Oracle® Application Testing Suite

Getting Started Guide

Version 9.20

E15487-04

November 2010

ORACLE®

Oracle Application Testing Suite Getting Started Guide, Version 9.20

E15487-04

Copyright © 1997, 2010, Oracle and/or its affiliates. All rights reserved.

Primary Author: Rick Santos

Contributing Author: Mary Anna Brown

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this software or related documentation is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation shall be subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License (December 2007). Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

This software is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications which may create a risk of personal injury. If you use this software in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure the safe use of this software. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software in dangerous applications.

Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

This software and documentation may provide access to or information on content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

Contents

3 Oracle OpenScript Tutorial

4 Oracle Load Testing Tutorial

5 Oracle Test Manager Tutorial

List of Figures

Preface

Welcome to Getting Started with Oracle Application Testing Suite. This guide explains how to get started using the features and options of Oracle OpenScript and Oracle Load Testing for testing Web pages or applications, and Oracle Test Manager for managing requirements, issues, and tests.

Audience

This guide is for Web test engineers who will be using the Oracle Application Testing Suite applications for regression testing, performance testing (load and scalability), and monitoring of a Web site or application.

Prerequisites

The tutorials in this guide assume an understanding of software or Web application testing concepts. Test engineers using the Oracle Application Testing Suite should be familiar with the concepts of regression testing, load testing, scalability testing, and operational monitoring.

Using This Guide

This guide is organized as follows:

Chapter 1, "Introduction" provides an overview of the major features of the tools included in the Oracle Application Testing Suite.

Chapter 2, "Oracle Application Testing Suite Basics" provides descriptions of the products in the Oracle Application Testing Suite and the main features of each.

Chapter 3, "Oracle OpenScript Tutorial" provides step-by-step instructions and explanations for building test scripts for testing Web pages or applications with Oracle OpenScript. The tutorial includes examples that highlight the script features, Databanks, and test cases.

Chapter 4, "Oracle Load Testing Tutorial" provides step-by-step instruction for using multiple OpenScript scripts to perform load and scalability testing of Web applications and back end systems. This chapter also explains how to configure ServerStats and generate reports from testing data.

Chapter 5, "Oracle Test Manager Tutorial" provides step-by-step instruction for using the main features of Oracle Test Manager. The tutorial includes examples for defining requirements and tests, and entering issues.

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible to all users, including users that are disabled. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Accessibility standards will continue to evolve over time, and Oracle is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For more information, visit the Oracle Accessibility Program Web site at http://www.oracle.com/accessibility/.

Accessibility of Code Examples in Documentation

Screen readers may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, some screen readers may not always read a line of text that consists solely of a bracket or brace.

Accessibility of Links to External Web Sites in Documentation

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/support/contact.html or visit http://www.oracle.com/accessibility/support.html if you are hearing impaired.

Related Documents

For more information, see the following documents in the Oracle Application Testing Suite documentation set:

- *Oracle Application Testing Suite Release Notes*
- *Oracle Functional Testing OpenScript User's Guide*
- *Oracle Load Testing Load Testing User's Guide*
- *Oracle Load Testing Load Testing ServerStats Guide*
- *Oracle Test Manager Test Manager User's Guide*

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1 Introduction

Oracle Application Testing Suite is an integrated, comprehensive Web application testing solution that provides all the tools you need to ensure the scalability and reliability of your business-critical applications.

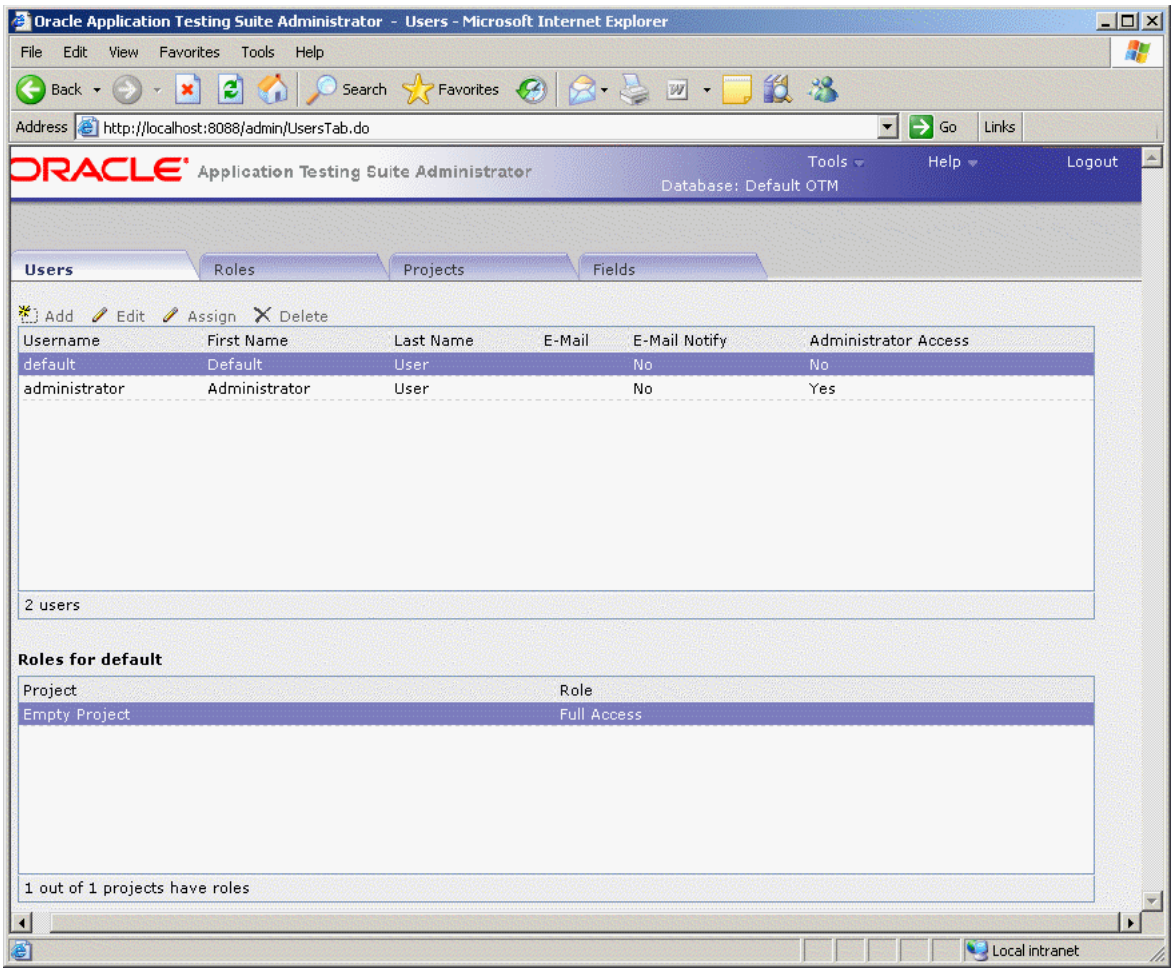
- Oracle OpenScript is an updated scripting platform for creating automated extensible test scripts in Java.
- Oracle Load Testing for load, scalability and stress testing. Oracle Load Testing also includes tools for server side monitoring and reporting
- Oracle Test Manager for organizing and managing your overall testing process.

This manual introduces you to the Oracle Application Testing Suite and provides step-by-step tutorials to help you get started using the tools.

1.1 About Oracle Application Testing Suite Administrator

The Administrator allows the Oracle Application Testing Suite system administrator to manage user accounts for Oracle Load Testing and Oracle Test Manager. For Oracle Load Testing, you define user accounts and the type of access allowed. For Oracle Test Manager, you define user accounts, roles, projects, and fields.

Figure 1–1 Administrator Main Window (Test Manager Database)



1.1.1 Administrator Feature Highlights

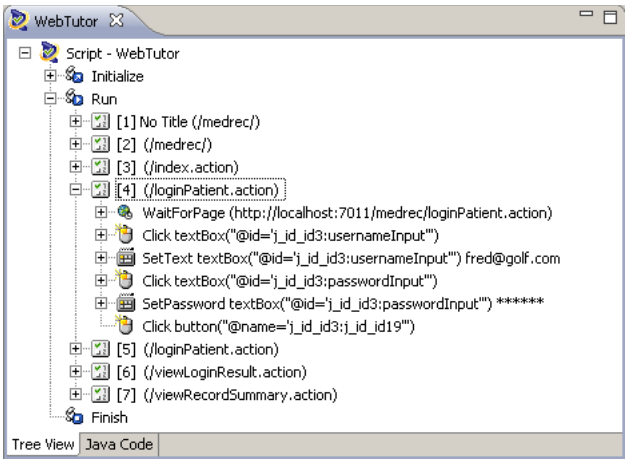
Oracle Application Testing Suite Administrator provides the following features:

- **User Accounts** - user accounts can be defined and customized for either Oracle Load Testing or Oracle Test Manager.
- **Roles** - roles with access permissions for read, write, delete, and execute can be defined and customized for Oracle Test Manager.
- **Projects** - projects can be defined and customized and assigned to specific user accounts for Oracle Test Manager.
- **Fields** - fields can be defined and customized for requirements, issues, tests, and test runs for Oracle Test Manager.

1.2 About Oracle OpenScript

Oracle OpenScript is built on a standards-based platform and provides the foundation for OpenScript Modules and Application Programming Interfaces (APIs). Combining an intuitive graphical interface with the robust Java language, OpenScript serves needs ranging from novice testers to advanced QA automation experts.

Figure 1–2 OpenScript Tree View Hierarchy



OpenScript APIs are used to build scripts for testing Web applications. The OpenScript API consists of a set of procedures that can be used to customize the scripts within the development environment. The API can also be used by advanced technical users to enhance scripts for unique testing needs

1.2.1 OpenScript Feature Highlights

OpenScript is the next generation environment for developing Oracle Application Testing Suite scripts for Web application testing. OpenScript offers the following advantages for Web-based application testing:

- **Scripting Workbench** - OpenScript provides an Eclipse -based scripting Workbench where you can create and run your automated test scripts. Users can use the Tree View graphical scripting interface for creating and editing scripts through the UI. Users can also switch to the Java Code View programming interface and leverage the integrated Eclipse IDE for creating and editing their scripts programmatically.

  Functional test scripts created in OpenScript can be played back to test and validate application functionality. Load test scripts created in OpenScript will run in Oracle Load Testing for application load testing, allowing users to simulate hundreds our thousands of users executing scripts at the same time.

- **Test Modules** - The OpenScript Test Modules provide application-specific test automation capabilities. Each Test Module is custom built to test a specific application or protocol. OpenScript includes several functional and load testing modules for testing Web-based applications. Additional modules can be developed for the OpenScript platform.

  OpenScript's Test Module interface is completely open and extendable by end-users. Users can leverage the Test Module API to build their own modules for testing specific applications or can extend an existing module to add custom functionality.

- **Graphical/Tree View Scripting Interface** - The OpenScript Tree View scripting interface provides a graphical representation of the test script. Multiple script windows can actually be open at the same time. Within each script window, the Tree View is broken down into three main script sections:

  - Initialize: For script commands that only execute once on the first iteration
  - Run: Main body of the script for commands that will run on every iteration

  - Finish: For script commands that only execute once on the last iteration

  Within each section, script Steps and Navigation nodes can be created automatically during script recording or manually through the Tree View user interface. Additional script commands will also be represented as nodes in Tree View including test cases, data inputs, log messages, etc. Each Tree View node has a corresponding representation in the Java Code View.

- **Programming/Code View Scripting Interface** - The OpenScript Java Code View scripting interface provides a Java representation of the test script. This view provides full access to Eclipse IDE for creating, editing & debugging script code. Script commands in Java are mapped to a corresponding representation in the Tree View. Users can edit their script in either the code or tree view and changes will be automatically reflected in both views.
- **Properties View & Results View** - The OpenScript Properties View allows users to view detailed properties for selected script nodes in the Tree View. The Results View shows detailed step-by-step results of script playback which are linked to the OpenScript display window.
- **Data Banking** - OpenScript allows users to parameterize script data inputs to perform data driven testing using Data Banking. Users can select any data inputs for their script and then substitute a variable to drive the input from an external file during playback. Multiple Data Bank files can be attached to a single script and users can specify how OpenScript assigns data during script playback.
- **Correlation** - The OpenScript Correlation interface allows users to create correlation libraries to automatically parameterize dynamic requests during playback. Correlation libraries contain rules for automatically handling dynamic request parameters such as urls, query strings and post data for the load testing modules.
- **OpenScript Preferences** - The OpenScript Preferences interface is where users specify settings to control script recording, script playback, correlation and general preferences for the OpenScript Workbench.
- **Multi-User Execution** - launch more than one OpenScript instance under separate named Windows user accounts. Playback for multiple scripts is supported using any of the following:
  - OpenScript Playback button
  - Command-Line Interface
  - Oracle Load Testing
  - Oracle Test Manager

1.3 About Oracle Load Testing

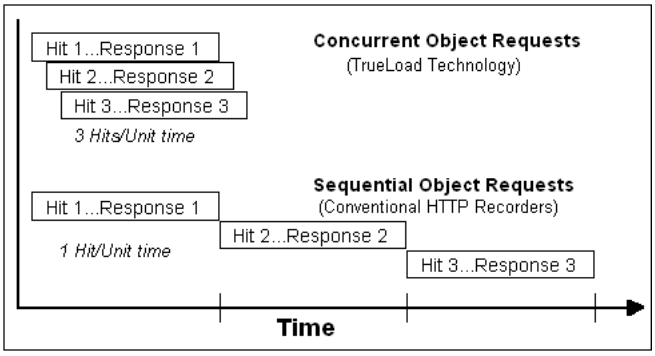
Oracle Load Testing provides an easy and accurate way to test the scalability of your e-Business applications. Oracle Load Testing emulates thousands of virtual users accessing your site simultaneously, and measures the effect of the load on application performance.

Figure 1–3 Oracle Load Testing Virtual Users View

VU-ID	Profile	Status	Iterations	Failed	Last Run Time	Current Step	System	Data Bank	Current Error	Previous Error
1	HTTPTutor1	Running	28	28	0.953	[1] Avitek Medical Records Application	OLT Server			
2	HTTPTutor2	Iteration delay	24	24	0.953	[1] Avitek Medical Records Application	OLT Server	avitek: [fred@golf.com, weblogic]		
3	HTTPTutor1	Running	23	23	0.953	[1] Avitek Medical Records Application	OLT Server			
4	HTTPTutor1	Running	18	18	0.953	[1] Avitek Medical Records Application	OLT Server			
5	HTTPTutor2	Iteration delay	16	16	0.953	[1] Avitek Medical Records Application	OLT Server	avitek: [larry@bball.com, weblogic]		
6	HTTPTutor1	Running	15	15	1.062	[1] Avitek Medical Records Application	OLT Server			
7	HTTPTutor1	Iteration delay	13	13	0.954	[1] Avitek Medical Records Application	OLT Server			
8	HTTPTutor2	Running	10	10	0.953	[1] Avitek Medical Records Application	OLT Server	avitek: [page@fish.com, weblogic]		
9	HTTPTutor1	Running	7	7	1.063	[1] Avitek Medical Records Application	OLT Server			

Oracle Application Testing Suite TrueLoad Technology ensures that your tests will closely correlate with real user-load so you can confidently use results from Oracle Load Testing to help make key decisions about your system's architecture, tuning, and hosting alternatives.

