-boundObjTypeId	Typeid is the object type as defined by Corba
rep-status	The status of the repository service
rep-ipAddress	The IP address of the repository service
rep-iiopPort	The IIOP port of the repository service
lic-status	The status of the license service
lic-ipAddress	The IP address of the license service
lic-iiopPort	The IIOP port of the license service
lic-flatLicenseTable	A table of flat licenses
lic-flatLicenseEntry	An entry in the flat licenses table
lic-flatLicenseName	The name of a flat license
lic-cpuBasedLicenseTable	A table of CPU-based licenses
lic-cpuBasedLicenseEntry	An entry in the CPU-based licenses table

Measurement	Description
lic-cpuBasedLicenseName	The name of a CPU-based license
lic-cpuBasedLicenseInUse	The usage of a CPU-based license
lic-PlatformTable	A table of platform entries
lic-PlatformEntry	An entry in the platform table
lic-platformName	The name of a platform
lic-platformRole	A server or client platform
lic-platformCPUmax	The maximum number of CPUs for a platform
lic-platformCPUinUse	The number of CPUs currently in use for a platform
lb-status	The status of the load balancing service
lb-ipAddress	The IP address of the load balancing service
lb-iiopPort	The IIOP port of the load balancing service
lb-objUsed	The number of object managers currently running
lb-objMax	The maximum number of object managers available
lb-objPeak	The maximum number of object managers used
lb-objMgrTable	A table of all object managers
lb-objMgrEntry	An entry in the object managers table
lb-objMgrLoad	The current load of object managers (in %)
lb-objMgrPort	The IIOP port number of the object manager
lb-objMgrStatus	The status of the object managers
lb-ejbUsed	The number of EJB managers currently running
lb-ejbMax	The maximum number of EJB managers available
lb-ejbPeak	The maximum number of EJB managers used
lb-ejbMgrTable	A table of all EJB managers
lb-ejbMgrEntry	An entry in the EJB managers table

Measurement	Description
lb-ejbMgrLoad	The current load of EJB managers (in %)
lb-ejbMgrPort	The IIOP port number of the EJB manager
lb-ejbMgrStatus	The status of EJB managers
lb-sloUsed	The number of stateless object managers currently running
lb-sloMax	The maximum number of stateless object managers available
lb-sloPeak	The maximum number of stateless object managers used
lb-sloMgrTable	A table of all stateless object managers
lb-sloMgrEntry	An entry in the stateless object managers table
lb-sloMgrLoad	The current load of stateless object managers (in %)
lb-sloMgrPort	The IIOP port number of the stateless object manager
lb-sloMgrStatus	The status of stateless object managers
trans-status	The status of the transaction service
trans-ipAddress	The IP address of the transaction service
trans-iiopPort	The IIOP port of the transaction service
trans-detOpen	The number of open transactions
trans-detExec	The number of transactions executed
trans-detCommit	The number of transactions committed
trans-detRollback	The number of transactions rolled back
trans-detFailed	The number of transactions failed
trans-detTimeout	The number of transactions timed out
mgrtr-status	The status of the transaction manager
mgrtr-ipAddress	The IP address of the transaction manager

Measurement	Description
mgrtr-iiopPort	The IIOP port of the transaction manager
snmp-status	The status of the snmp service
snmp-ipAddress	The IP address of the snmp service
snmp-iiopPort	The IIOP port of the snmp service
evt-status	The status of the event service
evt-ipAddress	The IP address of the event service
evt-iiopPort	The IIOP port of the event service
evt-detChannels	The number of channels registered to the event service
evt-detSuppliers	The number of event suppliers registered to the event service
evt-detConsumers	The number of event consumers registered to the event service
so-status	The status of the sharedObject service
so-ipAddress	The IP address of the sharedObject service
so-iiopPort	The IIOP port of the sharedObject service
so-sharedPools	The number of shared pools
so-sharedRDOs	The number of shared RDOs

ColdFusion Graph

The following default measurements are available for Allaire's ColdFusion server:

Measurement	Description
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 Out/sec	The number of bytes per second returned by the ColdFusion server.
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.

Fujitsu INTERSTAGE Graph

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

Microsoft Active Server Pages (ASP) Graph

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

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

Oracle9iAS HTTP Server Graph

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
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

Measurement	Description
<code>mod_setenvif.c</code>	Sets environment variables based on client information
<code>mod_actions.c</code>	Executes CGI scripts based on media type or request method
<code>mod_imap.c</code>	Handles imagemap files
<code>mod_asis.c</code>	Sends files that contain their own HTTP headers
<code>mod_log_config.c</code>	Provides user-configurable logging replacement for <code>mod_log_common</code>
<code>mod_env.c</code>	Passes environments to CGI scripts
<code>mod_alias.c</code>	Maps different parts of the host file system in the document tree, and redirects URLs
<code>mod_userdir.c</code>	Handles user home directories
<code>mod_cgi.c</code>	Invokes CGI scripts
<code>mod_dir.c</code>	Handles the basic directory
<code>mod_autoindex.c</code>	Provides automatic directory listings
<code>mod_include.c</code>	Provides server-parsed documents
<code>mod_negotiation.c</code>	Handles content negotiation
<code>mod_auth.c</code>	Provides user authentication using text files
<code>mod_access.c</code>	Provides access control based on the client hostname or IP address
<code>mod_so.c</code>	Supports loading modules (.so on UNIX, .dll on Win32) at run-time
<code>mod_oprocmgr.c</code>	Monitors JServ processes and restarts them if they fail
<code>mod_jserv.c</code>	Routes HTTP requests to JServ server processes. Balances load across multiple JServs by distributing new requests in round-robin order

Measurement	Description
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
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

Measurement	Description
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)
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 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.
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 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. <i>Up</i> implies that the agent can contact the server. <i>Down</i> 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
ServerQueueThroughput	The current throughput of execute queue, expressed as the number of requests processed per second

Measurement	Description
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 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
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 following measurements are available for the WebSphere 3.02 and 3.5.x (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

Measurement	Description
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
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

Measurement	Description
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
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
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

Measurement	Description
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
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
Committed	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

Measurement	Description
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 (Windows)*.

DB2 Graph

The DB2 graph shows the resource usage on the DB2 database server machine as a function of the elapsed scenario time.

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.
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".

Measurement	Description
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.
pipedsorts_requested	The number of piped sorts that have been requested.
pipedsorts_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.
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

Measurement	Description
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.
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.

Measurement	Description
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.
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.

Measurement	Description
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.
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.

Measurement	Description
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.
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.

Measurement	Description
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.
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.

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.

Measurement	Description
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.
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.

Measurement	Description
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.
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.

Measurement	Description
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.
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.

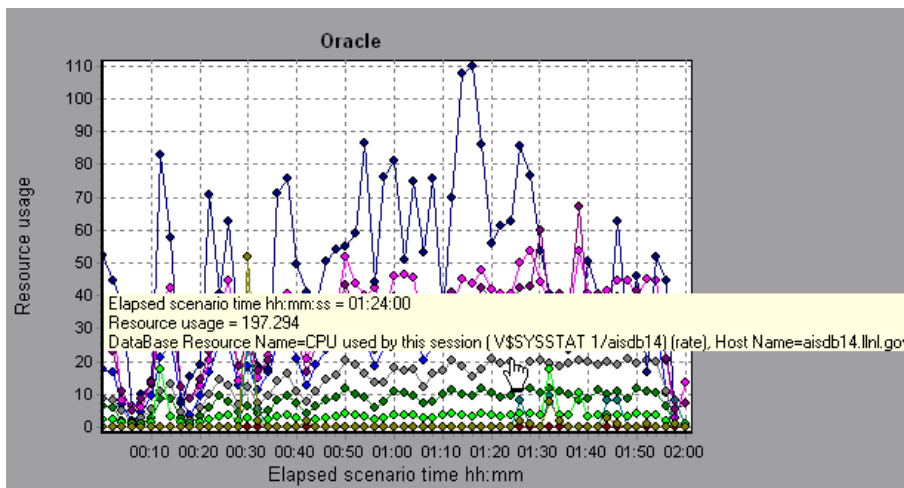
Measurement	Description
rows_inserted	The number of row insertions attempted.
rows_updated	The number of row updates attempted.
rows_selected	The number of rows that have been selected and returned to the application.
rows_written	The number of rows changed (inserted, deleted or updated) in the table.
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.
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.

Measurement	Description
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.
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, and system statistics, V\$SYSSTAT.

In the following Oracle graph, the V\$SYSSTAT resource values are shown as a function of the elapsed scenario time.



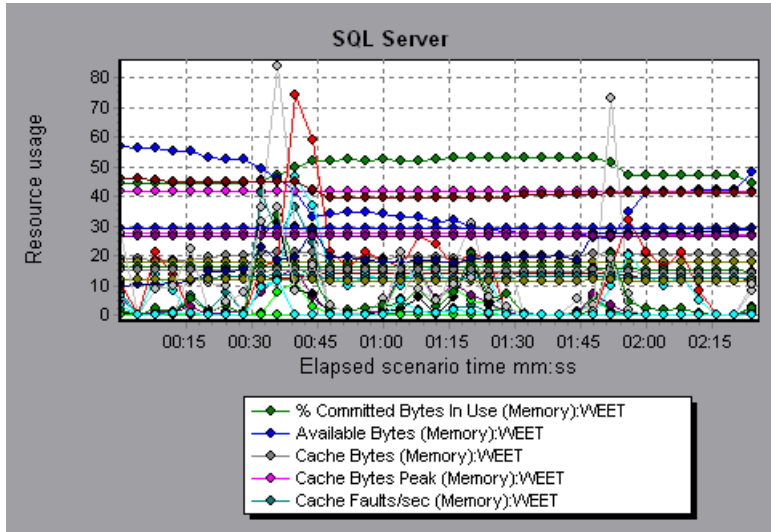
The following measurements are most commonly used when monitoring the Oracle server (from the V\$SYSSTAT table):

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

Measurement	Description
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
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 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)

Measurement	Description
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
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 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:

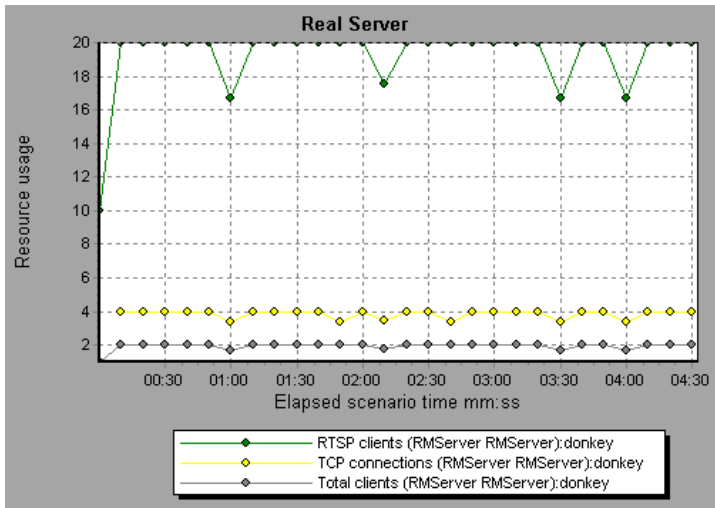
- ▶ RealPlayer Client Graph
- ▶ RealPlayer Server Graph
- ▶ Windows Media Server Graph
- ▶ Media Player Client Graph

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 (Windows)*.

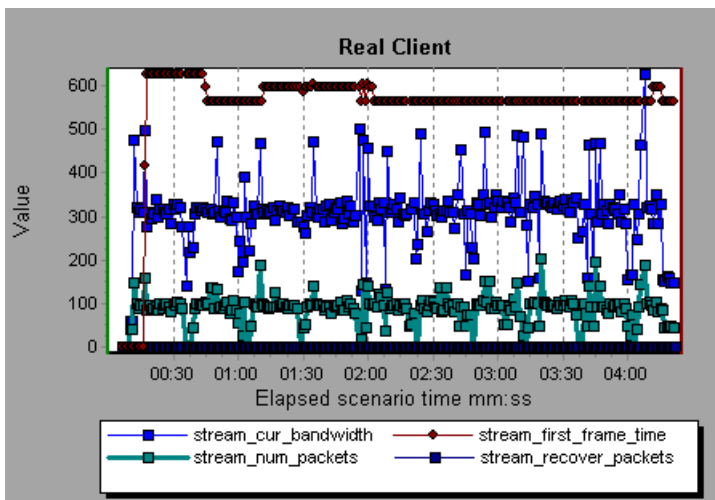
In order to display all the measurements on a single graph, the Analysis may scale them. The Legend tab indicates the scale factor for each resource. To obtain the true value, multiply the scale factor by the displayed value. For

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.



RealPlayer Client Graph

The Real Client graph shows statistics on the RealPlayer client machine as a function of the elapsed scenario time.



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.

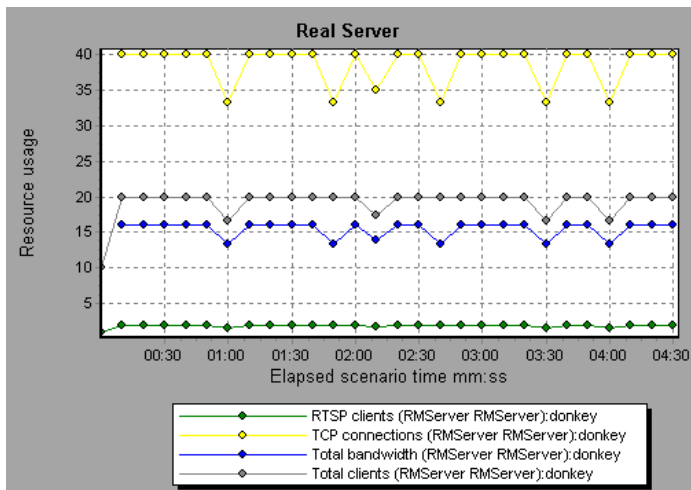
The following table describes the RealPlayer Client measurements that are monitored:

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
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

Measurement	Description
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

RealPlayer Server Graph

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



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.

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.

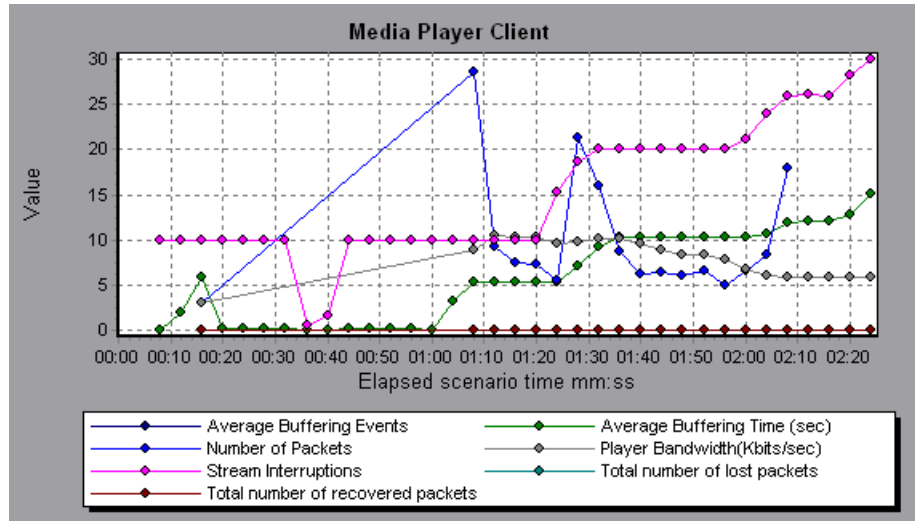
The following default measurements are available for the Windows Media Server:

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

Measurement	Description
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
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.



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
Stream Quality (Packet-level)	The percentage ratio of packets received to total packets
Current bandwidth (Kbits/sec)	The number of kbits per second received
Stream Packet Rate	The number of packets received
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.
Stream Quality (Sampling-level)	The percentage of stream samples received on time (no delays in reception)

16

ERP Server Resource Graphs

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

This chapter describes the:

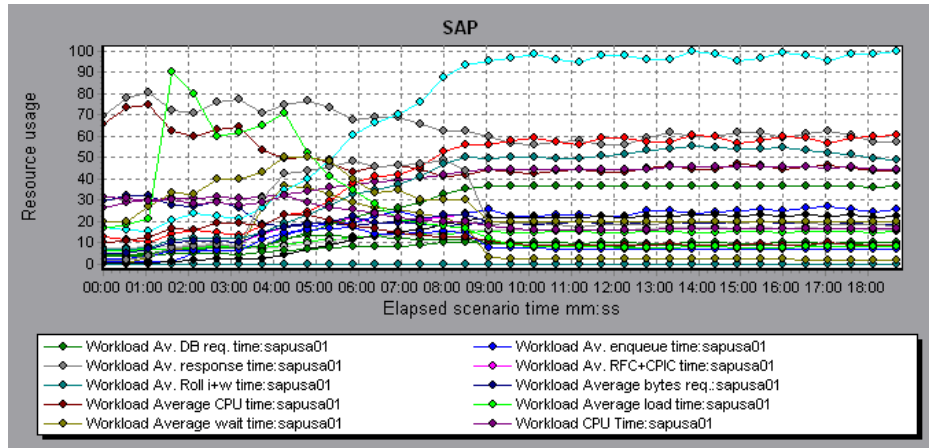
- SAP Graph

About ERP Server Resource Graphs

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

SAP Graph

The SAP graph shows the resource usage of a SAP R/3 system server as a function of the elapsed scenario time.



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.

Measurement	Description
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.
GUI time	The GUI time is measured in the work process and is the response time between the dispatcher and the GUI.
Roll ins	The number of rolled-in user contexts.
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.

Measurement	Description
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.

17

Java Performance Graphs

After a scenario run, you can use the Java performance monitor graphs to analyze the performance of Enterprise Java Bean (EJB) objects, Java-based applications, and the TowerJ Java virtual machine.