Figure 1–4 Concurrent Object Requests vs. Sequential Object Requests



1.3.1 Oracle Load Testing Feature Highlights

Oracle Load Testing offers the following advantages for Web-based application load testing:

- **Trueload Technology** - accurately emulates multi-threaded browser requests and automatically validates server responses for test results that closely correlate with real user testing.
- **Reusable Scripts** - uses the scripts created with Oracle OpenScript to emulate hundreds or thousands of virtual users.
- **Interactive What-If Analysis and Virtual User Display** - you can change the number and type of user on-the-fly to try "what-if" scenarios as you vary the

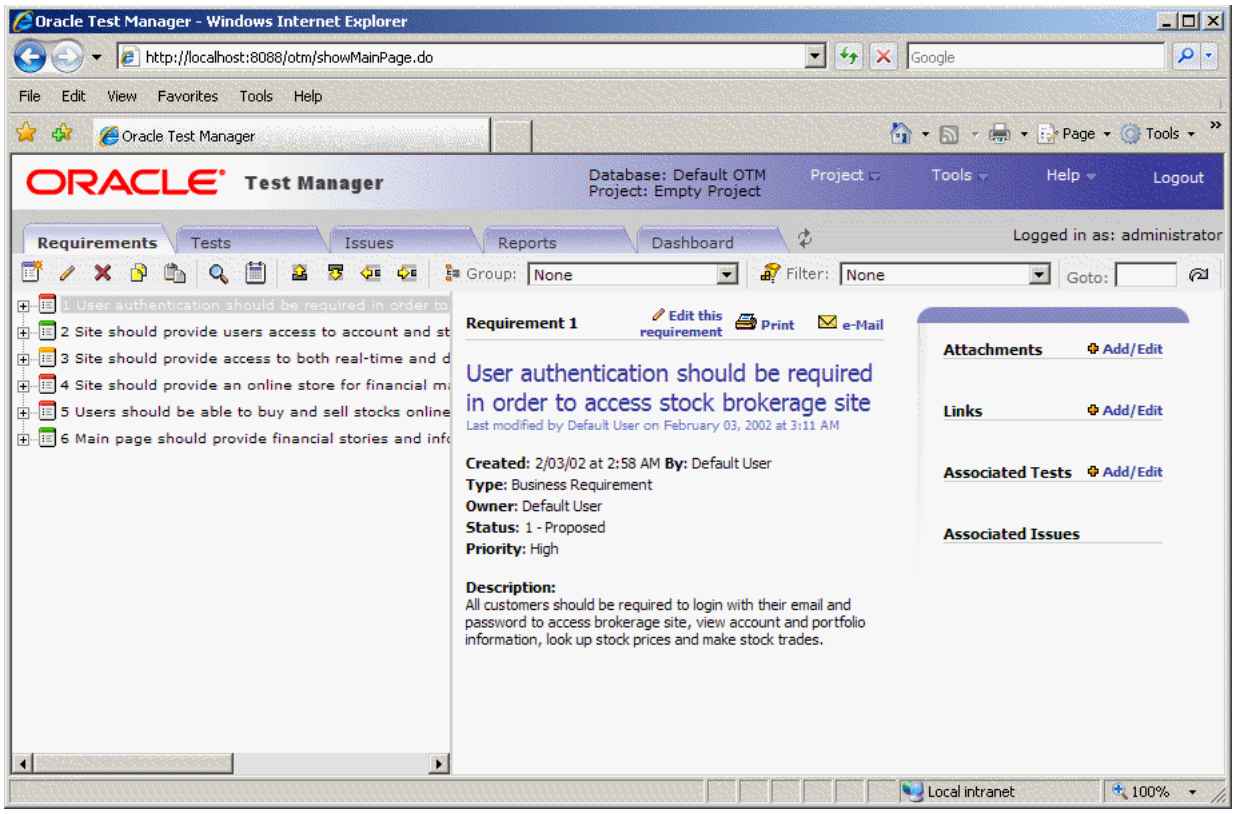
loading conditions or application settings. You can even view the actual pages seen by virtual users to aid in debugging.

- **Real-Time Graphs and Reports** - you can view real-time reports and graphs that include response time, error rates, number of users, and statistics such as hits per second, pages per second, etc.
- **Single Point of Control with Distributed Agents** - virtual users can be simulated by a single server or distributed amongst multiple servers located anywhere on a LAN or WAN.
- **Scenario Manager and Autopilot** - define any number of custom load scenarios by simply pointing and clicking on the names of the pre-recorded scripts and then specifying how many virtual users of each type you wish to run, and how you would like them to ramp up.
- **Post-run Analysis** - performance data can be accumulated at varying levels of granularity including profiles, scripts, groups of pages, individual pages, and objects on pages. Oracle Load Testing provides a comprehensive set of graphs and reports, and can also export data to external programs such as Microsoft Excel for further analysis.
- **Server-side monitoring with ServerStats** - server performance can be monitored for a variety of server-side application, database, system, and Web server statistics. You can configure ServerStats to display real-time performance statistics for the various hosts and services available from the server such as, percentage of CPU usage, memory usage, Web server statistics, etc.

1.4 About Oracle Test Manager

Oracle Test Manager is an easy to use tool that allows you to organize and manage your overall testing process. It provides a single unified platform for sharing information among team members.

Figure 1–5 Oracle Test Manager Main Window



Oracle Test Manager lets you create projects that group together and organize test scripts, requirements that need to be tested, and issues resulting from the tests. Once created, you can indicate the relationships among these items, allowing you to quickly and easily find all information pertaining to a particular test script, requirement, or issue.

1.4.1 Oracle Test Manager Feature Highlights

Oracle Test Manager offers the following features and advantages for integrated requirements management and defect tracking for both manual and automated tests:

Requirements Management - provides the ability to define and manage requirements for a specific project. You can specify details for each requirement, track the status of each requirement, and associate requirements with test cases to ensure testing coverage.

Test Planning and Management - provides the ability to define and manage a test plan that incorporates both manual and automated test cases. You can store Oracle Application Testing Suite scripts in the database, automatically execute scripts in Oracle OpenScript from the test plan interface, and automatically store the test results. You can also associate requirements to test cases to ensure testing coverage, and associate test cases with issues so they can be reproduced and to keep track of how the issues were identified.

Defect Tracking - provides the ability to create and manage defects, referred to as issues, for a specific project. You can associate test cases with issues so they can be reproduced and to keep track of how the issues were identified.

Integration with Oracle Application Testing Suite - seamlessly integrates with Oracle Application Testing Suite test solutions, providing the ability to automatically launch and execute Oracle OpenScript scripts for functional and regression testing as well as retrieve and archive the results. You can also launch third party products.

Reporting - generates reports in standard HTML format for managing the overall testing process. You can report on requirements, tests, and issues.

Administration - provides and administration tool for entering and managing user accounts, project permissions, and general tool preferences.

Custom Fields - provides the ability to add custom fields to the database for recording data specific to your projects.

Database Repository - provides the ability to store test assets including test scripts, results, attachments, requirements, test plans, and defects in a common database.

1.5 Oracle Application Testing Suite Database Configuration

The Oracle Application Testing Suite Database Configuration utility lets you add database connections for Oracle Load Testing and Oracle Test Manager. The Oracle Application Testing Suite installation includes a WebLogic web server and the Oracle 10*g* Express Edition Database by default. The Oracle Application Testing Suite Database Configuration utility can be used set the current database or to connect to databases other than the default.

1.5.1 Database Configuration Feature Highlights

The Database configuration utility provide the following features:

- **Add, Update, Delete Database Connections** - provides a convenient way to manage database connections for Oracle Load Testing and Oracle Test Manager.
- **Create Schemas and Tables** - automatically uses existing schemas or creates schemas and tables for additional databases.

2

Oracle Application Testing Suite Basics

This chapter explains how to get started using Oracle Application Testing Suite. It explains how to install and start the applications, and the features of the main windows.

2.1 Installing and Starting Oracle Application Testing Suite

To install Oracle Application Testing Suite:

1. Go to:
   http://www.oracle.com/technology/software/products/app-testing/index.html.

   Note: See the Oracle Application Testing Suite Release Notes for additional information about the download files and the installation procedure.

2. Download the Oracle Application Testing Suite product from the Oracle Web site and save it to a temporary directory on your hard disk.
3. Unzip the download file and then run setup.bat to install Oracle Application Testing Suite.
4. Follow the setup instructions to install the Oracle Application Testing Suite and OpenScript.

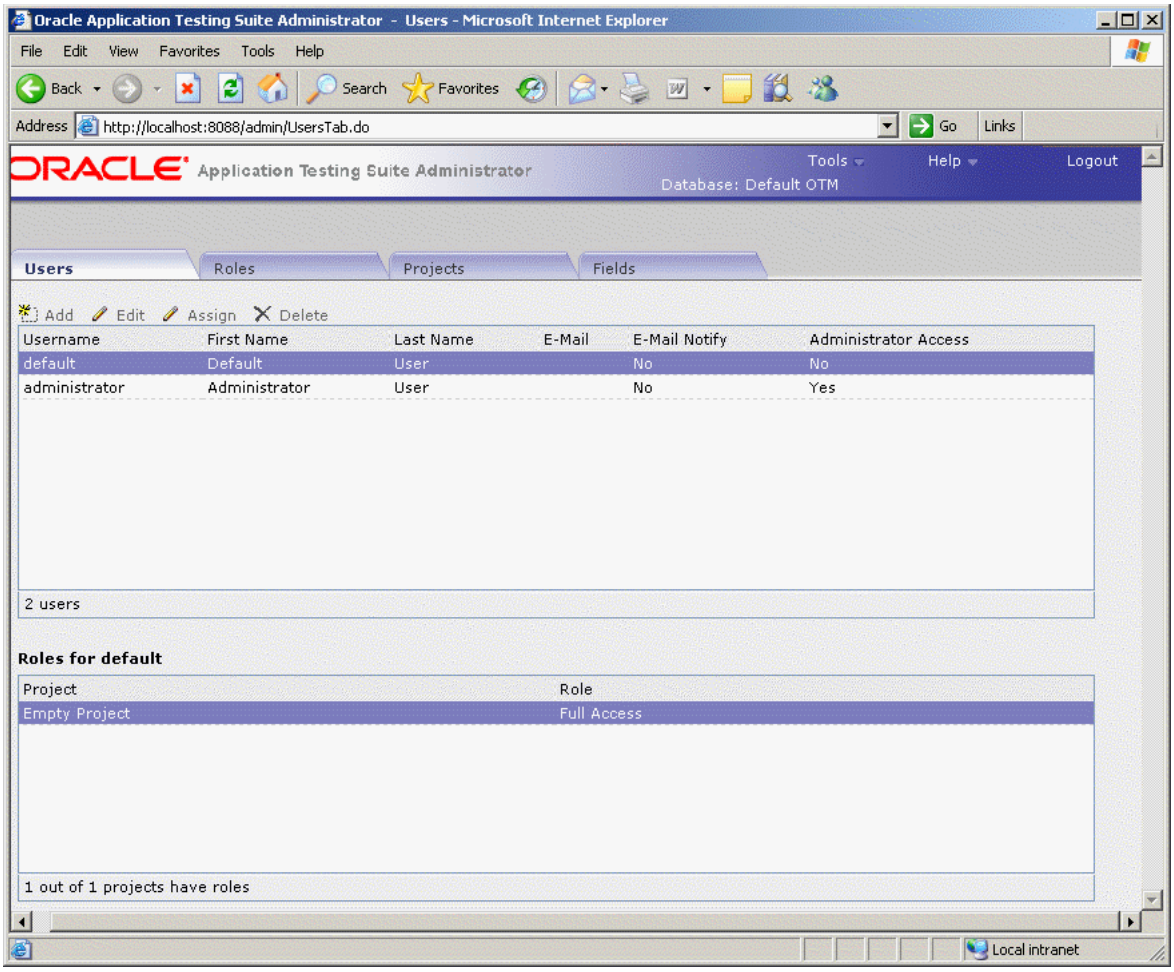
   During the Oracle Application Testing Suite installation, you will be required to enter a default password to be used with Oracle Application Testing Suite products. *Remember this password*. It will be required to log in to the Administrator, Oracle Load Testing, and Oracle Test Manager.
5. Select applications from the **Oracle Application Testing Suite** start menu to start the user interface for specific products.

The installation creates a default Administrator user name in the Oracle Application Testing Suite database. The first time you log into the Administrator, Oracle Load Testing, or Oracle Test Manager, enter the username **administrator** and the password you defined during the installation. You can use the Oracle Application Testing Suite Administrator to change the default users and customize the usernames and passwords for Oracle Application Testing Suite users.

2.2 Oracle Application Testing Suite Administrator Main Window Features

The Oracle Application Testing Suite Administrator is where you customize user access for Oracle Load Testing and Oracle Test Manager. For Oracle Test Manager, you can also customize roles, projects, and fields.

Figure 2–1 Oracle Application Testing Suite Administrator Main Window



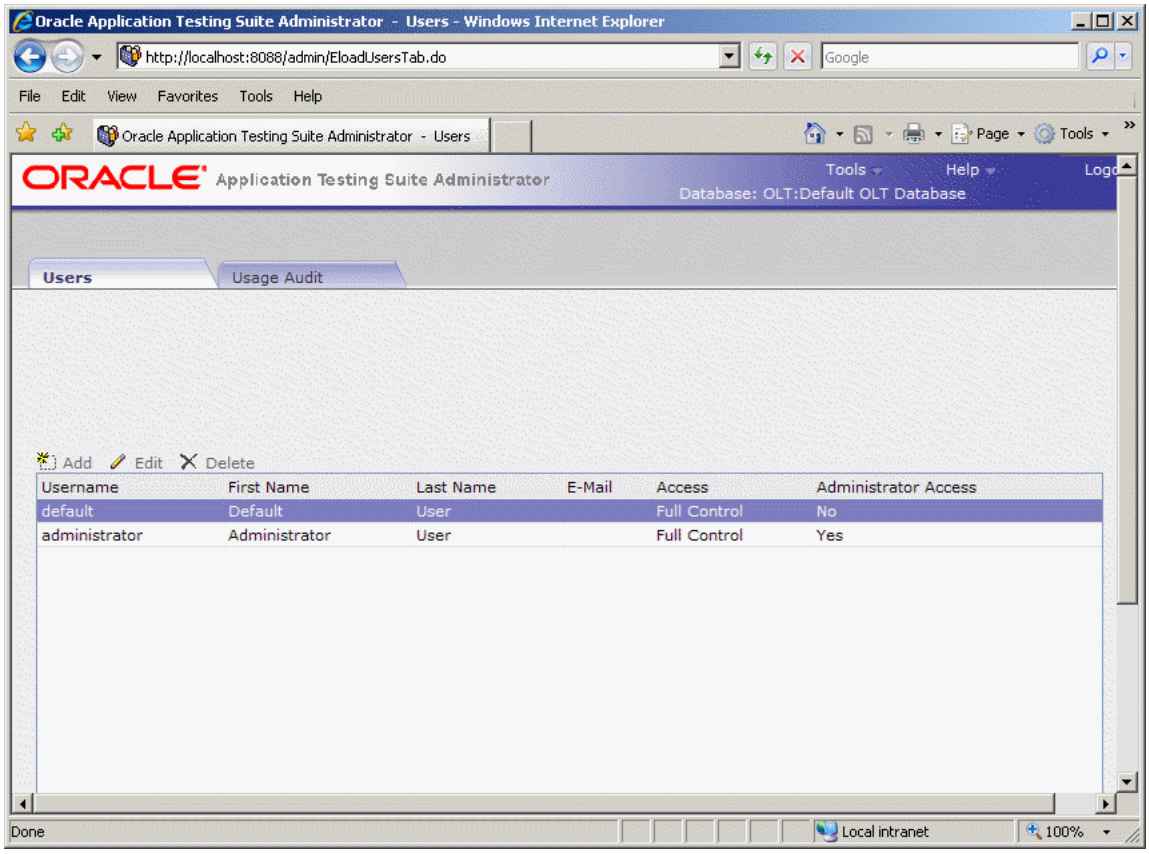
When you log into the Administrator and select an Oracle Load Testing database, only the **Users** and **Usage Audit** tabs appear. When you log into the Administrator and select an Oracle Test Manager database, the **Users**, **Roles**, **Projects**, and **Fields** tabs appear.

2.2.1 Users Tab

The Users tab is where you add, delete, and configure users for Oracle Load Testing and Oracle Test Manager for Applications.

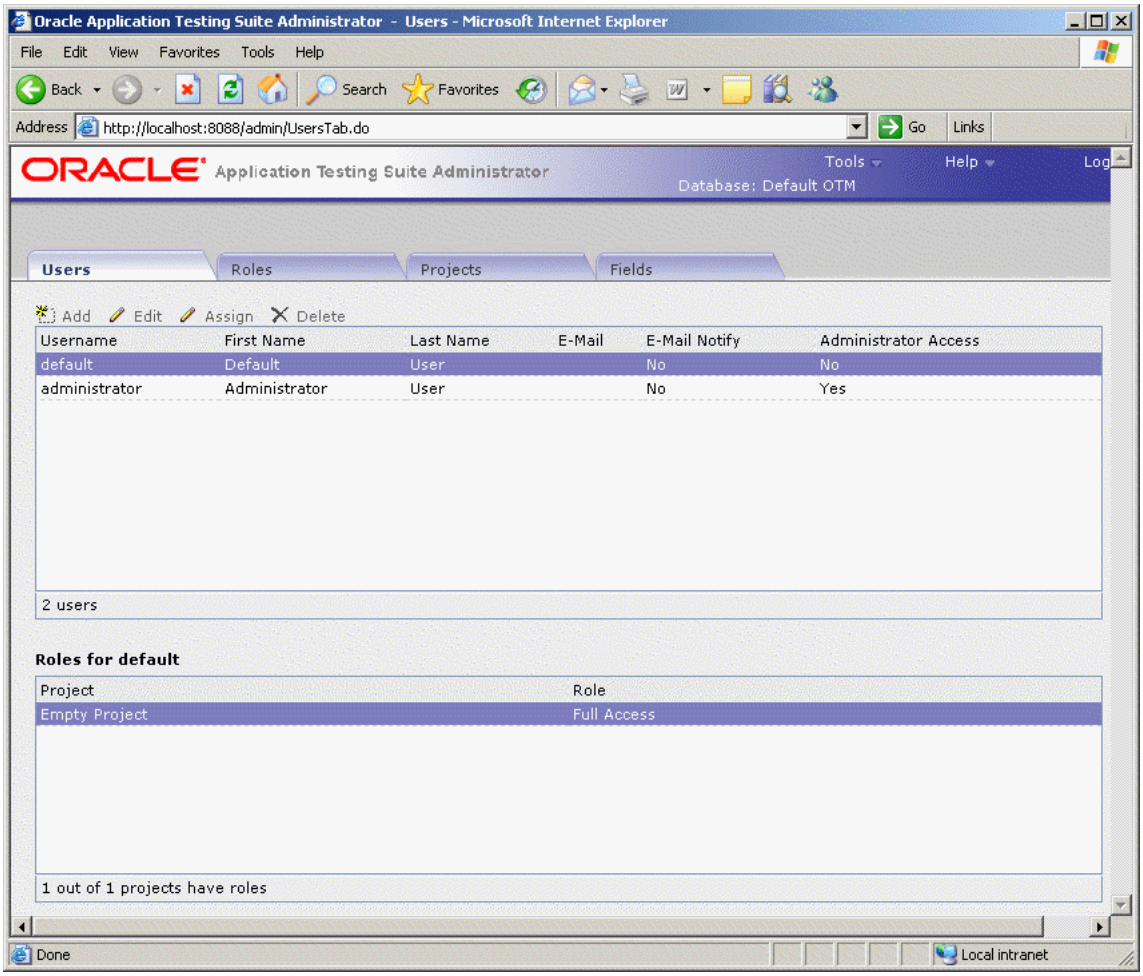
For Oracle Load Testing databases, the Users tab lets you add, edit, and delete users.