This chapter describes the:

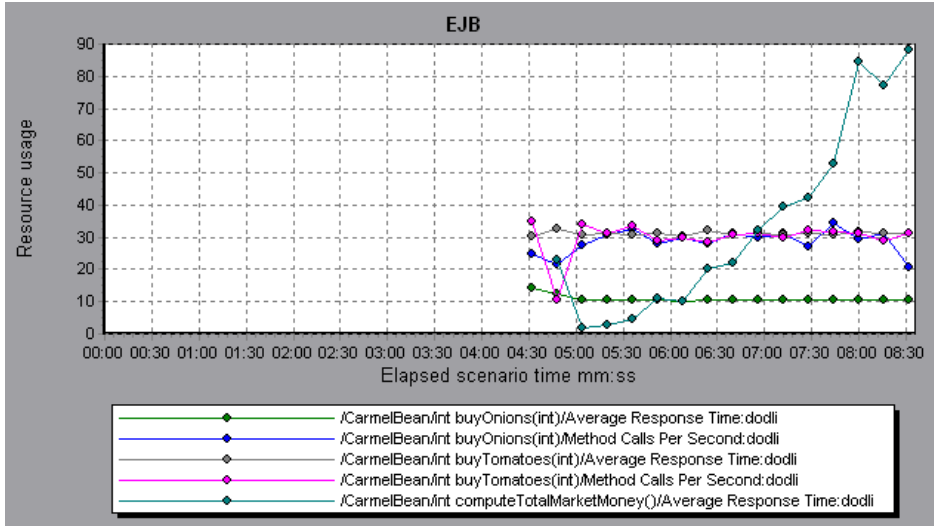
- EJB Graph
- JProbe Graph
- TowerJ Graph

About Java Performance Graphs

Java performance graphs provide you with performance information for Enterprise Java Bean (EJB) objects, Java-based applications, and the TowerJ Java virtual machine. Note that in order 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 (Windows)*.

EJB Graph

The EJB graph shows the resource usage of Enterprise Java Bean (EJB) objects on a WebLogic or WebSphere application server, as a function of the elapsed scenario time.



Note: There are differences in the scale factor between the two EJB object measurements.

The following counters can be measured for each EJB object method:

Measurement	Description
Average Response Time	The average response time of the EJB object being monitored.
Method Calls per Second	The number of EJB object method calls per second.

JProbe Graph

The JProbe graph shows the resource usage of Java-based applications as a function of the elapsed scenario time.

The following JProbe counters are available:

Measurement	Description
Allocated Memory (heap)	The amount of allocated memory in the heap (in bytes)
Available Memory (heap)	The amount of available memory in the heap (in bytes)

Note: There are differences in the scale factor between the two available measurements.

TowerJ Graph

The TowerJ graph shows the resource usage of the TowerJ Java virtual machine as a function of the elapsed scenario time.

The following measurements are available for the TowerJ Java virtual machine:

ThreadResource Measurement	Description
ThreadStartCountTotal	The number of threads that were started.
ThreadStartCountDelta	The number of threads that were started since the last report.
ThreadStopCountTotal	The number of threads that were stopped.
ThreadStopCountDelta	The number of threads that were stopped since the last report

GarbageCollection Resource Measurement	Description
GarbageCollectionCount Total	The number of times the garbage collector has run.
GarbageCollectionCount Delta	The number of times the garbage collector has run since the last report.
PreGCHeapSizeTotal	The total pre-GC heap space.
PreGCHeapSizeDelta	The total pre-GC heap space since the last report
PostGCHeapSizeTotal	The total post-GC heap space.
PostGCHeapSizeDelta	The total post-GC heap space since the last report.
NumPoolsTotal	The number of pools.
NumPoolsDelta	The number of pools since the last report.
NumSoBlocksTotal	The number of small object blocks.
NumSoBlocksDelta	The number of small object blocks since the last report.
NumLoBlocksTotal	The number of large object blocks.
NumLoBlocksDelta	The number of large object blocks.
NumFullSoBlocksTotal	The number of full small object blocks.
NumFullSoBlocksDelta	The number of full small object blocks since the last report.
TotalMemoryTotal	Total memory (heap size).
TotalMemoryDelta	Total memory (heap size) since the last report.
NumMallocsTotal	The number of current mallocs.
NumMallocsDelta	The number of current mallocs since the last report.
NumBytesTotal	The number of current bytes allocated.

GarbageCollection Resource Measurement	Description
NumBytesDelta	The number of current bytes allocated since the last report.
TotalMallocsTotal	The total number of mallocs.
TotalMallocsDelta	The total number of mallocs since the last report.
TotalBytesTotal	The total number of bytes allocated.
TotalBytesDelta	The total number of bytes allocated since the last report.

ExceptionResource Measurement	Description
ExceptionCountTotal	The number of exceptions thrown.
ExceptionCountDelta	The number of exceptions thrown since the last report.

ObjectResource Measurement	Description
NumberOfObjectsTotal	The number of objects.
NumberOfObjectsDelta	The number of objects since the last report.

Note: There are differences in the scale factor for some of the measurements.

18

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
- Creating a Merged Graph

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

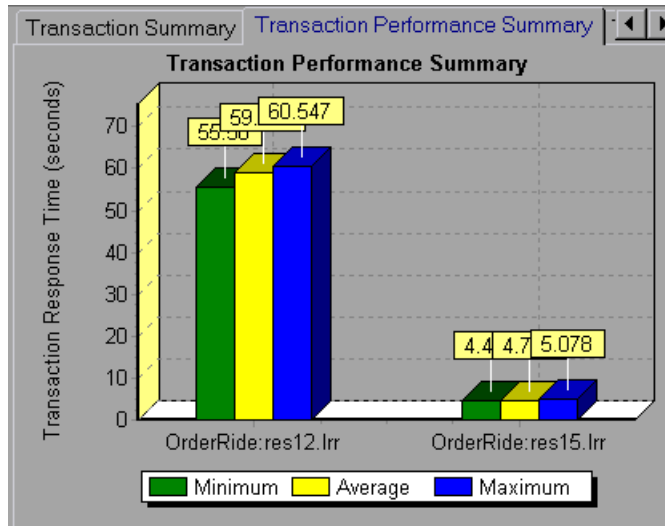
If you want to benchmark two hardware configurations, you run the same scenario with both configurations and compare the transaction response times using a single Cross Result graph.

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.



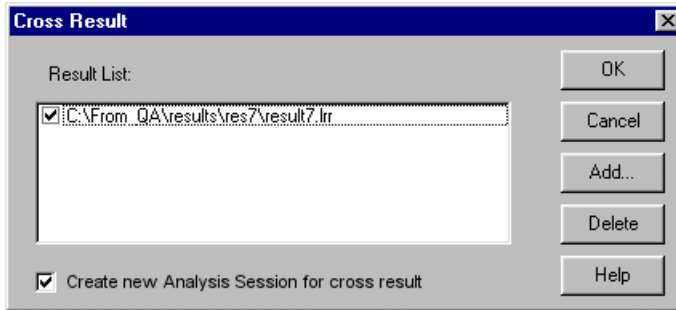
The Cross Result graphs have an additional filter and group by category: *Result Name*. The above graph is filtered to the *OrderRide* transaction for results *res12*, and *res15*, grouped by *Result Name*.

Generating Cross Result Graphs

You can create a Cross Result graph for two or more result sets.

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** check box.
- 6 Click **OK**. The 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

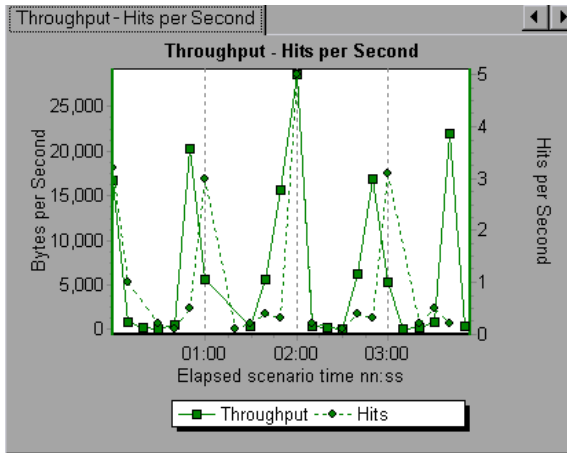
The Analysis lets you merge the results of two graphs from the same scenario into a single graph. The merging allows you to compare several different measurements at once. For example, you can make a merged graph to display the network delay and number of running Vusers, as a function of the elapsed time.

In order to merge graphs, their x-axis must be the same measurement. For example, you can merge Web Throughput and Hits per second, because the common x-axis is Scenario Elapsed Time. The drop-down list only shows the active graphs with an x-axis common with the current graph. The Analysis provides three types of merging:

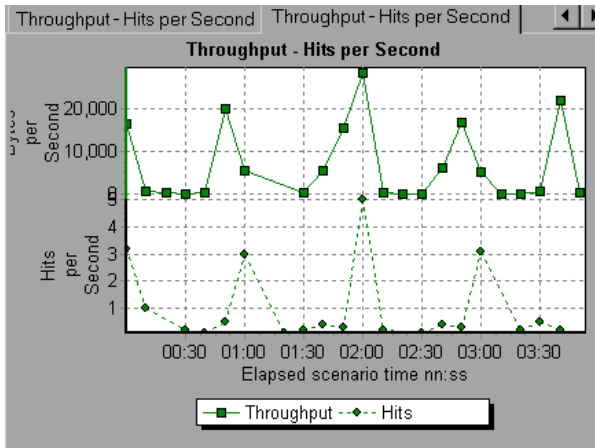
- Overlay
- Tile
- Correlate

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, the 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.

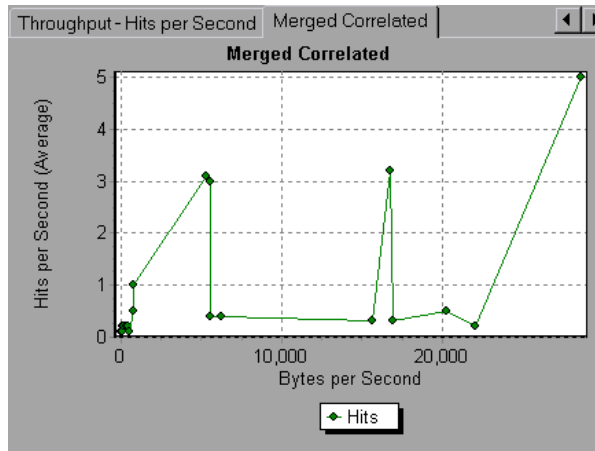


Tile: View contents of two graphs that share a common x-axis in a tiled layout, one above the other. In the following example the Throughput and Hits per Second graph are tiled one above the other.



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.

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.



Creating a Merged Graph

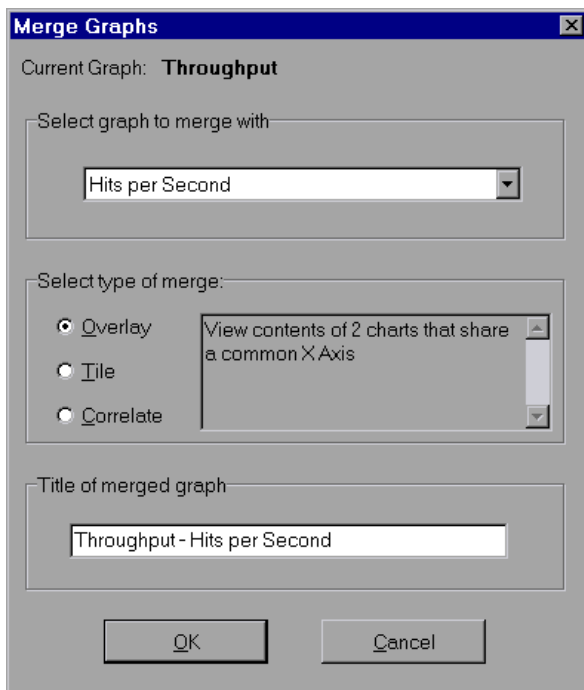
You can merge all graphs with a common x-axis.

To create a merged graph:

- 1 Select a graph in the tree view or click on its tab to make it active.



- 2 Choose **View > Merge Graphs** or click the **Merge Graphs** button. 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 type of merge: **Overlay**, **Tile**, or **Correlate**.
- 5 Specify a title for the merged graph. By default, the Analysis combines the titles of the two graphs being merged.
- 6 Click **OK**.
- 7 Filter the graph just as you would filter any ordinary graph.

19

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

About Analysis Reports

After running a scenario, you can view reports that summarize your system's performance. The 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 the Analysis to create an HTML report. The 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. It 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.

The screenshot shows the 'Analysis Summary' window with the following details:

- Scenario Name:** C:\Sanity_Analysis\Scenario\Scenario.lrs
- Results in session:** F:\Results\Sessions\v\Jres_amazon\res_amazon.lrs
- Duration:** 20 minutes and 49 seconds.

Statistics Summary

- Maximum Running Vusers:** 3
- Total Throughput (bytes):** 50,222,795
- Throughput (bytes/second):** Average: 40,210
- Total Hits:** 12,217
- Hits per Second:** Average: 10

Transaction Summary

Transactions: Total Passed: 73 Total failed: 73 Total aborted: 3

Transaction Name	Minimum	Average	Maximum	90 Percent	Pass	Fail	Abort
t_amazon_account	2.143	2.56	3.245	2.68	12	0	0
t_amazon_book	7.871	8.825	10.425	10.31	10	2	0

You can save the Summary report to an Excel file by selecting **View > Export Summary to Excel**.

Creating HTML Reports

The 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.

The screenshot shows a web browser window with the following content:

Reports List

- Summary
- Page Download Time Breakdown
- Web Page Breakdown
- Downloaded Content Size (KB)
- Transaction Performance Summary

Analysis Summary Period: 07/00/2001 10:05:00

Scenario Name: C:\Eonity\Analysis\Scenario\Scenario.lrs
 Results in session: F:\Results\Sessions\v70\res_amazon\v70Report.html
 Duration: 20 minutes and 49 seconds.


Statistics Summary

- Maximum Running Users: 3
- Total Throughput (bytes): 50,224,95
- Throughput (bytes/second): Average: 40,210
- Total Hits: 12,217
- Hits per Second: Average: 10

Transaction Summary

Transaction Name	Minimum	Average	Maximum	90 Percent	Pass	Fail
transaction_000001	2.143	2.56	3.245	2.66	12	0

To create HTML reports:

- 1 Open all graphs that you want to be included in the report.
- 2  Choose **Reports > HTML Report** or click the **Create HTML Report** button. The Select Report Filename and Path dialog box opens.

- 3 Specify a path and file name for the HTML report and click **OK**. The 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, the Analysis opens your default browser and displays the Summary report.
- 4 To view an HTML report for one of the graphs, click on 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

The Analysis provides several built-in reports which contain detailed summaries about the scenario, the transactions and Vusers.

To display a report:

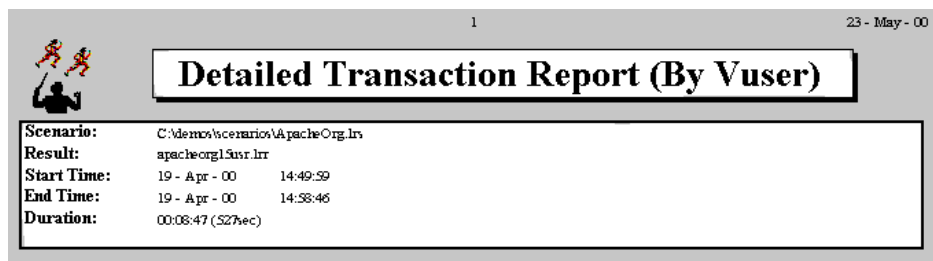
- 1 Open the desired Analysis session file (.lra extension), or LoadRunner result file (.lrr extension), if it is not already open.
- 2 From the Reports menu choose a report. The report is generated and displayed. You can display multiple copies of the same report.

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.

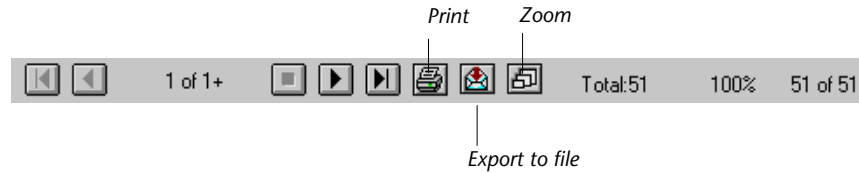
Start time: The time at which the Run Scenario command was executed.

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.



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						
<i>Vuser</i>	<i>Host</i>	<i>Ready At</i>	<i>Running At</i>	<i>Duration</i>	<i>Termination Status</i>	
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

<i>Transaction</i>	<i>Start time</i>	<i>End time</i>	<i>Duration</i>
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

<i>Transaction</i>	<i>Start time</i>	<i>End time</i>	<i>Duration</i>
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.

<u>Group:</u> g2						
<i>Vuser</i>	<i>Host</i>	<i>Ready At</i>	<i>Running At</i>	<i>Duration</i>	<i>Termination Status</i>	
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

<i>Vusers:</i>	5			
<i>Failed:</i>	0	<i>Error:</i>	0	<i>Stopped:</i> 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 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

<i>Transaction</i>	<i>Start time</i>	<i>End time</i>	<i>Duration</i>	<i>Think Time</i>	<i>Wasted Time</i>	<i>Result</i>
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

			<u>P e r f o r m a n c e (sec)</u>				
<i>Vuser</i>	<i>Pass</i>	<i>Fail</i>	<i>Min</i>	<i>Avg</i>	<i>Max</i>	<i>STD</i>	
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	

20

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
- ▶ Creating a New Session Using TestDirector
- ▶ Opening an Existing 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.