Figure 2–2 Users Tab for Oracle Load Testing Users



For Oracle Test Manager databases, the Users tab lets you add, edit, and delete users, and assign specific projects to users.

Figure 2–3 Users Tab for Oracle Test Manager Users



The Users tabs can have the following options:

Add - displays the Add User dialog box for adding a new user.

Edit - displays the Edit User dialog box for the selected user. You can change the user's name, username, or password.

Assign - displays the Edit Role dialog box for assigning roles to the selected user for the selected projects.

Delete - deletes the selected user.

Username - displays the name the user will use to log on. Click the column header to sort users in ascending order by this field.

First Name - displays the user's first name. Click the column header to sort users in ascending order by this field.

Last Name - displays the user's last name. Click the column header to sort users in ascending order by this field.

E-mail - displays the user's email address. When **Enable E-mail notification** is selected for the user, the user will receive email notifications when items are created, or when the owner or assigned to field is changed for issues. Click the column header to sort users in ascending order by this field.

E-mail Notify - indicates whether the user will receive email notifications. Click the column header to sort users in ascending order by this field.

Administrator Access - indicates whether the user can access the Oracle Test Manager Administrator.

Active - this column is only displayed when named user licenses are being used. Indicates whether the user is active, that is, allowed to log in using a named user license.

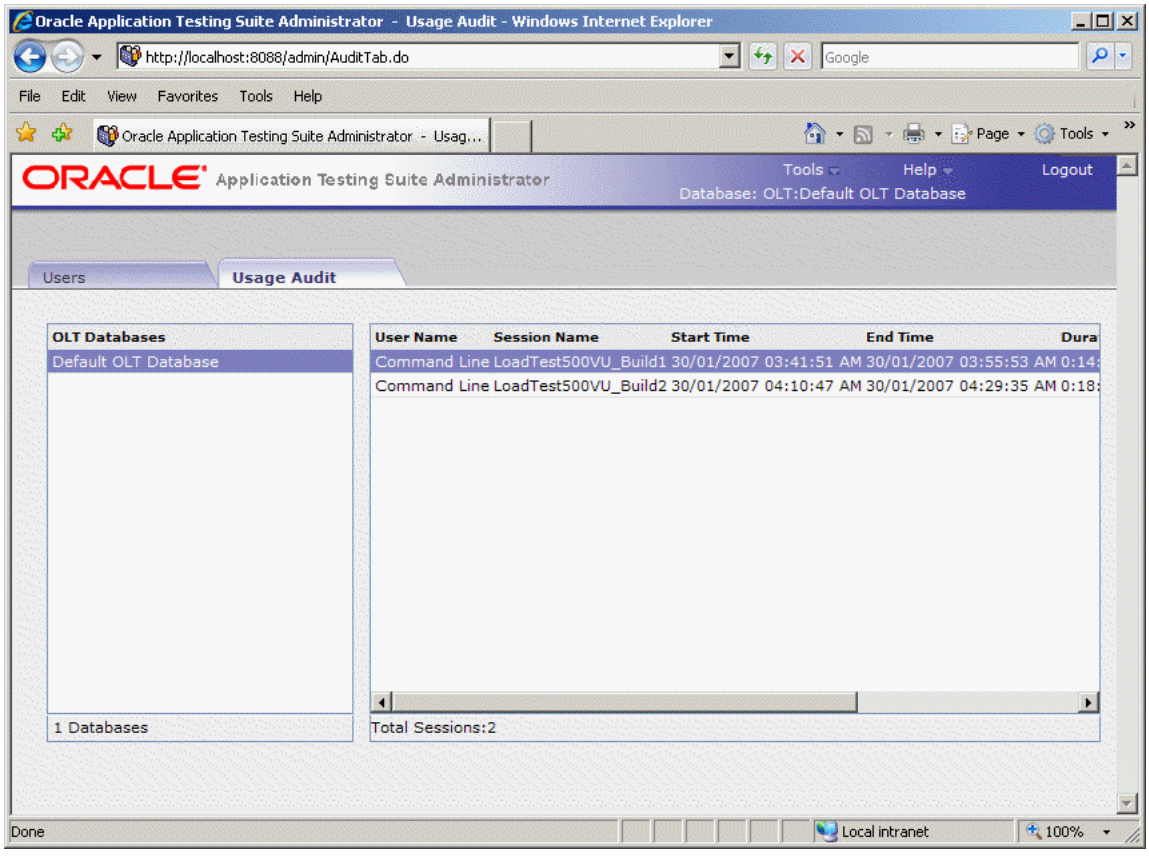
Roles for <user> - displays the roles assigned to the selected user for each project in the database.

- **Project** - displays a list of the projects in the database.
- **Role** - displays the role of the selected user for the project.

2.2.2 Usage Audit Tab

The Usage Audit tab is where you review and audit the load testing sessions stored in the Oracle Load Testing database.

Figure 2–4 Usage Audit Tab for Oracle Load Testing Users



The Usage Audit tab has the following options:

OLT Databases - lists the installed Oracle Load Testing Databases available for auditing.

User Name -shows the name of the user who ran the load test. "Anonymous" indicates the login feature was disabled for the instance of Oracle Load Testing that ran the test

and there is no username associated with the test. "Command Line" indicates the load test ran from the command line interface.

Session Name - shows the name of the load testing session.

Start Time - shows the start date and time for the load testing session.

End Time - shows the end date and time for the load testing session.

Duration (HH:MM:SS) - shows the duration of the load testing session in hours, minutes, and seconds.

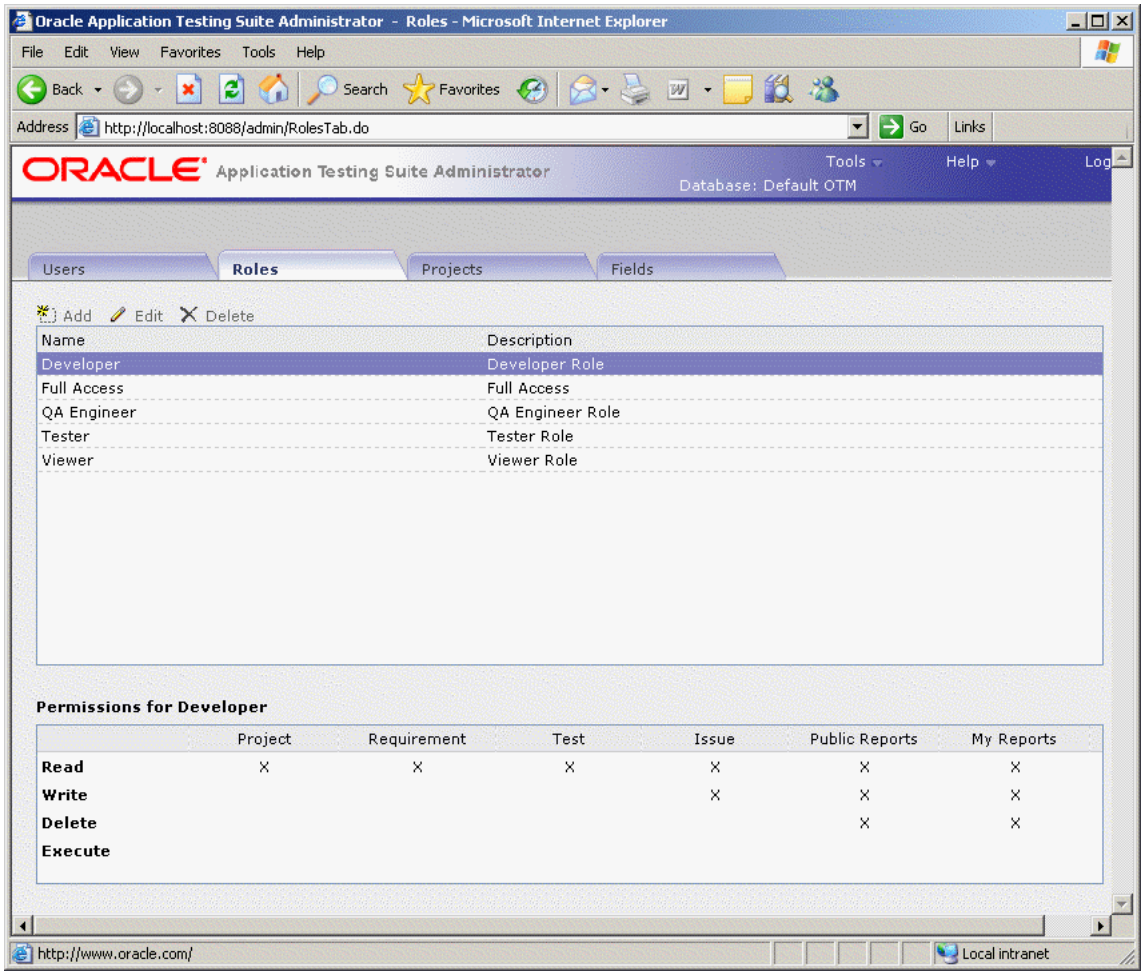
Machine Name - shows the name of the machine on which the load testing session was run.

Max VU Count - shows the maximum count of Virtual Users that were run for the load testing session.

2.2.3 Roles Tab

The Roles tab is where you configure roles. Roles determine the read, write, delete, and execute permissions for users in projects. Once roles are created, you assign them to users for each project that you want them to have access to. A user's role can differ from project to project. Click **Assign** on either the Projects tab or Users tab to assign roles.

Figure 2–5 Roles Tab for Oracle Test Manager Users



The Roles tab has the following options:

Add - displays the Add Role dialog box for adding a role.

Edit - displays the Edit Role dialog box for editing the selected role.

Delete - deletes the selected role. If a role is in use, you will be asked to assign another role to users assigned to this role.

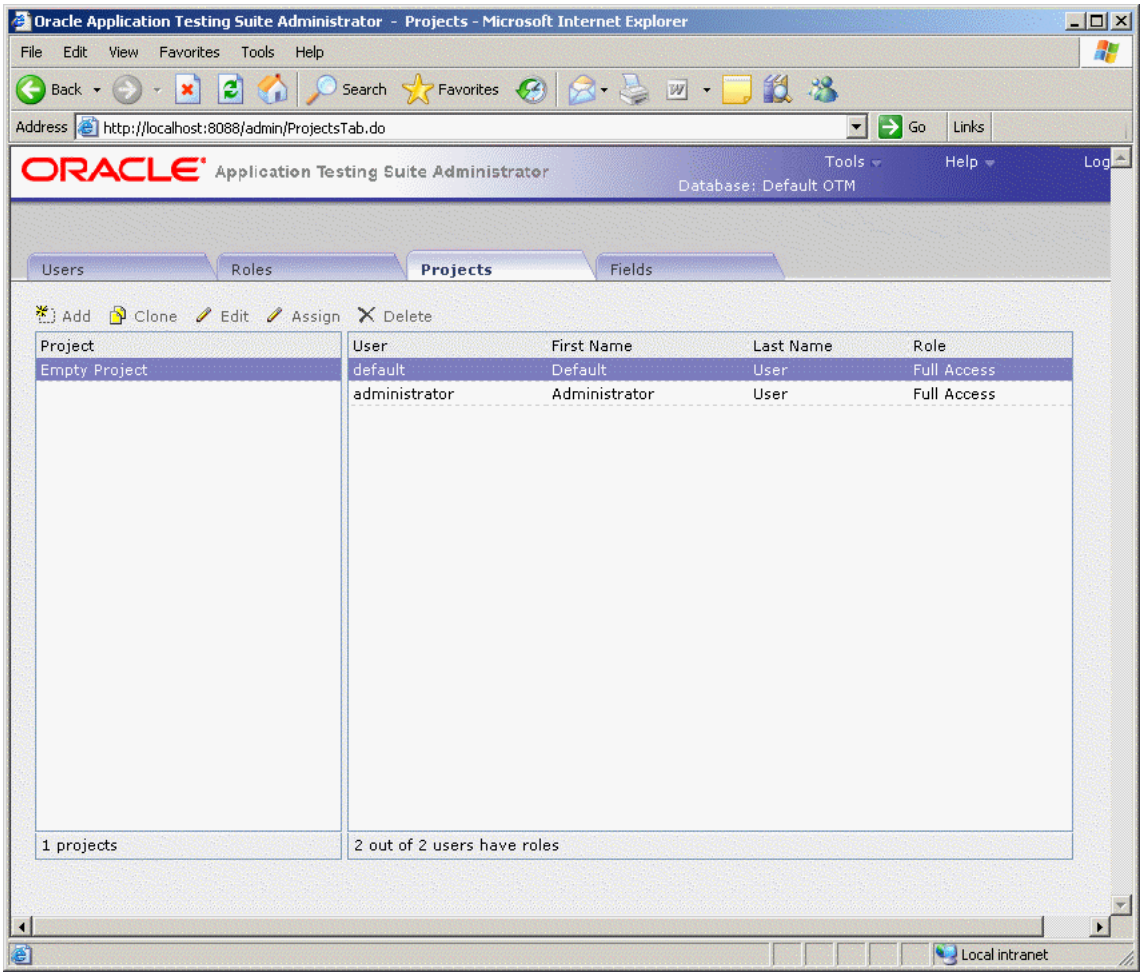
Permissions for <role> - displays the read, write, delete, and execute permissions for this role for projects, requirements, tests, and issues.

- **Project** - displays the read, write, and delete permissions for projects for users assigned to this role.
- **Requirement**- displays the read, write, and delete permissions for requirements for users assigned to this role.
- **Test** - displays the read, write, delete, and execute permissions for tests for users assigned to this role.
- **Issue** - displays the read, write, and delete permissions for issues for users assigned to this role.
- **Public Reports** - displays the read, write, and delete permissions for public reports assigned to this role.
- **My Reports** - displays the read, write, and delete permissions for reports assigned to this role.

2.2.4 Projects Tab

The Projects tab is where you maintain projects.

Figure 2–6 Projects Tab for Oracle Test Manager Users



The Projects tab has the following options:

Add- displays the Add Project dialog box for adding a new project.

Clone - displays the Clone Project dialog box for duplicating the selected project. When you clone a project, user roles are the same as the original project.

Edit - displays the Edit Project dialog box for changing the name of the selected project.

Assign - displays the Edit Role dialog box for assigning roles to users for the selected project(s).

Delete - deletes the selected project.

Project - displays a list of projects in the database.

User - displays the users in the database.

First Name - displays the user's first name.