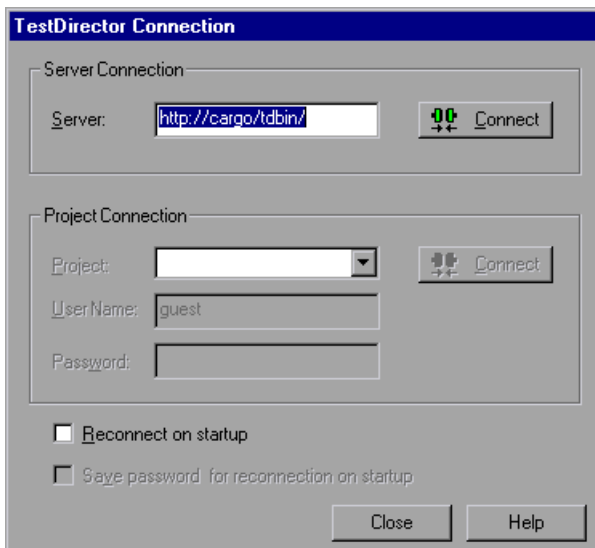
Connecting LoadRunner to TestDirector

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 the 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 read-only format in the Server box.
- 4 From the Project box in the Project Connection section, select a TestDirector project.
- 5 In the User Name box, type a user name.
- 6 In the Password box, type a password.
- 7 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.

- 8 To automatically reconnect to the TestDirector server and the selected project on startup, select the **Reconnect on startup** check box.
- 9 If you select **Reconnect on startup**, you can save the specified password to reconnect on startup. Select the **Save password for reconnection on startup** check box.

If you do not save your password, you will be prompted to enter it when LoadRunner connects to TestDirector on startup.

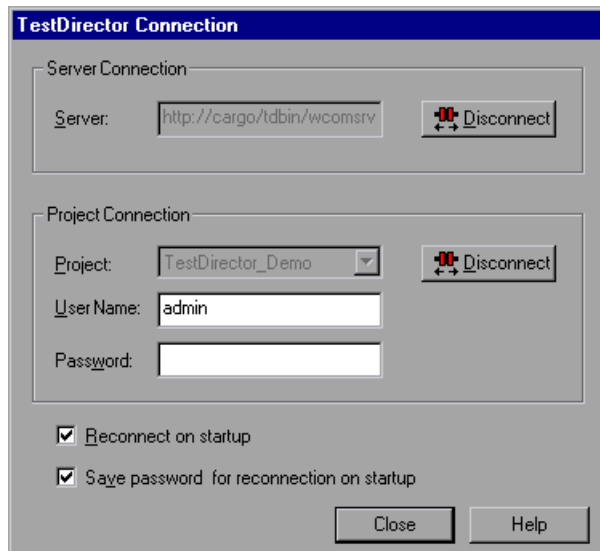
- 10 Click **Close** to close the TestDirector Connection dialog box.

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.



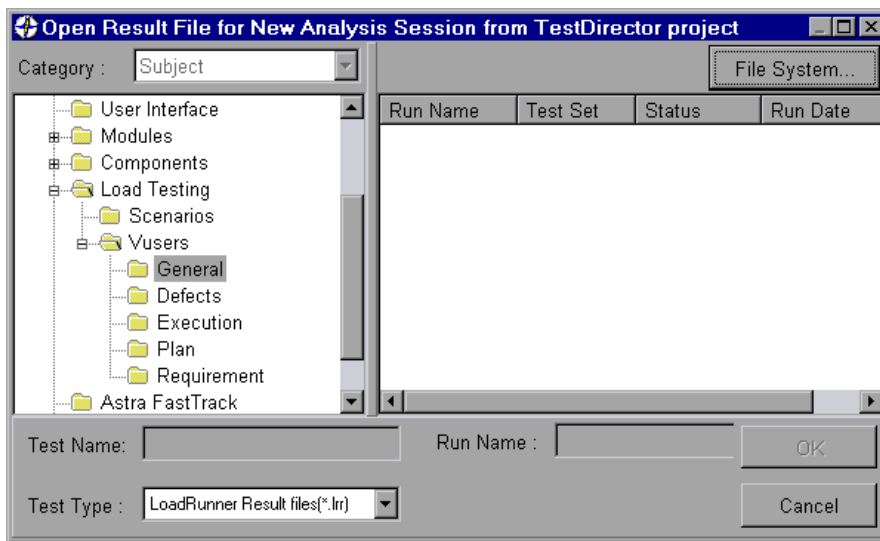
- 2 To disconnect LoadRunner from the selected project, click **Disconnect** in the Project Connection section.
- 3 To disconnect LoadRunner from the selected server, click **Disconnect** in the Server Connection section.
- 4 Click **Close** to close the TestDirector Connection dialog box.

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.

To create a new session using results from a TestDirector project:

- 1 Connect to the TestDirector server (see “Connecting LoadRunner to TestDirector” on page 230).
- 2 In the Analysis, choose **File > New** or click the **Create New Analysis Session** button. The Open Result File for New Analysis Session from TestDirector Project dialog box opens and displays the test plan tree.



To open a result file directly from the file system, click the **File System** button. The Open Result File for New Analysis Session dialog box opens. (From the Open Result File for New Analysis Session dialog box, you may return to the Open Result File for New Analysis Session from TestDirector Project dialog box by clicking the TestDirector button.)

- 3** Click 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.

- 4** Select an Analysis session from the Run Name list. The scenario appears in the read-only Test Name box.
- 5** Click **OK** to open the session. LoadRunner loads the session. The name of the session appears in the Analysis title bar.

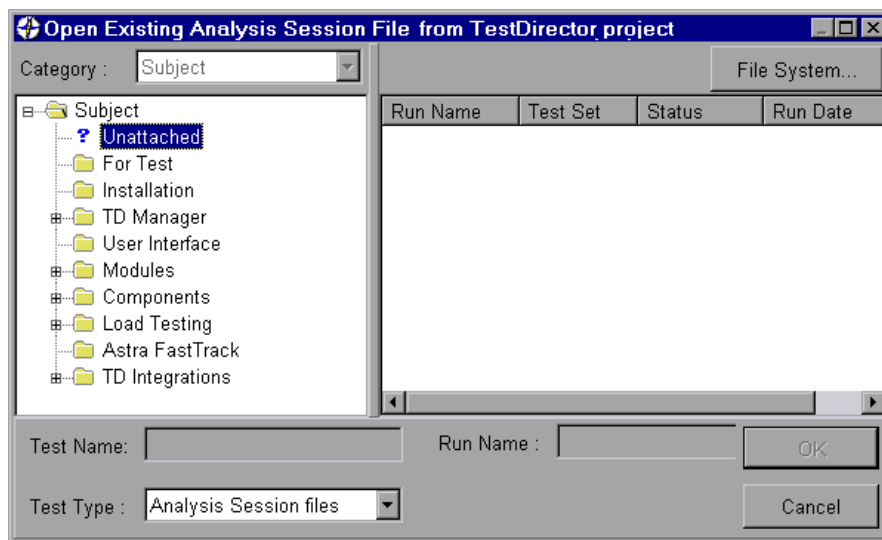
Note: You can also open Analysis sessions from the recent session list in the File menu. If you select a session located in a TestDirector project, but LoadRunner is currently not connected to that project, the TestDirector Connection dialog box opens. Enter your user name and password to log in to the project, and click **OK**.

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.

To open a session from a TestDirector project:

- 1** Connect to the TestDirector server (see “Connecting LoadRunner to TestDirector” on page 230).
- 2** In the Controller, choose **File > Open** or click the **File Open** button. The Open Existing Analysis Session File from TestDirector Project dialog box opens and displays the test plan tree.



To open a scenario directly from the file system, click the **File System** button. The Open Existing Analysis Session File dialog box opens. (From the Open Existing Analysis Session File dialog box, you may return to the Open Existing Analysis Session File from TestDirector Project dialog box by clicking the TestDirector button.)

- 3 Click 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.

- 4 Select a session from the Run Name list. The session appears in the read-only Test Name box.
- 5 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 LoadRunner is 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 (see “Connecting LoadRunner to TestDirector” on page 230).
- 2 Select **File > Save** and save the session in the TestDirector data directory.

21

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

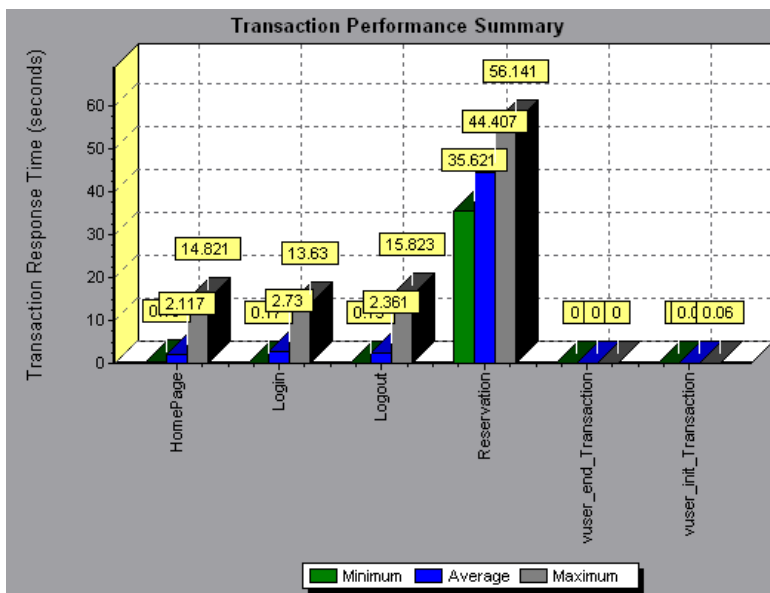
Note: The chapter presents examples from Web load tests.