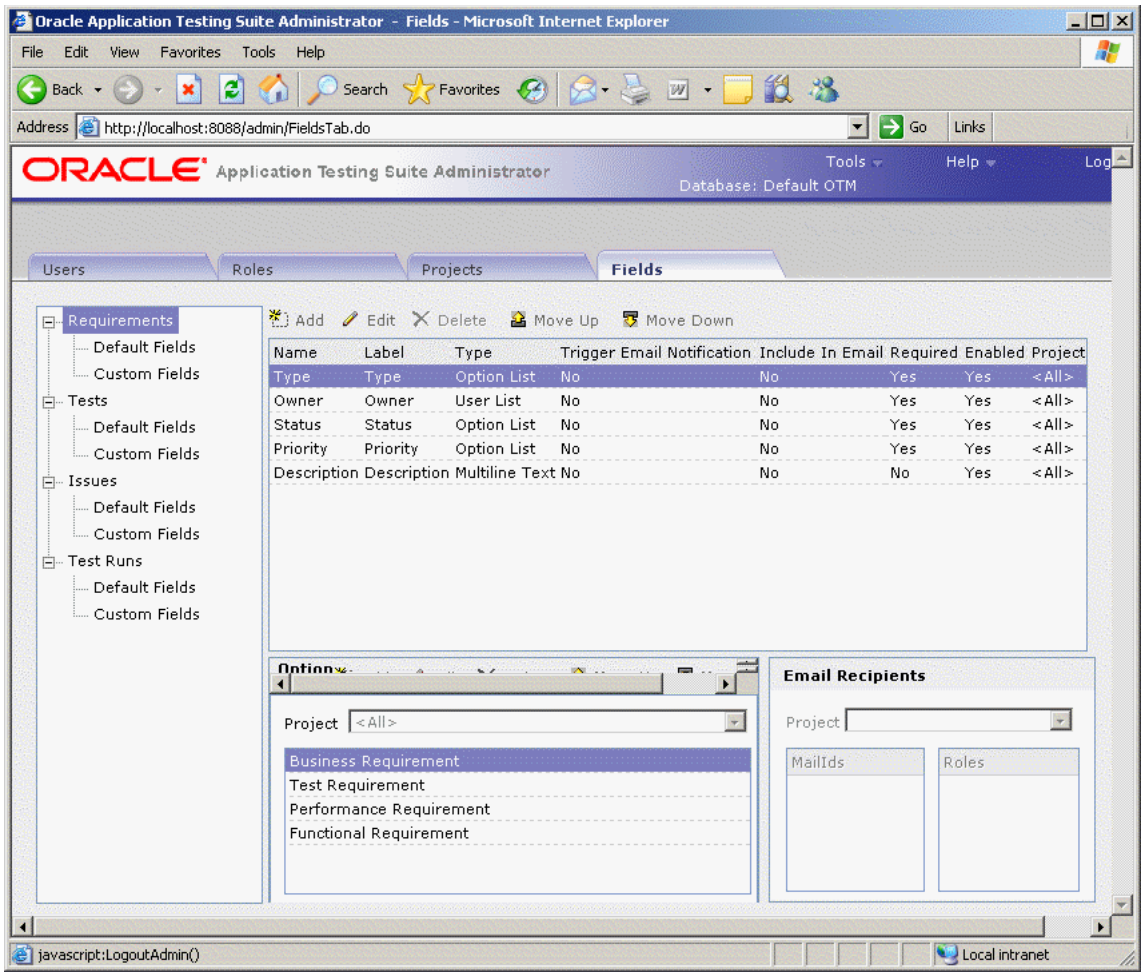
Last Name - displays the user's last name.

Role - displays the users' roles in the selected database.

2.2.5 Fields Tab

The Fields tab is where you customize both default and custom fields. These fields are used in Oracle Test Manager for maintaining details about requirements, tests, issues, and test runs.

Figure 2–7 Fields Tab for Oracle Test Manager Users



The Fields tab has the following options:

<Field List> - lists the categories and types of fields that you can customize. The categories are:

- **Requirements** - fields that pertain to requirements that appear in the Add/Edit Requirements dialog boxes and are displayed in the right pane.
- **Tests** - fields that pertain to tests that appear in the Add/Edit Tests dialog boxes and are displayed in the right pane.
- **Issues** - fields that pertain to issues that appear in the Add/Edit Issues dialog boxes and are displayed in the right pane.
- **Test Runs** - fields that pertain to test runs that appear in the Run Test dialog box and are displayed in the Result Parameters section of the right pane when you click the run date of a test in the Run History section.

Each category has two types of fields:

- **Default Fields** - these are the fields that are shipped with the product. You can add and delete options and change the labels.
- **Custom Fields** - these are user-created fields. These fields are used for entering information in Oracle Test Manager. They are added to the input and edit dialog boxes for requirements, tests, and issues. They are displayed in the right pane with the default fields, and they can be used for grouping and reporting.

Name - displays the field name.

Label - displays the label displayed in Oracle Test Manager.

Type - displays the type of field. The options are:

- **Option List** - lets you select an option from a list.
- **Option List/Text** - lets you select an option from a list or enter text.
- **User List** - creates a list of the users in the database and lets you select one.
- **Text** - lets you enter one line of text.
- **Multiline Text** - lets you enter multiple lines of text.
- **Multiline/Append** - creates a multiline text field and when editing, lets you choose to append new text to existing text. If you choose to append, the date and user name are automatically added.
- **Heading** - lets you create a heading for grouping custom fields. The heading is for display purposes only in the right-hand pane of Oracle Test Manager.

Trigger Email Notification - indicates whether an email will be sent to the configured recipients when this field changes.

Include In Email - indicates whether the field should be included in email.

Required - indicates whether the field is required, that is, data must be entered when the requirement, test, or issue is created.

Enabled - indicates whether the field is being used in Oracle Test Manager.

Project - indicates the project to which this field applies.

Add - displays the Add Field dialog box for adding a custom field.

Edit - displays the Edit Field dialog box for the selected custom field.

Delete - deletes the selected custom field.

Move Up - moves the selected field up one place.

Move Down - moves the selected field down one place.

Option Lists - lets you maintain the options for the selected field if the field is an option list type of field.

- **Add** - displays the Add Option dialog box for adding an option to the selected field.
- **Edit** - displays the Edit Option dialog box for renaming the selected option.
- **Delete** - deletes the selected option.
- **Move Up** - moves the selected option up one place.
- **Move Down** - moves the selected option down one place.
- **Project** - lets you select the project to which the options apply. This field is only available when you select the **Project Specific Options** check box in the Add

Custom field dialog box for this field. When this check box is selected you can add options specific to each project; otherwise, all projects will have the same options.

Email Recipients - lists the roles and/or email addresses of the people that will receive an email when the selected field changes.

- **Project** - select the project for which you want to display the email recipients.
- **MailIds** - lists the email addresses to which email will be sent when the field changes.
- **Roles** - lists the roles of the email recipients to which email will be sent when the field changes.

When you select the category in the left pane, the associated fields are displayed in the top right pane. When you select the field in the top right pane, its associated options are displayed in the Option List.

2.3 Oracle OpenScript Main Window Features

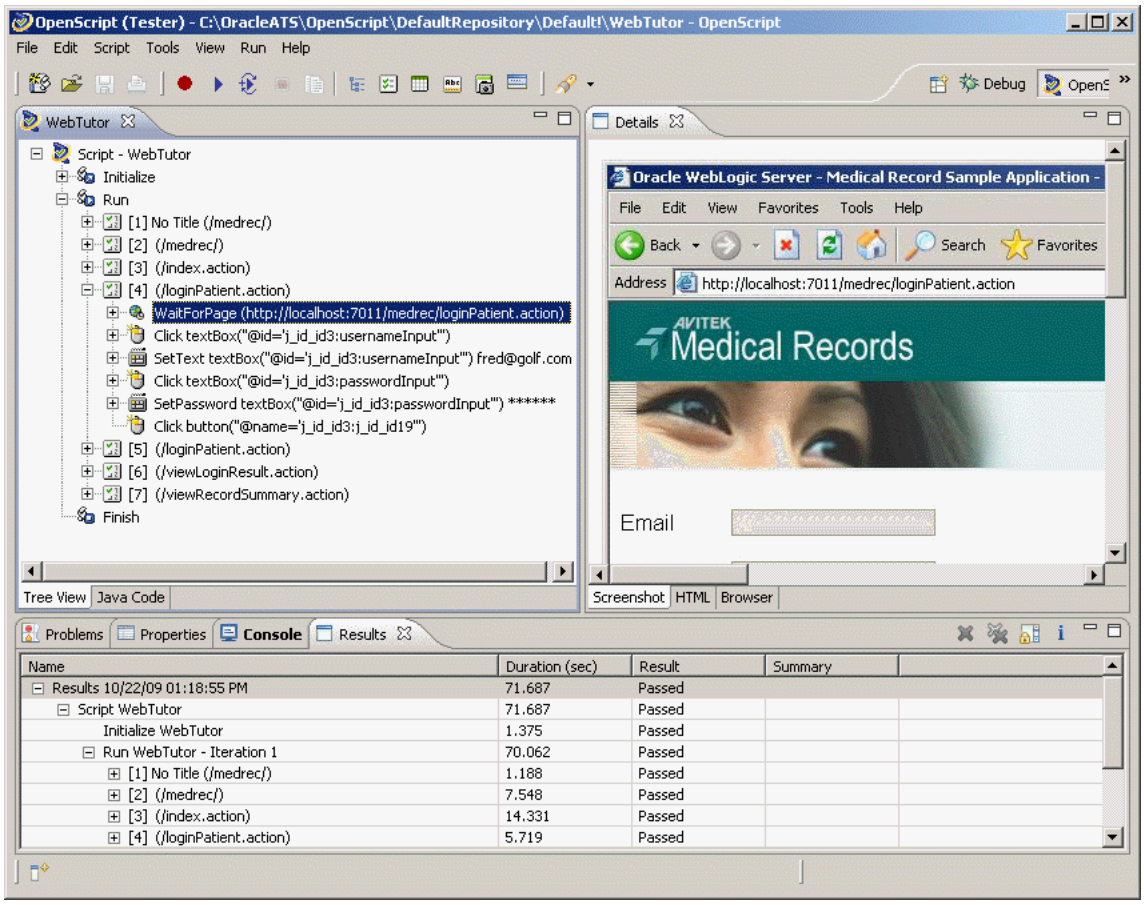
The Oracle OpenScript main window is where you develop the scripts used for functional/regression testing and load/performance testing of your Web site or application. Specific types of scripts you develop using Oracle OpenScript can also be used by Oracle Load Testing or Oracle Test Manager.

Scripts represent a sequence of actions and tests performed on a Web site or application. Scripts are used by Oracle OpenScript for regression testing and Oracle Load Testing for performance (load and scalability) testing.

The Oracle OpenScript main window consists of the menu bar, toolbar, and the scripting workbench in the Eclipse Integrated Development Environment (IDE). The Workbench provides an Eclipse-based scripting platform where you can create and run your automated test scripts. Users can use the Tree View graphical scripting interface for creating and editing scripts through the UI. Users can also switch to the Java Code view programming interface and leverage the integrated Eclipse IDE for creating and editing their scripts programmatically.

The workbench includes a Tester Perspective and a Developer Perspective. The Tester Perspective provides a convenient way to record and edit scripts and view the playback results. The Developer Perspective provides advanced options for developers when creating and editing scripts using the advanced features of OpenScript and the Eclipse development platform.

Figure 2–8 Oracle OpenScript Main Window



The OpenScript Test Modules provide application-specific test automation capabilities. Each Test Module is custom built to test a specific application or protocol. OpenScript includes several functional and load testing modules for testing Web-based applications.

2.3.1 Script View

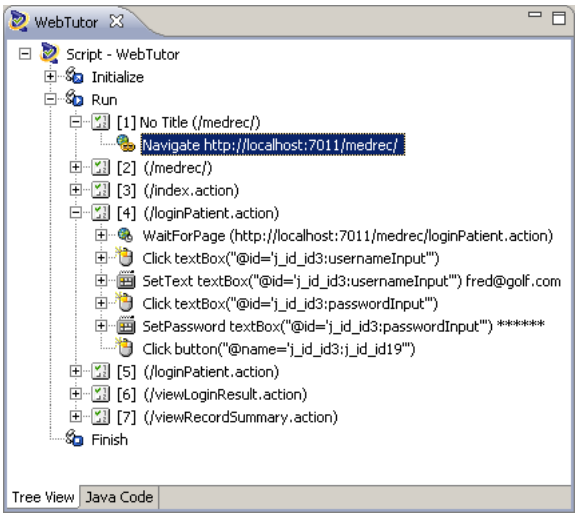
Shows the recorded script in two tabs: Tree View and Java Code. The Tree View tab shows the steps and pages and the Initialize, Run, and Finish nodes of each step using a graphical tree view. The Java Code tab shows the underlying Java code used for the script.

The script view is where you perform the majority of script editing actions. The Script view has the following tab views:

2.3.1.1 Tree View

The Tree View shows the script navigations and data as nodes in a collapsible tree view. The Tree View corresponds to the Java Code view. Any changes in the Tree View will be automatically updated in the Java Code view.

Figure 2–9 Script Tree View



The Tree View has the following standard nodes:

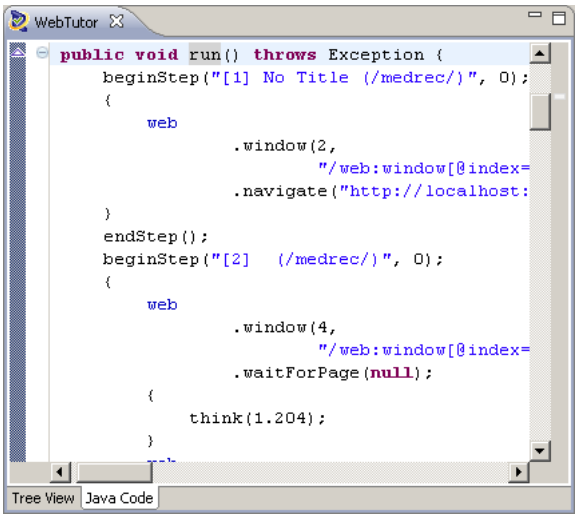
- Initialize - specifies script actions to perform once at the beginning of script playback.
- Run - specifies script actions to perform one or more times during script playback depending upon databanks or other custom programming.
- Finish - specifies script actions to perform once at the end of script playback.

Use the Record options and right-click shortcut menu to add options to script nodes or modify the properties of script nodes in the Tree View.

2.3.1.2 Java Code

The Java Code view shows the script navigations and data as Java programming code. The Java Code view corresponds to the Tree View. Any changes in the Code View will be automatically updated in the Tree View.

Figure 2–10 Script Java Code View

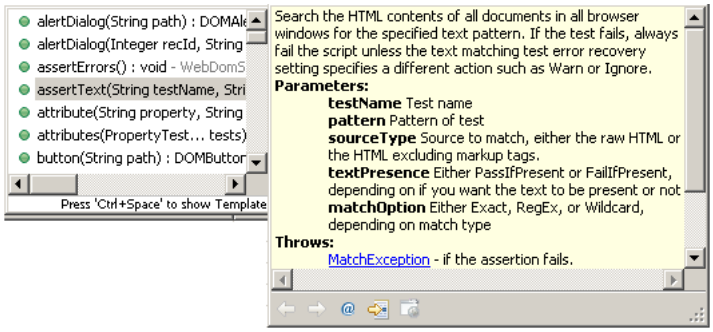


The Java Code view has the following standard procedures:

- `initialize()` - corresponds to the Initialize node of the Tree View and executes any custom code added once at the beginning of script playback.
- `run()` - corresponds to the Run node of the Tree View and executes recorded and custom code one or more times during script playback depending upon databanks or other custom programming.
- `finish()` - corresponds to the Finish node of the Tree View and executes any custom code added once at the end of script playback.

Use Ctrl-space to open an Intellisense window listing available procedures.

Figure 2–11 Java Code View Intellisense Window

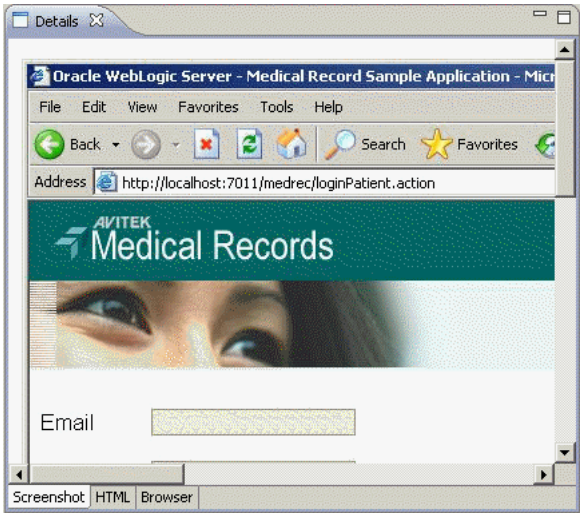


See the API Reference in the OpenScript Platform Reference help for additional programming information.

2.3.2 Details View

The Details view shows the content details for URL navigations added to the script.

Figure 2–12 Details View Showing Screenshot Tab View



The Details view may have the following tab views depending upon the selected script node and type of script:

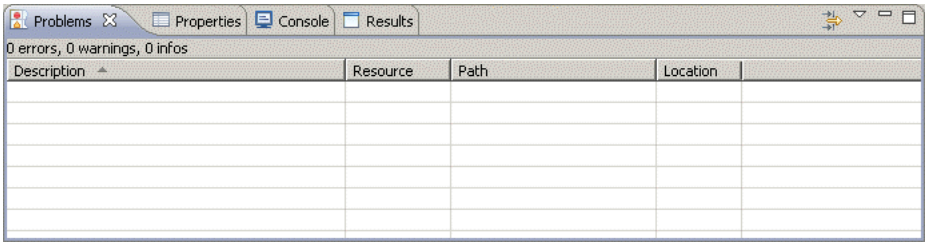
- ScreenShot - shows a screen capture of the web page.
- Browser - shows the Browser rendered page for the script navigation selected in the tree view.

- HTML - shows the HTML source for the script navigation selected in the tree view.
- Headers - shows the Request Header and Response Header source for the script navigation selected in the tree view.
- Comparison - shows the recorded and playback text for the Content, Request Header, or Response Header selected in the Compare list. The Comparison tab appears only after a script is played back and a navigation is selected in the Results View.
- Results Report - shows the results report for the script playback. The Results Report tab appears only after a script is played back and a navigation is selected in the Results View.

2.3.3 Problems View

The Problems view shows any problems in the script code that may produce errors or prevent compiling the script.

Figure 2–13 Problems View



The Problems view shows the following information:

- # error, # warnings, # infos - shows the number of errors, warning messages, and information messages in the problems view.
- Description - shows a description of the errors, warning messages, and information messages.
- Resource - shows the name of the resource file where the error, warning, or information message was generated.
- Path - shows the script name, workspace, and repository path where the resource file is located.
- Location - shows the location/line number where the error, warning, or information message was generated.

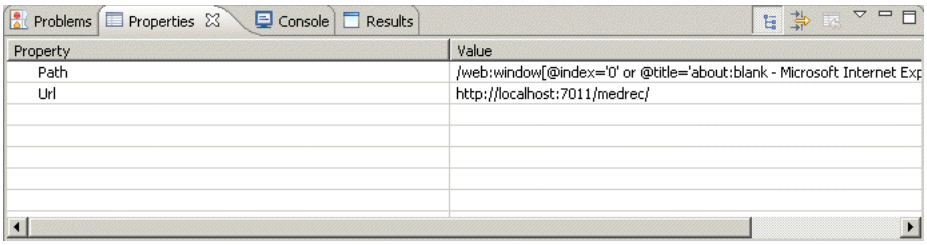
The following toolbar button is available in the Problems View:

- Configure the filters to be applied to this view - opens a dialog box for configuring the filters to apply to the Problems View.

2.3.4 Properties View

The Properties view shows the properties for the selected node in the script.

Figure 2–14 Properties View



The Properties view shows the following information:

- Property - shows the names of the properties for the script node. The properties vary depending upon which type of script node is selected.
- Value - shows the value of the script node properties. Property values can be edited in the properties view.

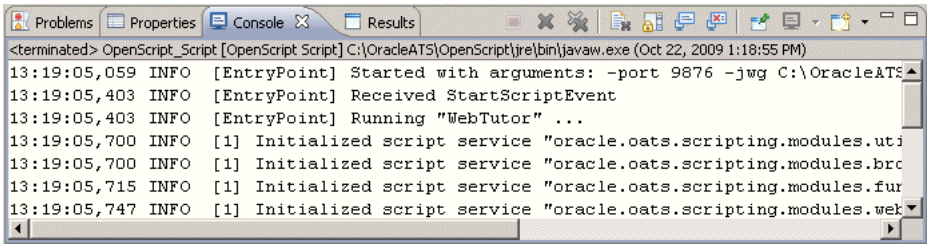
The following toolbar buttons are available in the Properties View:

- Show Categories - toggles the property categories.
- Show Advanced Properties - toggles the advanced properties.
- Restore Default Value - restores any changed property values to the default values.

2.3.5 Console View

The Console view shows the playback command output and status information for the script. Script log message also appear in the Console.

Figure 2–15 Console View

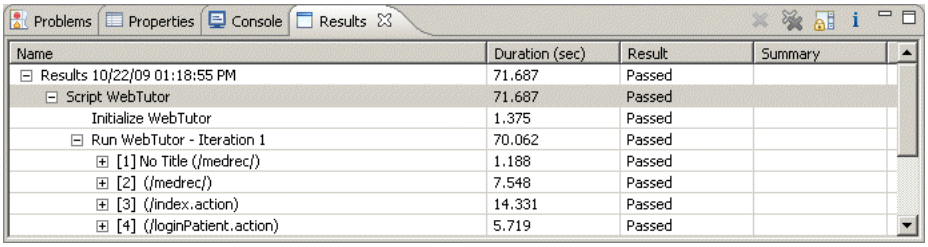


See the Process Console View topics in the reference section of the Java development user guide online help for additional information about console toolbar options.

2.3.6 Results View

The Results view shows the playback results for the script.

Figure 2–16 Results View



The Results view shows the following information:

- Name - shows the test date or navigation name.
- Duration - shows the playback time for the page navigations.
- Result - shows the playback result: Passed or Failed.
- Summary - shows the data values from the Data Bank that are passed to parameters or it shows failure descriptions.

The following toolbar buttons are available in the Result View:

- Delete Result - deletes the selected result row.
- Delete All Results - deletes all rows from the Results View.
- Scroll Lock - toggles scroll lock on and off for the Result View.
- Properties - opens the Properties for the selected result.

2.3.7 Other Views

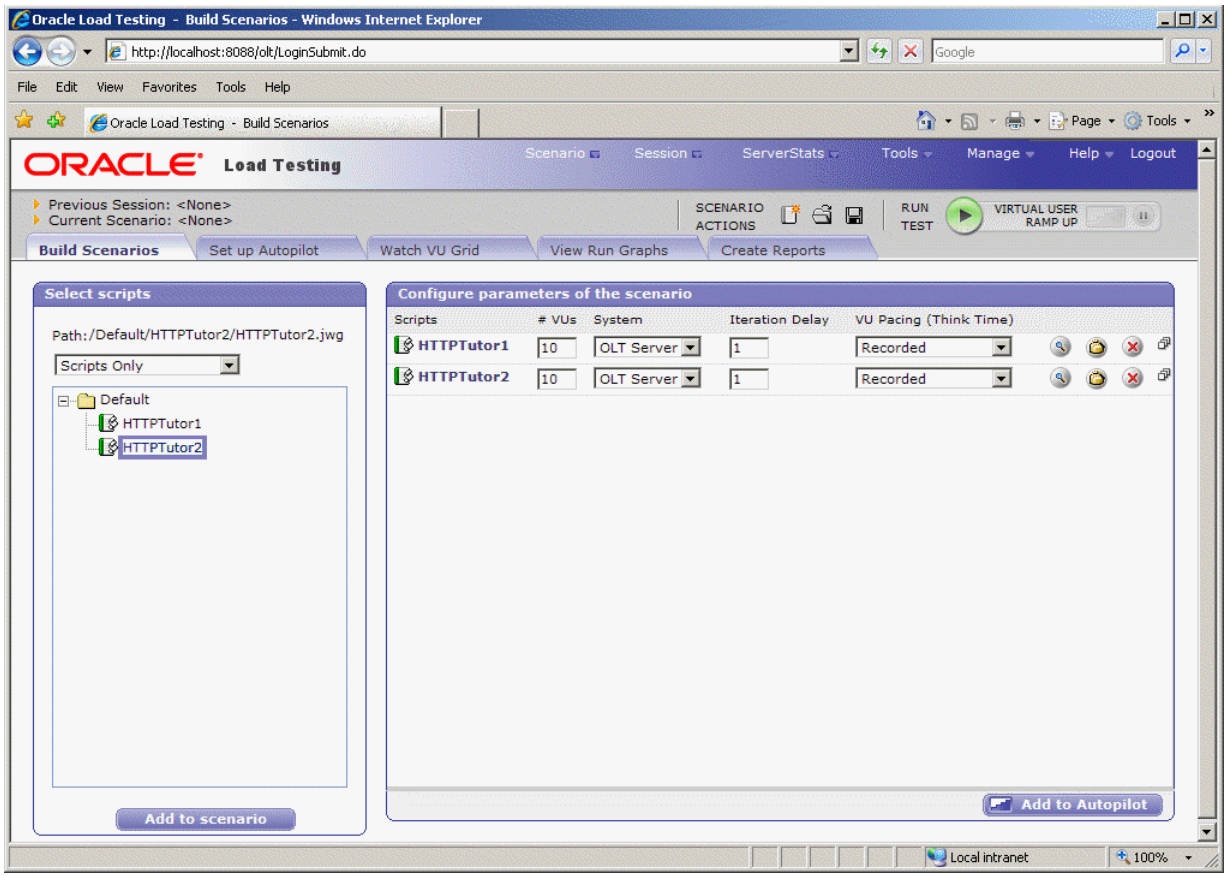
The OpenScript Developer Perspective includes several other views, such as Debug, Breakpoints, Navigator, and Package, that can be used by advanced users for specific testing and debugging purposes. See the *OpenScript User's Guide* for additional information.

2.4 Oracle Load Testing Main Window Features

The Oracle Load Testing main window is where you perform the majority of your load/performance testing activities. Oracle Load Testing uses scripts that you develop using Oracle OpenScript.

The main window consists of the menu bar, toolbar, and the controller tab dialogs.

Figure 2–17 Oracle Load Testing Main Window

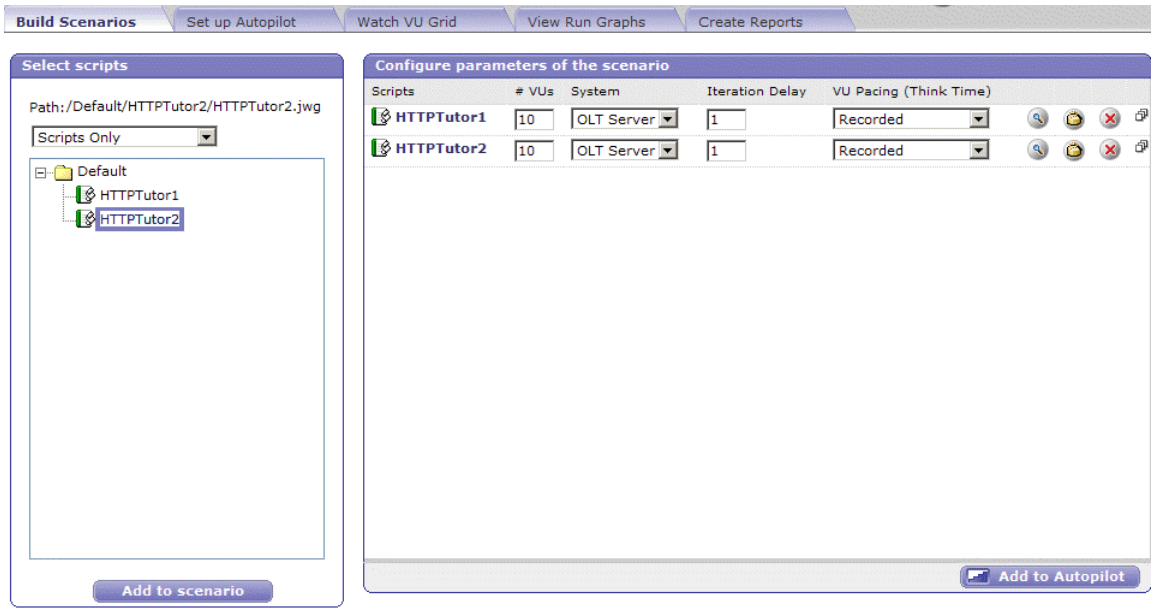


You can open the Oracle Load Testing application from the **Start** menu.

2.4.1 Build Scenario Tab

The Build Scenario tab is where you specify information about the virtual users to include in the load test and the attributes for each set of virtual users.

Figure 2–18 Build Scenarios Tab



You can define user profiles that specify which scripts the users playback to emulate real users and how many virtual users to emulate.

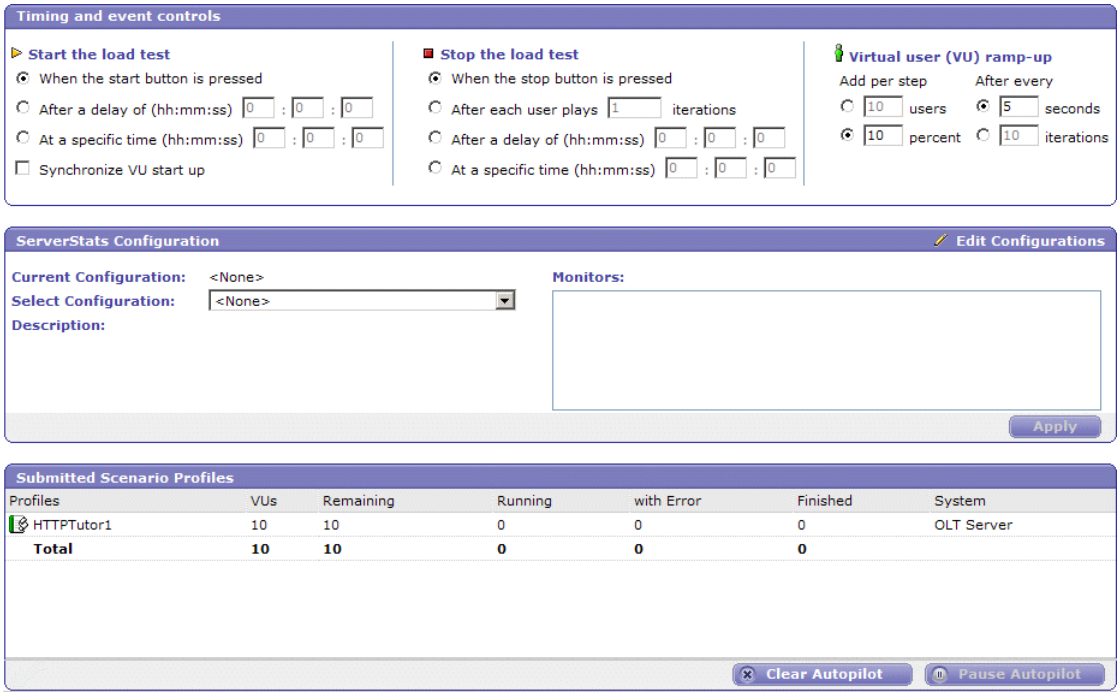
2.4.2 Set up Autopilot Tab

The Autopilot tab is where you specify the information needed to control how the scenario starts and runs. The Autopilot controls the starting and stopping of the scenario, the rate at which new virtual users are started, and shows the total number of virtual users and the number of running virtual users.

You specify the session, start and stop times, and the virtual user rampup specifications for the Submitted Scenario Profile. It also shows the list of virtual user profiles submitted in the Oracle Load Testing scenario.

The Autopilot tab is also where you select the ServerStats configuration to apply to the load test scenario. ServerStats configurations define the metrics or metric profiles to use to monitor server-side application, database, system, and Web server statistics during a load test scenario.

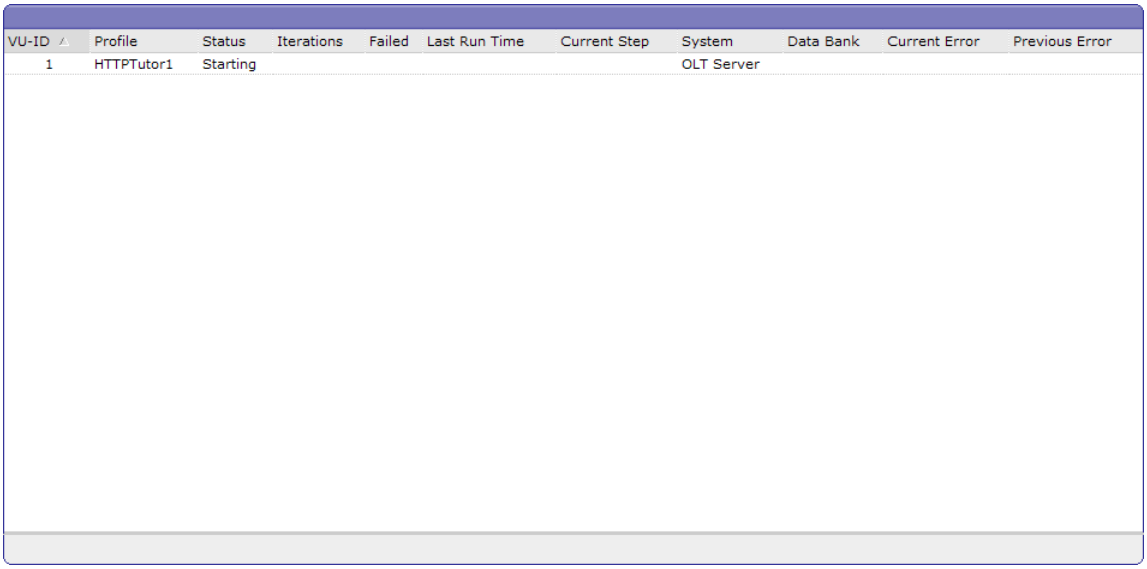
Figure 2–19 Autopilot Tab



2.4.3 Watch Virtual User Grid Tab

The Watch Virtual User Grid tab lists the currently running virtual users and the profile and playback details associated with each.

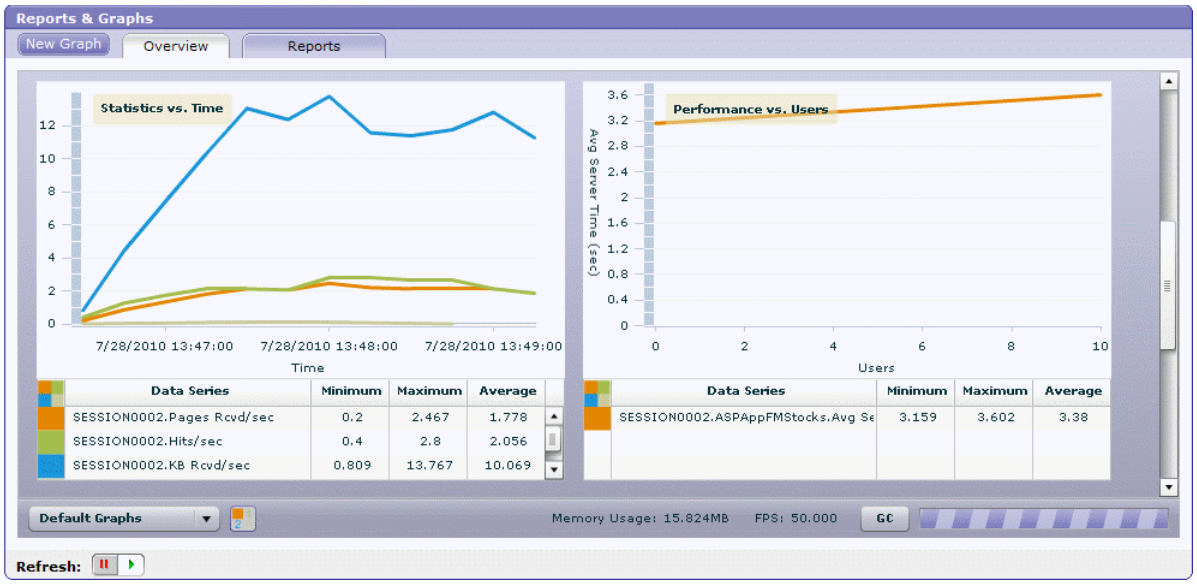
Figure 2–20 Watch Virtual User Grid Tab



2.4.4 View Run Graphs Tab

The View Run Graphs tab lets you define graphs and view the graphs at run-time.