Analyzing Transaction Performance

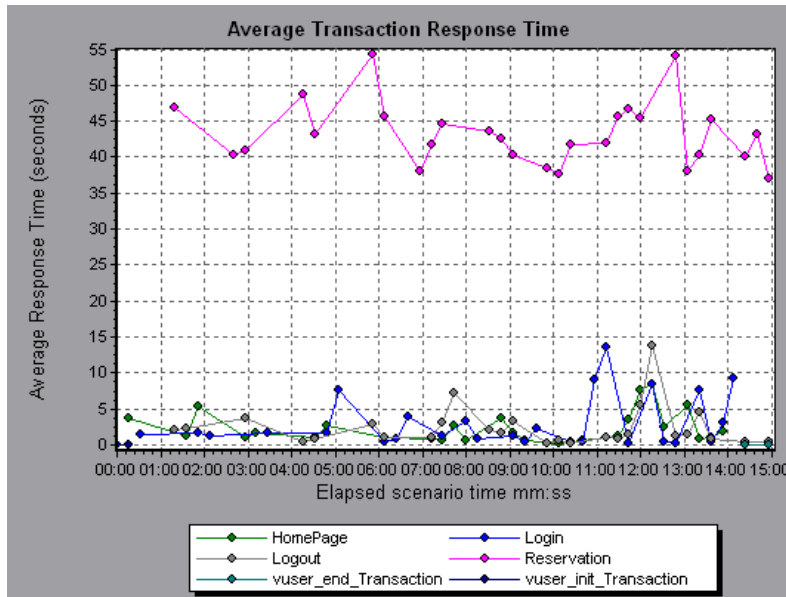
The **Average Transaction Response Time** and **Transaction Performance Summary** graphs should be your starting point in analyzing a scenario run. Using the Transaction Performance Summary graph, you can determine which transaction(s) had a particularly high response time during scenario execution. 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.



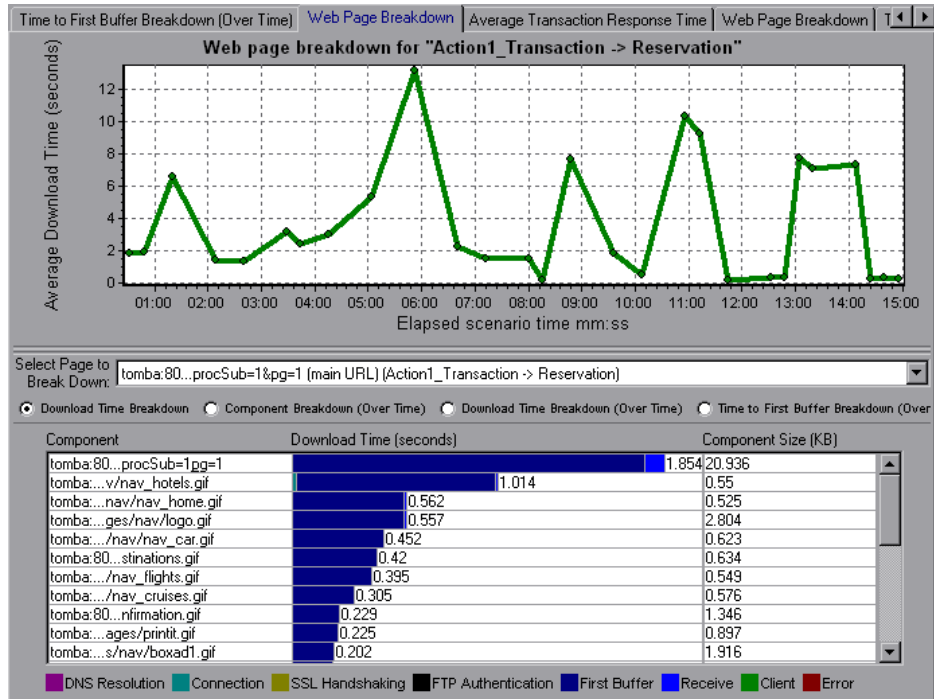
In order to pinpoint the problem and understand why response time was high for the Reservation transaction during this scenario, it is necessary to break down the transactions and analyze the performance of each page component. To break down a transaction, right-click it in the Average Transaction Response 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.

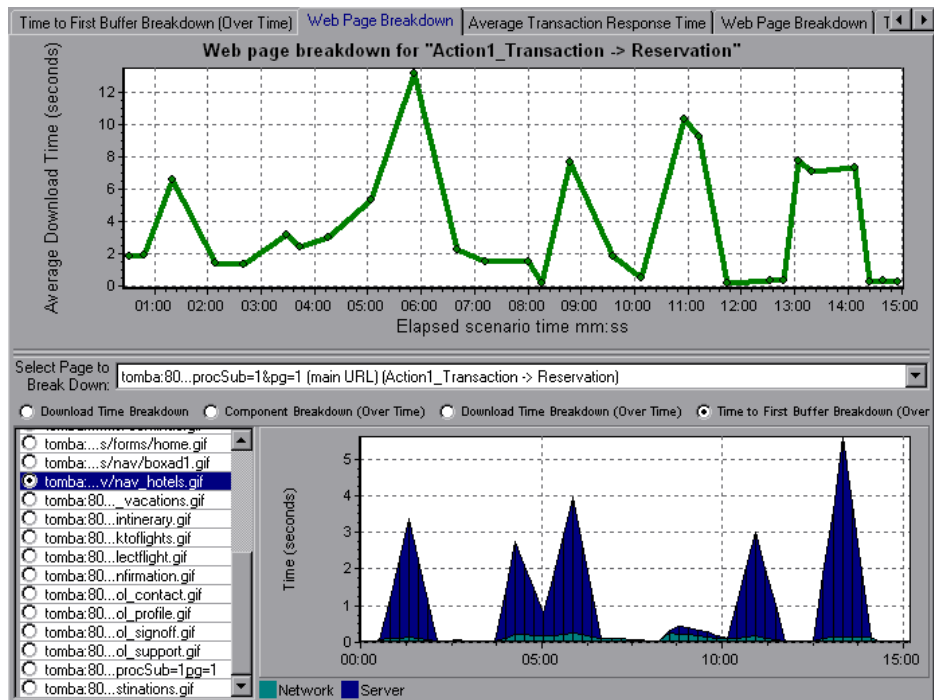
Question 2: Which page components were responsible for the high transaction response time? Were the problems that occurred network- or server-related?

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 87.

To identify whether a problem is network- or server-related, select the Time to First Buffer Breakdown (Over Time).



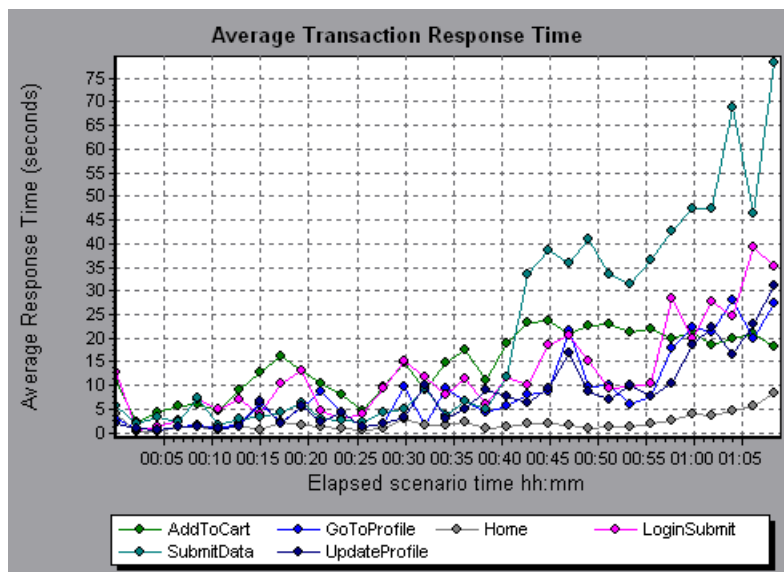
The above graph demonstrates that the server time was much higher than the network time. If the server time is unusually high, use the appropriate server graph to identify the problematic server measurements and isolate the cause of server degradation. If the network time is unusually high, use the Network 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.