Figure 2–21 View Run Graphs Tab

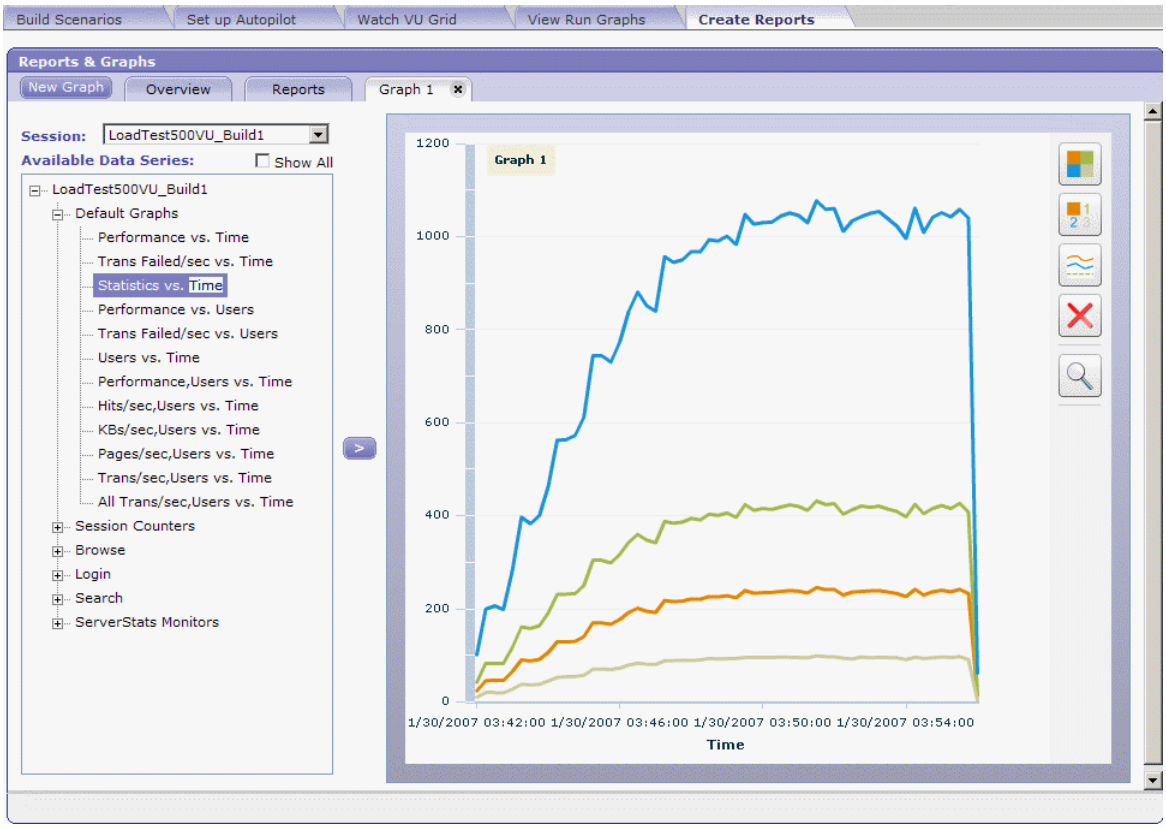


You can also view the Performance Statistics report from the View Run Graphs tab. The Performance Statistics window shows a summary of the performance data for the running virtual users.

2.4.5 Create Reports Tab

The Create Reports tab lets you create post session reports and graphs.

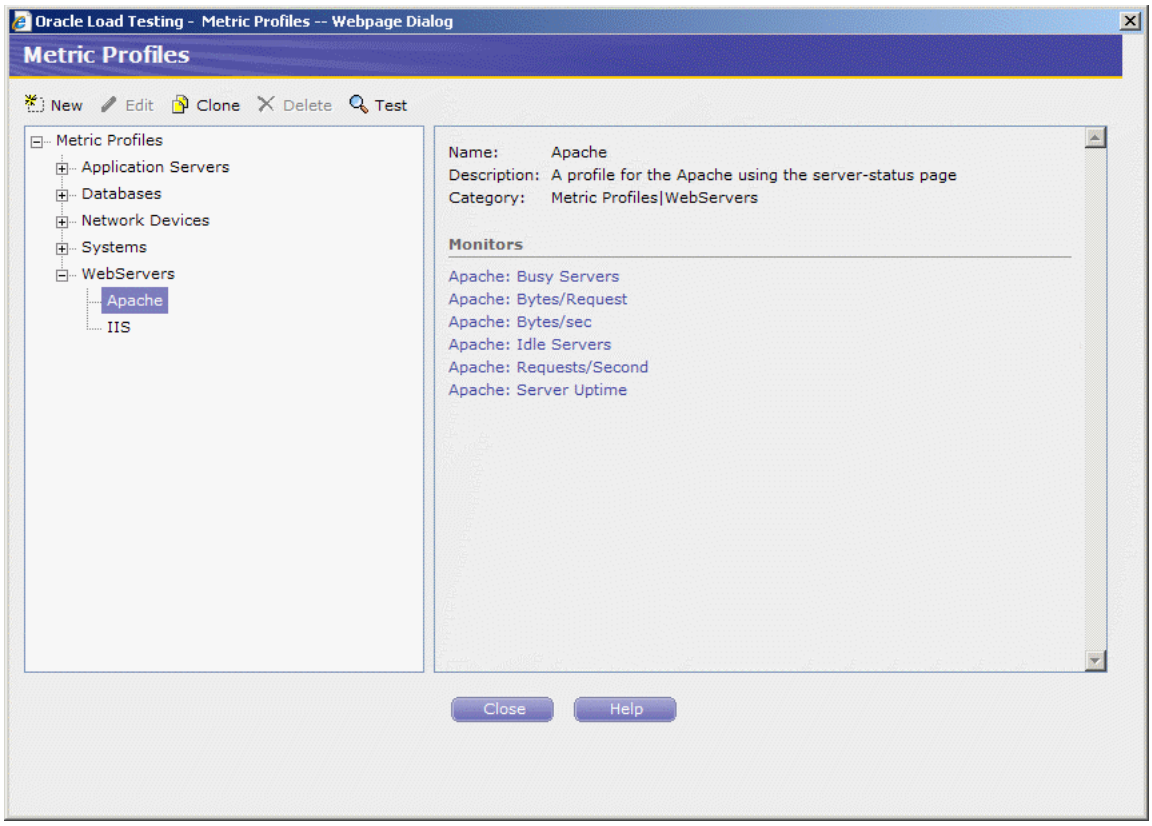
Figure 2–22 Create Reports Tab



2.4.6 ServerStats

The ServerStats component of Oracle Load Testing lets you monitor a variety of server-side application, database, system, and Web server statistics. You can configure ServerStats to display real-time performance statistics for the various hosts and services available from the server such as, percentage of CPU usage, memory usage, Web server statistics, etc.

Figure 2–23 ServerStats Metric Profiles Window



You can monitor specific counters in real time using the visual indicator gauges or using graphs. In addition to performance monitoring, ServerStats let you define scripts that can log warnings or alarms if a server's counter performance goes outside a defined range.

2.5 Oracle Test Manager Main Window Features

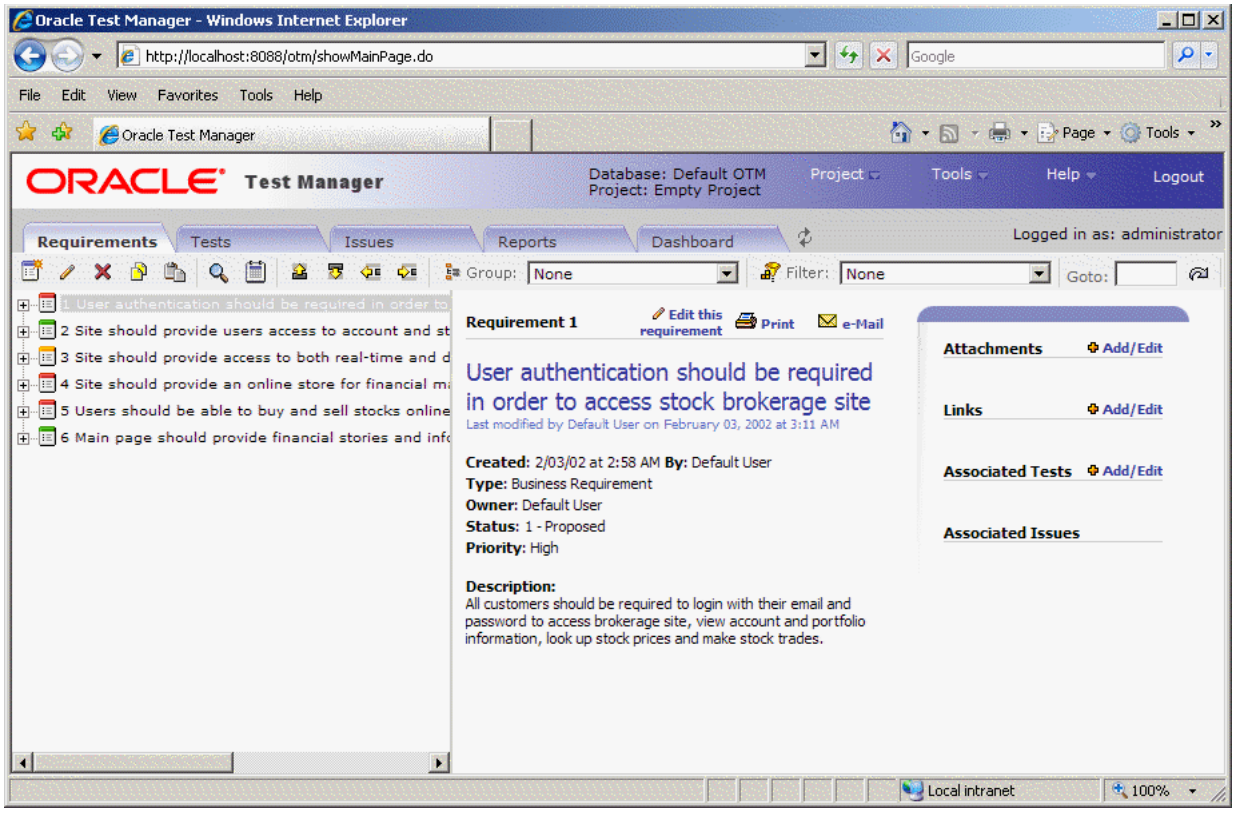
Oracle Test Manager lets you create projects that group together and organize test scripts, requirements that need to be tested, and issues resulting from the tests. Once created, you can indicate the relationships among these items, allowing you to quickly and easily find all information pertaining to a particular test script, requirement, or issue.

The main window consists of the menu bar, toolbar, and the tab views.

2.5.1 Requirements Tab

The requirements tab lets you work with requirements.

Figure 2–24 Requirements Tab



The number of requirements displayed is determined by the number you enter in the Maximum Tree Nodes field in options. You can:

- Expand and collapse the tree view by clicking on the plus and minus signs.
- View the next or previous group of requirements by clicking the **Next** or **Previous** button.
- Hold the mouse over a node to view a count of its child nodes.
- Move requirements by selecting a requirement and using the move buttons.
- Search requirements by clicking the **Find** button.
- Group requirements by clicking **Group**.
- Filter requirements by clicking **Filter**.
- Toggle between the tree view and grid view by clicking the **Tree View** and **Grid View** buttons.

The color of the icon in front of the requirement indicates its priority. The default colors are as follows and can be changed by changing the order of the Requirement Priorities in the Administrator.

- Red, high priority
- Yellow, medium priority
- Green, low priority

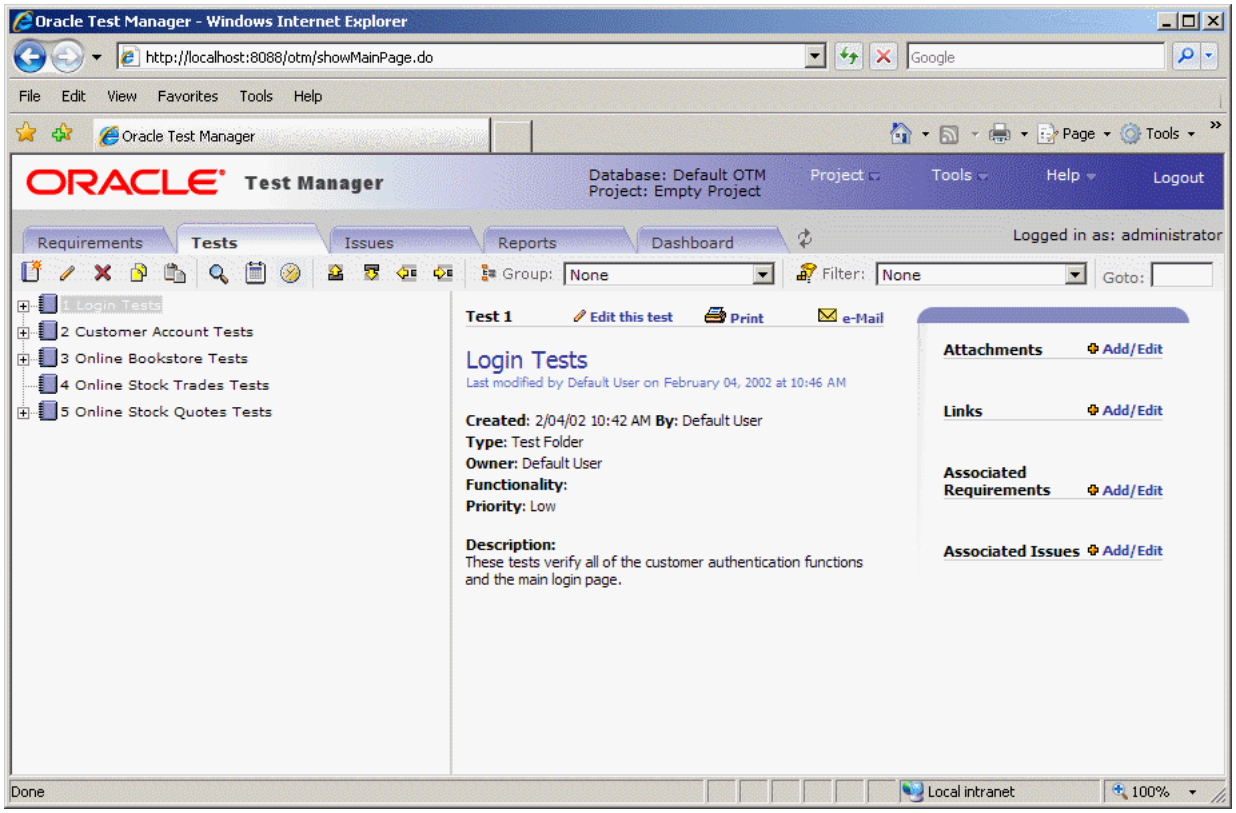
The right pane lists the selected requirement's details. In the upper right corner, associated tests and issues are listed as well as attachments and links. You can:

- Click **Edit this requirement** to open the Edit Requirement dialog box.
- Click **Print** to print the right pane.
- Click **e-Mail** to e-mail the requirement. The title and description are automatically copied to the e-mail.
- Click on an associated test or issue to view its details.
- Click on an attachment to open it in the appropriate application.
- Click on a link to view the URL in a separate browser window.
- Select **Add/Edit** to add or edit attachments, links, or associated items.

2.5.2 Tests Tab

The tests tab lets you work with tests.

Figure 2–25 Tests Tab



The number of tests displayed is determined by the Maximum Tree Nodes setting in options. You can:

- Expand and collapse the tree view using the plus and minus signs.
- View the next or previous group of tests by clicking the **Next** or **Previous** button.
- Hold the mouse over a node to view a count of its child nodes.
- Move tests by selecting the test and using the **Move** buttons.
- Search tests by clicking the **Find** button.
- Group tests by clicking the **Group** button.

- Filter tests by clicking the **Filter** button.
- Schedule when to run tests by clicking the **Schedule** button.
- Toggle between the tree view and grid view by clicking the **Tree View** and **Grid View** buttons.

The icon in front of the test indicates the type of test as follows:

- Manual test - blue with a pencil.
- Test folder - blue with a spiral.
- Oracle OpenScript - blue with a pencil.
- Test group - blue with a plus sign.
- 3rd Party test - blue with two stars

The color of the icon in front of the test indicates the last result from running the test, as follows:

- Green, passed
- Red, failed
- Yellow, warning
- Blue, not run
- Silver, currently running

The right pane lists the selected test's details. Test steps and run history are displayed. You can:

- Click **Edit this test** to open the Edit Test dialog box.
- Click **Print** to print the right pane.
- Click **e-Mail** to e-mail the test. The title and description are automatically copied to the e-mail.
- Click **Run this test** to start the Run Manual Test wizard or run an Oracle OpenScript test.
- Click **Delete Results** to display the Delete Results dialog box for deleting results from particular test runs.
- Click the date in the **Run History** section to display result details for a particular run.

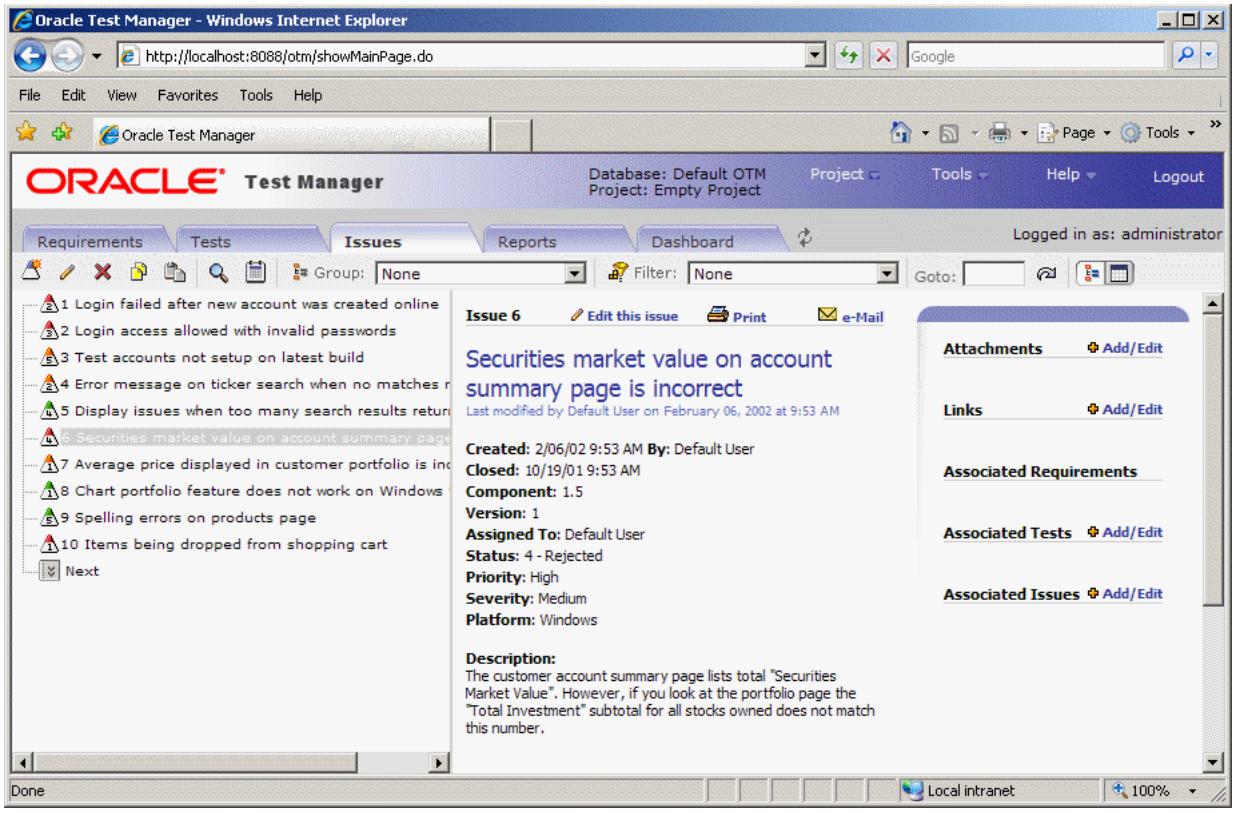
In the upper right corner or the right pane, associated requirements and issues are listed as well as attachments and links. You can:

- Click on an associated requirement or issue to view its details.
- Click on an attachment to open it in the appropriate application.
- Click on a link to view the URL in a separate browser window.
- Select **Add/Edit** to add or edit attachments, links, or associated items.

2.5.3 Issues Tab

The issues tab lets you work with issues.

Figure 2–26 Issues Tab



The number of issues displayed is determined by the Maximum Tree Nodes setting in options. You can:

- View the next or previous group of issues by clicking the **Next** or **Previous** button.
- Group issues by clicking the **Group** button.
- Filter issues by clicking the **Filter** button.
- Hold the mouse over a node to view a count of its child nodes.
- Search issues by clicking the **Find** button.
- Toggle between the tree view and grid view by clicking the **Tree View** and **Grid View** buttons.
- Display a particular issue by entering the issue number in the **Goto** field and clicking the **Goto** button.

The color of the icon in front of the issue indicates its priority. The default colors are as follows and can be changed by changing the order of the Issue Priorities in Oracle Test Manager Administrator:

- Red, high priority
- Yellow, medium priority
- Green, low priority

The number inside the icon corresponds to the status number.

The right pane lists information about the selected issue including the issue's details, solution, priority, and status. You can:

- Click **Edit this issue** to open the Edit Issue dialog box.
- Click **Print** to print the right pane.
- Click **e-Mail** to e-mail the issue. The title and description are automatically copied to the e-mail.

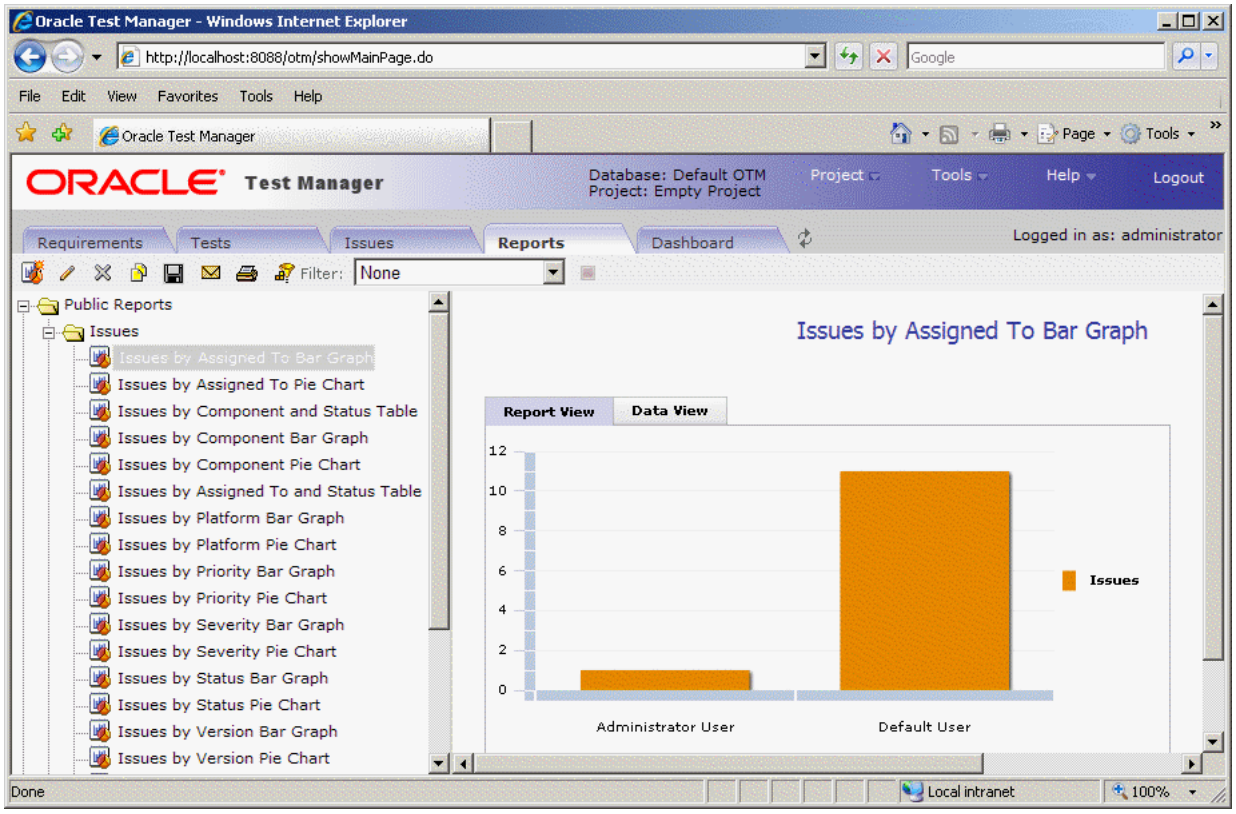
In the upper right corner of the right pane, associated requirements, tests, and issues are listed as well as attachments and links. You can:

- Click on an associated test, requirement, or issue to view its details.
- Click on an attachment to open it in the appropriate application.
- Click on a link to view the URL in a separate browser window.
- Select **Add/Edit** to add or edit attachments or links.

2.5.4 Reports Tab

The Reports tab lets you work with both standard and custom reports.

Figure 2–27 Reports Tab



Oracle Test Manager comes with a standard set of reports that can be viewed as either a graphic or as data. In addition, you can create custom reports to display only the data that you are interested in. You can:

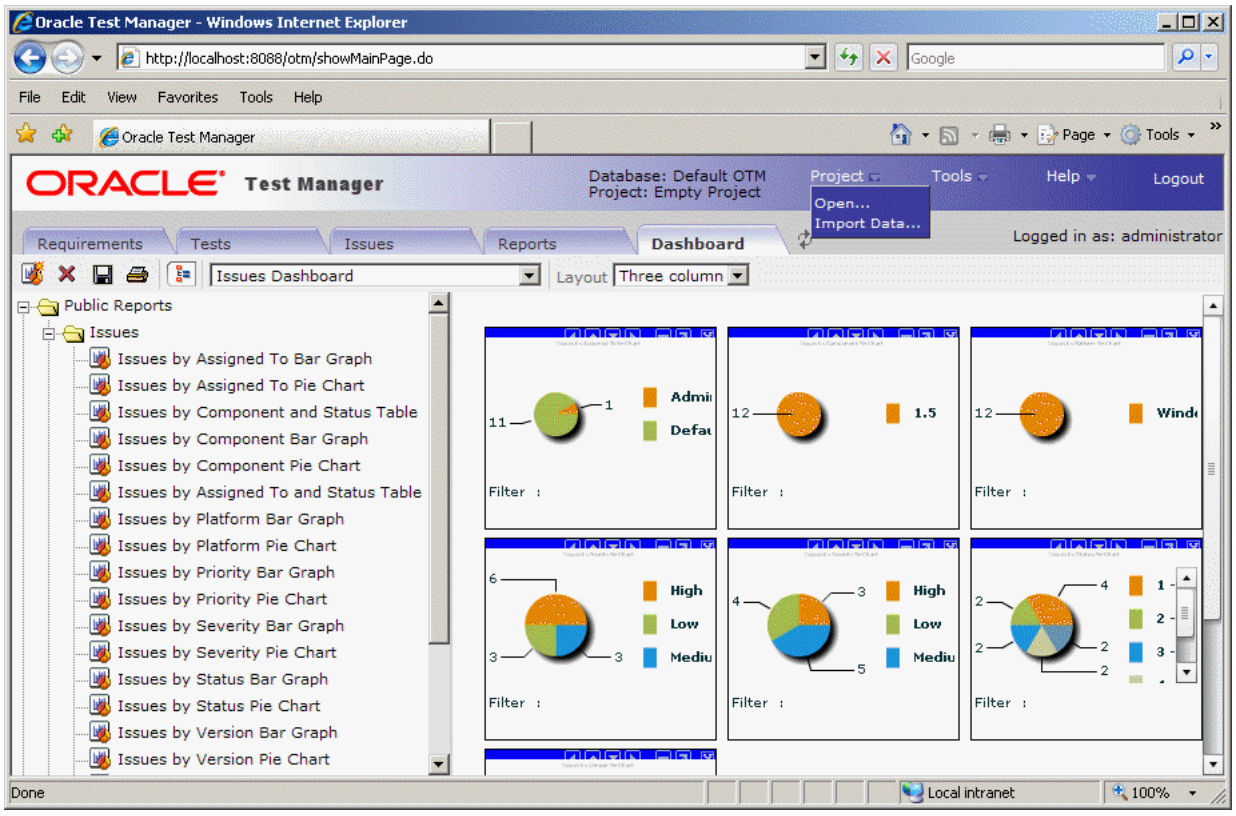
- View a standard or custom report by selecting it from the left tree.
- Add a custom report by clicking **Add**.
- Edit a custom report by clicking **Edit**.
- Delete a custom report by clicking **Delete**.

- Clone an existing report to create a copy of it that you can then edit by clicking **Clone**.
- Save custom reports by clicking **Save**.
- Email reports by clicking **Email**.
- Print reports by clicking **Print**.
- **Filter** the fields in the report to display only the data in which you are interested.
- Export reports to jpg, and xls formats.

2.5.5 Dashboard Tab

The Dashboard tab lets you view an overview of reports.

Figure 2–28 Dashboard Tab



One Dashboard report is available for requirements, tests, and issues. You can customize which reports are displayed for each and then save the view. In addition, you can select the number of columns to use for the display.

You can:

- Add reports to the view by clicking **Add**.
- Remove the selected dashboard report by clicking **Delete**.
- Save the customized view by clicking **Save**.
- Print dashboard reports by clicking **Print**.
- Toggle the display of the report tree by clicking **Toggle.**

Each report has the following toolbar with the following options, described from left to right.

Move Left - moves the report one space to the left.

Move Up - moves the report up one space.

Move Down - moves the report down one space.

Move Right - moves the report one space to the right.

Minimize - minimizes the report.

Maximize - displays the report in a separate window. From there you can toggle between report and data views, and export the report.

Delete - removes the report from the display.

3

Oracle OpenScript Tutorial

This tutorial walks you through the main features of Oracle OpenScript. The tutorial consists of the following examples:

- **Starting the Avitek Medical Records Sample Application** - explains how to start the sample Avitek Medical Records Server for use with this tutorial.
- **Starting Oracle OpenScript** - describes how to start OpenScript.
- **Creating a Web Functional Test Script** - describes how to create a Web Functional test script project and record a script.
- **Working with Scripts** - explains the features of the OpenScript script tree and how to examine the structure and content of a recorded script.
- **Playing Back a Web Functional Test Script** - explains the procedure for playing back Web Functional Test scripts that you have recorded and viewing the results.
- **Adding Tests to the Script** - explains how to add tests to your OpenScript scripts.
- **Creating an HTTP Test Script** - describe how to create an HTTP protocol load test script project and record a load test script.
- **Creating an HTTP Test Script with Databanks** - describe how to create an HTTP protocol load test script project and explains how to configure and use Databanks to run iterative tests on a login form using data from an external file.
- **Stopping the Avitek Medical Records Server** - explains how to stop the sample Avitek Medical Records Server used with this tutorial.

The tutorial is designed to be followed sequentially from beginning to end. Many of the examples are interrelated and build upon the steps in previous examples.

3.1 Starting the Avitek Medical Records Sample Application

The tutorial uses the Avitek Medical Records Sample Application to record and playback OpenScript scripts. The WebLogic server and Avitek Medical Records Sample Application need to be started on the system before creating OpenScript script projects and recording scripts.

To start the Weblogic Server and Avitek Medical Records Sample Application:

1. Select **Programs** from the **Start** menu and then select **Sample** from the **Oracle Application Testing Suite** menu.
2. Select **Start MedRec Application**.
3. Wait until the Avitek Medical Records Sample Application starts in the Weblogic server.

4. Close the browser window after the server and sample application are started. When recording OpenScript scripts, the browser window will automatically open again to specify the Web address to record.

3.2 Starting Oracle OpenScript

To start OpenScript:

- Select **Programs** from the **Start** menu, then select **OpenScript** from the **Oracle Application Testing Suite** Start menu.

3.3 Example 1: Creating a Web Functional Test Script

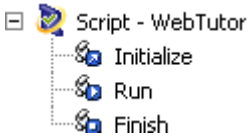
This example illustrates the creation of an OpenScript Web Functional test script project and recording a script. A script project creates the basic structure of an OpenScript script. Initially, the script project includes only the Initialize, Run, and Finish script nodes with the underlying Java code. You use the recording capabilities of OpenScript to record page navigations and generate the Java code for the script.

3.3.1 Creating a Web Functional Test Script Project

To create a Web Functional Test Script Project:

1. Select **New** from the **File** menu.
2. Select **Web** under the **Functional Testing (Browser/GUI Automation)** folder.
3. Click **Next**.
4. Make sure the Repository is set to Default.
5. Enter WebTutor as the script name.
6. Click **Finish**. OpenScript creates the script project and shows the Initialize, Run, and Finish nodes in the Script View as follows:

Figure 3–1 OpenScript Script Project Tree



3.3.2 Recording a Web Functional Test Script

Recording scripts captures the page navigations and generates the Java code that will be used to drive script playback for testing purposes. Web Functional test scripts record and playback actions performed on browser objects.