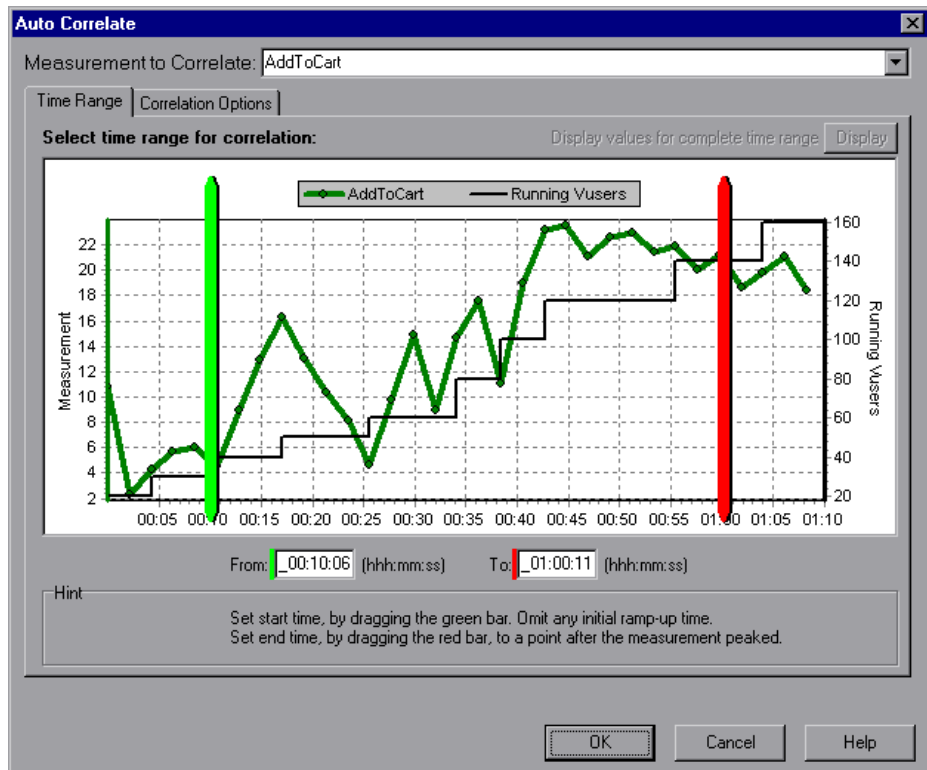
Question 3: Did a bottleneck occur in the system? If so, what was the cause of the problem?

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.

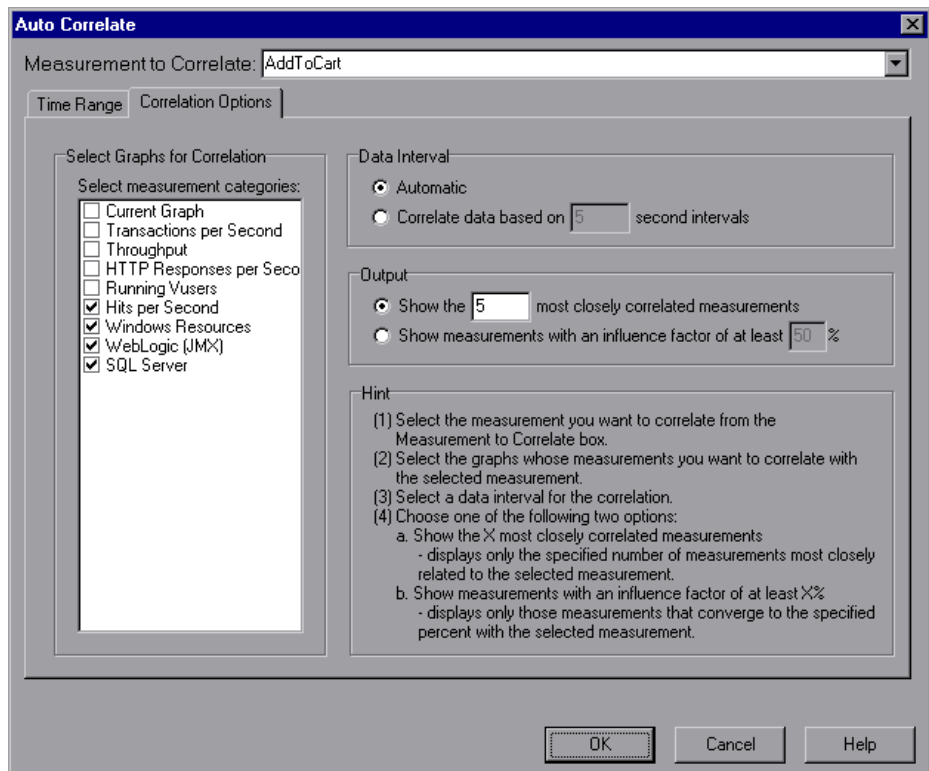


The above graph demonstrates that the response time for the SubmitData transaction was relatively high toward the end of the scenario. To correlate this transaction with all of the measurements collected during the scenario, right-click the SubmitData transaction and select **Auto Correlate**.

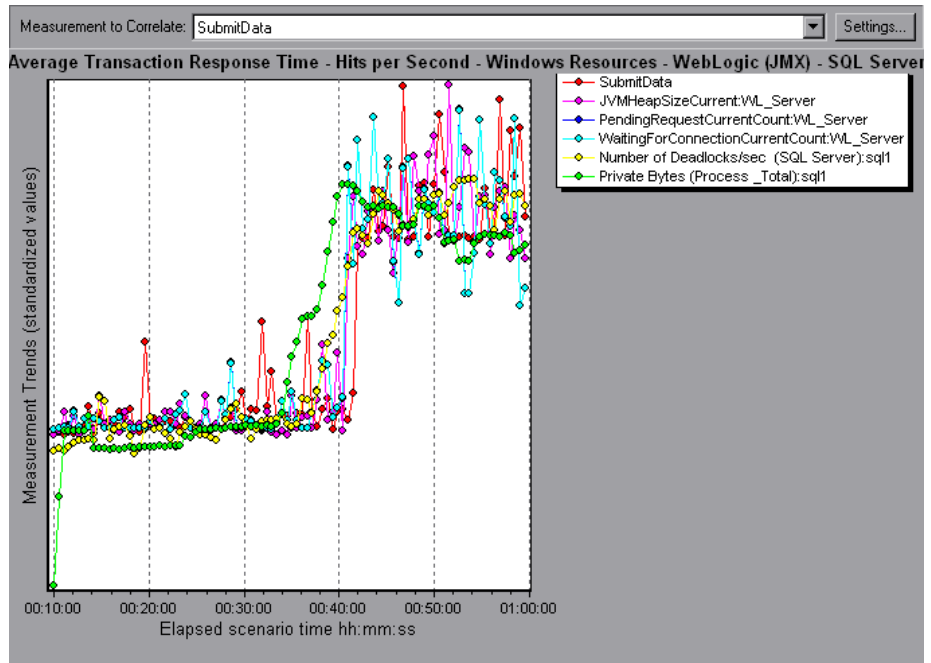
In the dialog box that opens, choose the time frame you want to examine.



Click the **Correlation Options** tab, select the graphs whose data you want to correlate with the SubmitData transaction, and click **OK**.

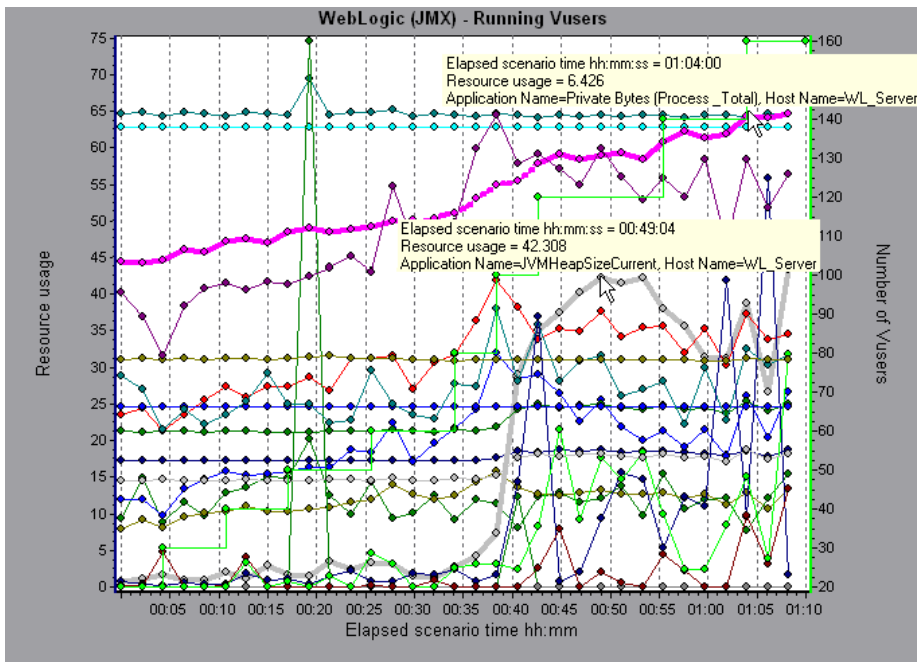


In the following graph, the Analysis displays the five measurements most closely correlated with the SubmitData transaction.



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

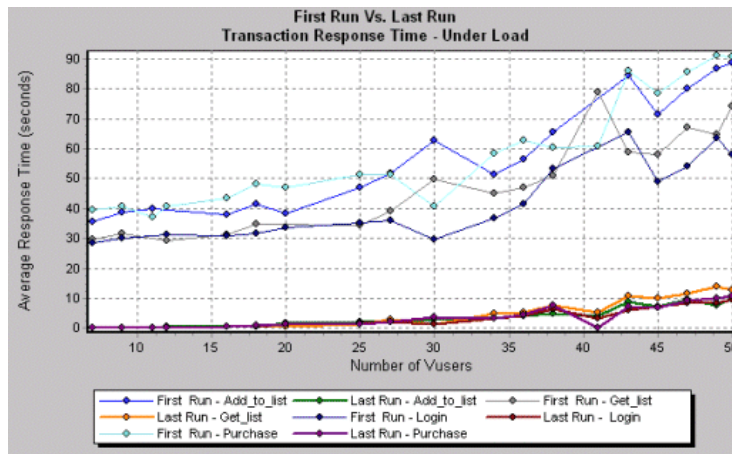
Network bottlenecks can usually be identified when the load increment is considerable and is not significantly affecting any of the server side components, as in the case of informational sites using many static Web pages. In twenty-five percent of such cases, the pipe to the Internet cannot sufficiently 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 Users, 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 the Analysis, therefore, the customer was able to increase site performance tenfold.

Index

A

- Acrobat Reader ix
- Activity reports 221
- advanced display settings
 - chart 36
 - series 37
- aggregating data 5
- Analysis
 - interpreting graphs 237–249
 - overview 1–20
 - sessions 3
 - working with 21–46
- Apache graph 122
- Ariba graph 131
- ASP graph 149
- ATG Dynamo graph 133
- auto correlating measurements 43, 242
- Average Transaction Response Time graph 56, 238, 242

B

- Books Online ix
- breaking down transactions 82
- BroadVision graph 136
- Brokat Twister graph 143

C

- chart settings 36
- Check Point FireWall-1 graph 120
- Client Time 89
- ColdFusion graph 147
- collating scenario results 4
- comparing scenario runs 249
- connecting to TestDirector 230
- Connection time 88

- Context Sensitive Help x
- coordinates of a point 22
- Cross Result graphs 209–216

D

- data aggregation 5
- Data Point report 226
- Data Points (Average) graph 101
- Data Points (Sum) graph 100
- Database options 10
- DB2 graph 168
- Detailed Transaction report 227
- disconnecting from TestDirector 232
- display options
 - advanced 36
 - standard 34
- DNS Resolution time 88
- documentation set x
- Downloaded Component Size graph 97
- drill down 23

E

- EJB graph 204
- enlarging graphs 26
- ERP Server Resource graphs 199–202
- Error graphs 51–53
- Error Statistics graph 52
- Error Time 89
- Errors per Second graph 53
- Excel file
 - exporting to 40
 - viewing 220

F

- Failed Transaction report 224
- Failed Vuser report 225
- filtering graphs 29
- FireWall Server graphs 119–120
- First Buffer time 88
- FTP Authentication time 89
- Fujitsu INTERSTAGE graph 148
- Function Reference ix

G

- General options 9
- granularity 26
- graph
 - Apache 122
 - Ariba 131
 - ATG Dynamo 133
 - Average Transaction Response Time 56
 - BroadVision 136
 - Brokat Twister 143
 - Check Point FireWall-1 120
 - ColdFusion 147
 - Data Points (Average) 101
 - Data Points (Sum) 100
 - DB2 168
 - Downloaded Component Size 97
 - EJB 204
 - Error Statistics 52
 - Errors per Second 53
 - Fujitsu INTERSTAGE 148
 - Hits per Second 68
 - HTTP Responses per Second 71
 - HTTP Status Code Summary 70
 - iPlanet/Netscape 126
 - JProbe 205
 - Microsoft Active Server Pages (ASP) 149
 - Microsoft IIS 124
 - Network Delay Time 115
 - Network Segment Delay 117
 - Network Sub-Path Time 116
 - Oracle 182
 - Oracle9iAS HTTP 150
 - Page Component Breakdown 83

graph (cont'd)

- Page Component Breakdown (Over Time) 85
- Page Download Time Breakdown 87
- Page Download Time Breakdown (Over Time) 91
- Pages Downloaded per Second 74
- RealPlayer Client 192
- RealPlayer Server 194
- Rendezvous 50
- Retries per Second 76
- Retries Summary 77
- Running Vusers 48
- SAP 200
- SilverStream 154
- SNMP Resources 110
- SQL Server 184
- Sybase 186
- Throughput 69
- Time to First Buffer Breakdown 93
- Time to First Buffer Breakdown (Over Time) 95
- Total Transactions per second 59
- TowerJ 205
- Transaction Performance Summary 61
- Transaction Response Time (Distribution) 64
- Transaction Response Time (Percentile) 63
- Transaction Response Time (Under Load) 62
- Transaction Summary 60
- Transactions per second 58
- TUXEDO Resources 111
- Vuser Summary 49
- WebLogic (JMX) 157
- WebLogic (SNMP) 155
- WebSphere 160
- Windows Media Player Client 197
- Windows Media Server 195
- Windows Resources 104

graph types, Analysis

- Database Server Resources 167–190
- ERP Server Resource Monitor 199–202
- Errors 51–53
- FireWall Server Monitor 119–120
- Java Performance 203–207
- Network Monitor 113–118
- Streaming Media Resources 191–198
- System Resources 103–112
- Transaction 55–65
- User-Defined Data Points 99–101
- Uuser 47–50
- Web Application Server
 - Resources 129–166
- Web Page Breakdown 79–98
- Web Resources 67–77
- Web Server Resources 121–127

graphs, working with

- background 37
- crossing results 209–216
- display options 34
- merging 213
- overlying, superimposing 213

H

- Hits per Second graph 68
- HTML reports 220
- HTTP Responses per Second graph 71
- HTTP Status Code Summary graph 70
- HTTPS 247

I

- IIS graph 124
- interpreting Analysis graphs 237–249
- iPlanet/Netscape graph 126

J

- Java Performance graphs 203–207
- JProbe graph 205

L

- legend 38
- legend preferences 37
- lr_user_data_point 99

M

- marks in a graph 37
- measurement trends, viewing 42
- measurements, auto correlating 43, 242
- Media Player Client graph 197
- Merging graphs 213
- Microsoft Active Server Pages (ASP)
 - graph 149
- Microsoft IIS graph 124
- MSDE 12

N

- Network Delay Time graph 115
- Network Monitor graphs 113–118
- Network Segment Delay graph 117
- Network Sub-Path Time graph 116

O

- Oracle graph 182
- Oracle9iAS HTTP graph 150
- Output window 2
- overlay graphs 213

P

- packets 114
- Page Component Breakdown (Over Time)
 - graph 85
- Page Component Breakdown graph 83
- Page Download Time Breakdown
 - (Over Time) graph 91
- Page Download Time Breakdown graph 87
- Pages Downloaded per Second graph 74
- Performance reports 221

R

- raw data 40
- RealPlayer Client graph 192
- RealPlayer Server graph 194
- Receive time 89
- Rendezvous graph 50
- report
 - Data Point 226
 - Detailed Transaction 227
 - Failed Transaction 224
 - Failed Vuser 225
 - Scenario Execution 223
 - Transaction Performance by Vuser 228
- report viewer 222
- reports 217–228
 - Activity and Performance 221
 - displaying 221
 - HTML 220
 - summary 219
- Retries per Second graph 76
- Retries Summary graph 77
- Running Vusers graph 48
- run-time settings 17

S

- SAP graph 200
- scale factor 121, 191
- scale of graph 26
- Scenario Execution report 223
- scenario run-time settings 17
- security problems 247
- session information 16
- sessions 3
- SilverStream graph 154
- SNMP Resources graph 110
- spreadsheet view 40
- SQL Server graph 184
- SSL Handshaking time 88
- standardizing y-axis values 42
- Streaming Media graphs 191–198
- summary data, viewing 4
- Summary report 219
- superimposing graphs 213
- Support Information x

- Support Online x
- Sybase graph 186

T

- templates
 - applying 15
 - saving 14
- TestDirector
 - integration 229–236
 - opening a new session 233
 - opening an existing session 235
 - saving sessions to a project 236
 - TestDirector integration
 - connecting to TestDirector 230
 - disconnecting from TestDirector 232
- three-dimensional properties 37
- Throughput graph 69
- time filter, setting 7
- Time to First Buffer Breakdown (Over Time) graph 95
- Time to First Buffer Breakdown graph 93
- tool options
 - setting database options 10
 - setting general options 9
- Total Transactions per Second graph 59
- TowerJ graph 205
- Transaction graphs 55–65
- Transaction Performance by Vuser report 228
- Transaction Performance Summary graph 61, 238
- Transaction Response Time graphs 56–65
 - Average 56
 - Distribution 64
 - Percentile 63
 - Under Load 62
- Transaction Summary graph 60
- transaction, breakdown 82
- Transactions per Second graph 58
- TUXEDO graph 111

U

user_data_point function 99
User-Defined Data Point graphs 99–101

V

viewing measurement trends 42
Vuser graphs 47–50
Vuser Summary graph 49

W

Web Application Server Resource
graphs 129–166
Web Page Breakdown graphs 79–98, 240
activating 81
Web Resource graphs 67–77
Web Server Resource graphs 121–127
WebLogic (JMX) graph 157
WebLogic (SNMP) graph 155
WebSphere graph 160
Windows Media Server graph 195
Windows Resources graph 104

X

x-axis interval 26

Y

y-axis values, standardizing 42

Z

zoom 26



MERCURY INTERACTIVE

Mercury Interactive Corporation

1325 Borregas Avenue
Sunnyvale, CA 94089 USA

Main Telephone: (408) 822-5200

Sales & Information: (800) TEST-911

Customer Support: (877) TEST-HLP

Fax: (408) 822-5300

Home Page: www.mercuryinteractive.com

Customer Support: support.mercuryinteractive.com



L RAN 007. 5/01