In this example, we will open the Avitek Medical Records sample application and log in as a patient. During recording, we will add a Text Matching test to verify that the log in was successful.

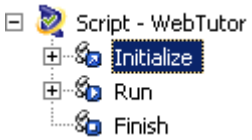
To record a Web Functional Test Script:

1. Select **Record** from the **Script** menu or click the Record toolbar button. OpenScript opens a browser window and the OpenScript toolbar window.
2. Enter `http://<machineName>:7011/medrec/` into the browser Address line and press **Enter**.

3. When the application loads, click **Start using MedRec!**. A second browser window opens with Avitek Medical Records Sample Application.
4. Click Login under the Patient section.
5. Enter `fred@golf.com` as the Email address.
6. Enter `weblogic` as the password.
7. Click **Submit**.
8. Click the Text Matching Test button on the OpenScript floating toolbar.
9. Enter `WebText1` as the Test name.
10. Enter `Successfully logged in! Click here to continue` as the Text to Match.
11. Make sure **Source** is set to **HTML Display Contents**.
12. Make sure **Pass when** is set to **Pass if present**.
13. Make sure **Match** is set to **Exact**.
14. Make sure V**erify only, never fail** is selected.
15. Click **OK**.
16. Click Successfully logged in! Click here to continue.
17. Click Logout in the Avitek Medical Records application.
18. Close the Avitek Medical Records application browser window.
19. Close the Weblogic Server Avitek Medical Records Sample Application browser window.

    Recording automatically stops when you close the second browser window. The script tree view includes plus icons to indicate additional nodes have been added to the script as follows:

Figure 3–2 OpenScript Script Tree After Recording



20. Save the script.

3.4 Example 2: Working with Scripts

This example explains the features of the OpenScript Script tree and how to examine the structure and content of a recorded script.

After recording a script, you can expand the Tree View to view the page navigations, actions, and parameters recorded to the script. Before starting this example, make sure the script that you recorded in Example 1 is still displayed.

3.4.1 Viewing Information About Script Items

To view information about script items:

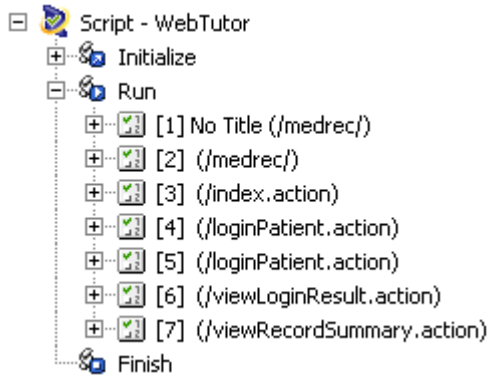
1. Expand the Initialize section to expand the tree as follows:

Figure 3–3 Script Tree with Expanded Initialize Section

The Initialize section executes commands that will be run only once at the beginning of script playback. In our recorded script, Launch Browser is the only action in the Initialize section.

2. Expand the Run section to expand the tree as follows:

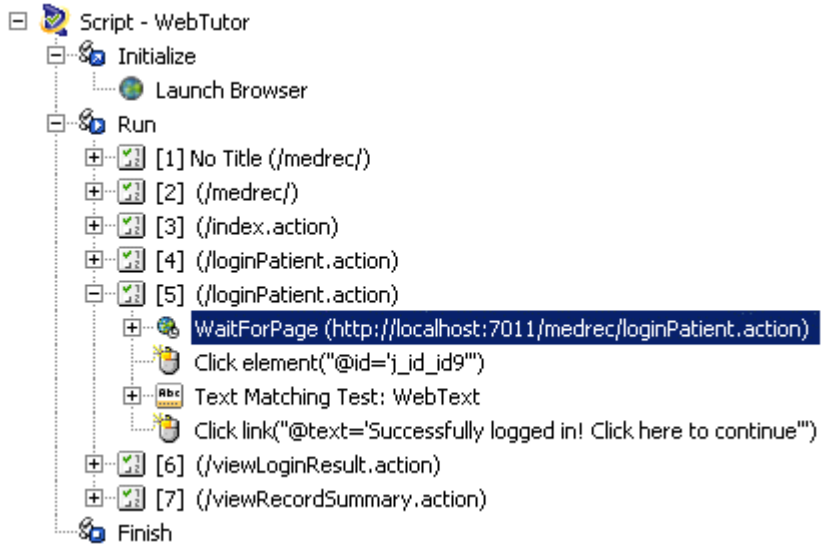
Figure 3–4 Script Tree with Expanded Run Section



The Run node contains the navigations recorded to the script as Step Groups. By default, OpenScript Web Functional test scripts create Step Group nodes based on web page navigations. Step Group preferences can be changed in the OpenScript Preferences.

3. Expand the [5] (/loginPatient.action) Step Group and select the WaitForPage (http://*systemName*:7011/medrec/loginPatient.action) node as follows:

Figure 3–5 Script Tree with Expanded Step Group Node



The Step Group node shows the actions and parameters recorded for specific navigations and actions on a Web page.

The Details View tabs show the details for a specific navigation including a screen shot of the recorded page, the HTML source, and the browser rendering of the page.

4. Click the Screenshot, HTML, and Browser tabs to view the script details.
5. Expand the WaitForPage (http://*systemName*:7011/medrec/loginPatient.action) node. The script records Think time nodes to include the amount of user delay that occurred during recording as follows:

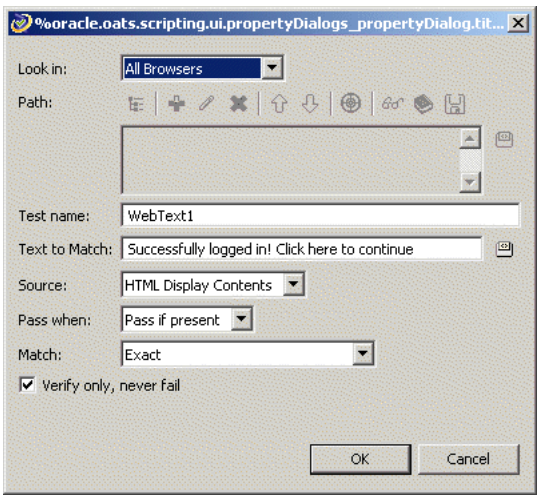
Figure 3–6 Script Think Node

6. Select the Think node, click the right mouse button and select **Properties**. The Think time properties lets you edit the delay recorded to a script as follows:

Figure 3–7 Script Think Node Properties

7. Change the Think time value and click **OK**. This is useful if you want to reduce long a delay time that may have occurred during recording of script.
8. Right-click the Text Matching Test: Web Text node and select **Properties**. You can view or change the Text Matching test properties as follows:

Figure 3–8 Text Matching Test Properties



9. Click **Cancel** to leave the Text Matching test properties unchanged.
10. Open and view the properties of other script nodes.

3.4.2 Using the Java Code View

When you create a script project and record a script, the Java code is automatically generated. The Java Code view shows the script navigations and data as Java programming code. The Java Code view corresponds to the Tree View. Any changes in the Code View will be automatically updated in the Tree View. The Java Code view has the following standard procedures:

- `initialize()` - corresponds to the Initialize node of the Tree View and executes any custom code added once at the beginning of script playback.
- `run()` - corresponds to the Run node of the Tree View and executes recorded and custom code one or more times during script playback depending upon databanks or other custom programming.
- `finish()` - corresponds to the Finish node of the Tree View and executes any custom code added once at the end of script playback.

Use Ctrl-space to open an Intellisense window listing available procedures. See the API Reference in the OpenScript Platform Reference help for additional programming information.

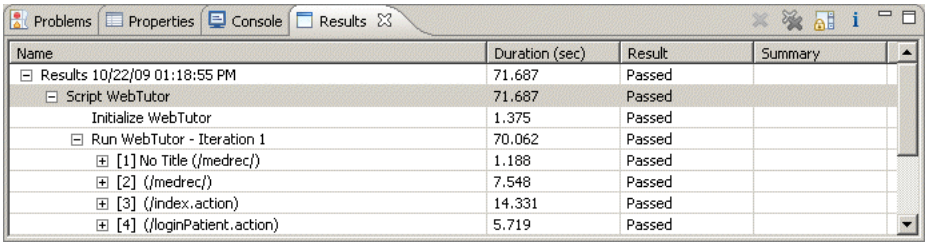
3.5 Example 3: Playing Back a Web Functional Test Script

This example explains the procedure for playing back Web Functional Test scripts that you have recorded. It also shows the results log. Web Functional Test scripts perform actions on browser objects. During playback, the browser will open and you will be able to watch as script actions occur.

To play back an OpenScript script:

1. Select **Playback** from the **Script** menu or click the Playback toolbar button to play back the recorded script. The navigations and actions in the script Step Groups will be played back in the order recorded. The Browser navigates to each page, executes the any tests for each page, and shows the results in the Result view.
2. When playback finishes, expand the Results tree to view the results for each Step Group as follows:

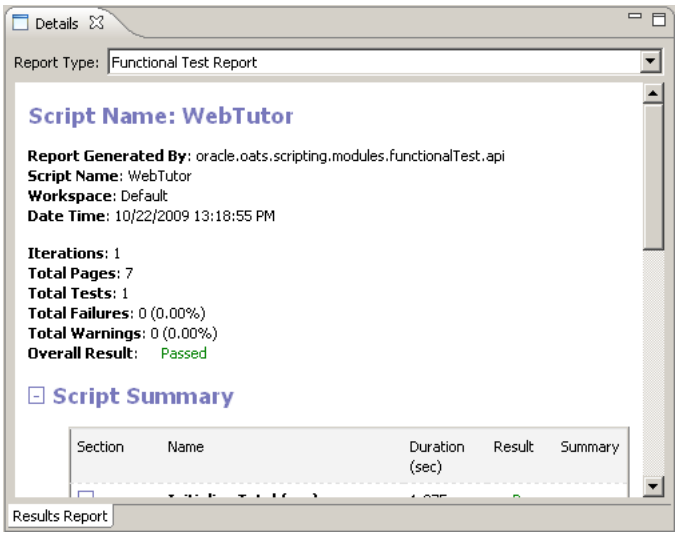
Figure 3–9 Script Playback Results in the Results View



   The "passed" results indicate that all referenced resources are available. Oracle OpenScript automatically generates a results report and opens the report in the Details view.

3. Select the top-level Result node or the script name in the Results view to view the results report in the Details view as follows:

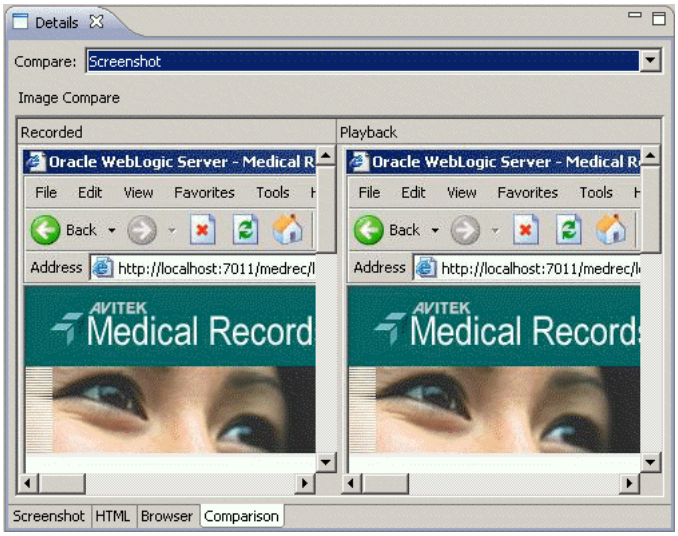
Figure 3–10 Results Report in the Details View



Notice that all tests passed. This is because you played back the script using the same version of the Web pages that was used to record the script. This establishes a baseline of tests for the Web application or Web site's content and structure.

4. Expand the Run node in the Results view then expand the results for Step Group [5] (/loginPatient.action).
5. Select the WaitForPage node under [5] (/loginPatient.action). The Details view shows the results for the specific Step Group.
6. Click the Comparison tab in the Details view. The Comparison tab lets you compare the recorded HTML content, screenshots, and browser renderings of page navigations to the play back values as follows:

Figure 3–11 Details View Showing the Web Functional Test Comparison Tab



7. Select the Content and Browser options in the **Compare** list to compare the recorded and playback HTML source and browser renderings of the page.

3.6 Example 4: Adding Tests to the Script

This example explains how to add tests to your OpenScript scripts. Make sure the script you recorded in the previous example is open in OpenScript.

OpenScript provides the ability to add the following test types to the pages in your Web Functional test script:

- Text Matching Test
- Server Response Test
- Object Test
- Table Test

Other OpenScript test modules provide test types specific to the type of module.

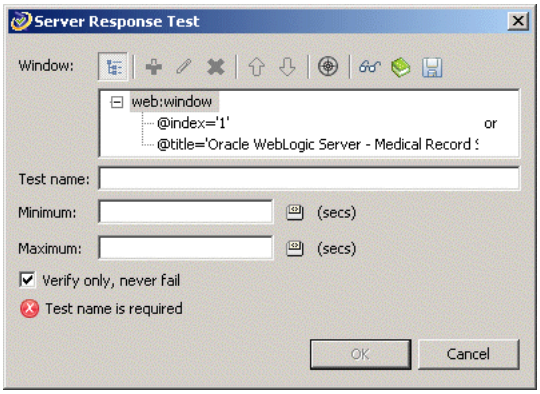
3.6.1 Inserting a Server Response Test Case

Server Response test cases measure the response time of a server access for a page in the script.

To add a Server Response test:

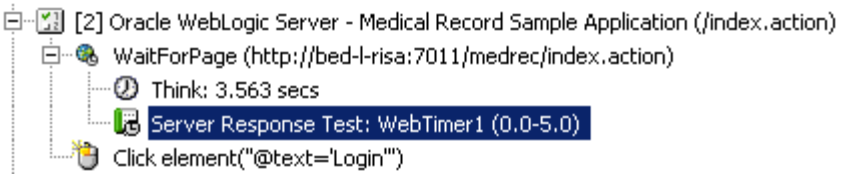
1. Expand the [2] (/medrec/) Step Group in the script tree.
2. Select the WaitForPage (http://*systemName*:7011/medrec/index.action) node.
3. Select **Add** from the **Script** menu, then select **Other**.
4. Expand the Web Tests folder, select **Server Response Test**, and click **OK**. OpenScript opens the Server Response Test properties dialog box as follows:

Figure 3–12 Server Response Test Properties Dialog Box



5. Type WebTimer1 as the test case name.
6. Set the **Minimum** time to 0 seconds.
7. Set the **Maximum** time to 5 seconds.
8. Click **OK** and view the test in the script. OpenScript adds the test to the Step Group as follows:

Figure 3–13 Server Response Test Added to the Script Tree



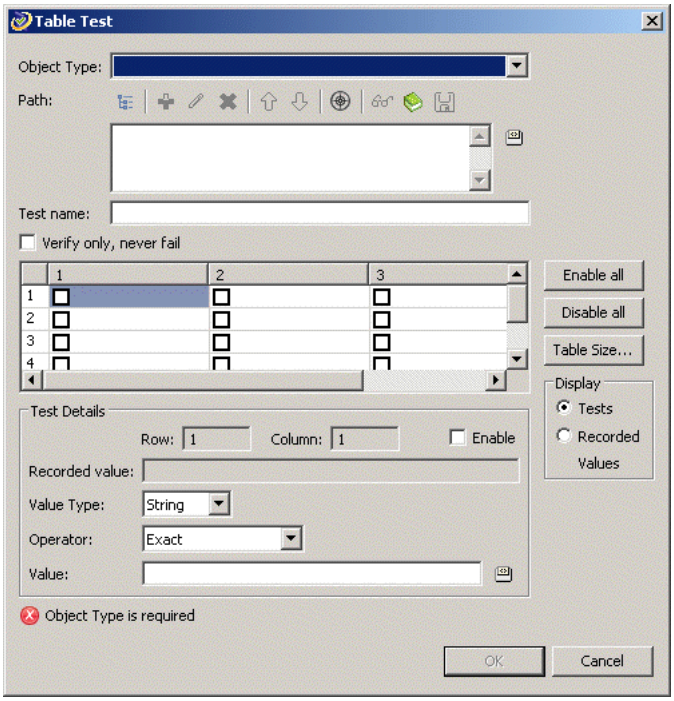
3.6.2 Inserting a Table Test

Table test cases let you define a custom test on a Web page table object. The Table Test properties lets you select the table object directly from the Web page by highlighting it with the mouse. The properties also let you specify the table object property to test and the type of test to perform.

To add a Table test:

1. Expand the [5] (/viewLoginResult.action) Step Group in the script tree.
2. Select the WaitForPage (http://*systemName*:7011/medrec/patient/viewLoginResult.action) node.
3. Select **Add** from the **Script** menu, then select **Other**.
4. Expand the Web Tests folder, select **Table Test**, and click **OK**. OpenScript opens the Table Test properties dialog box as follows:

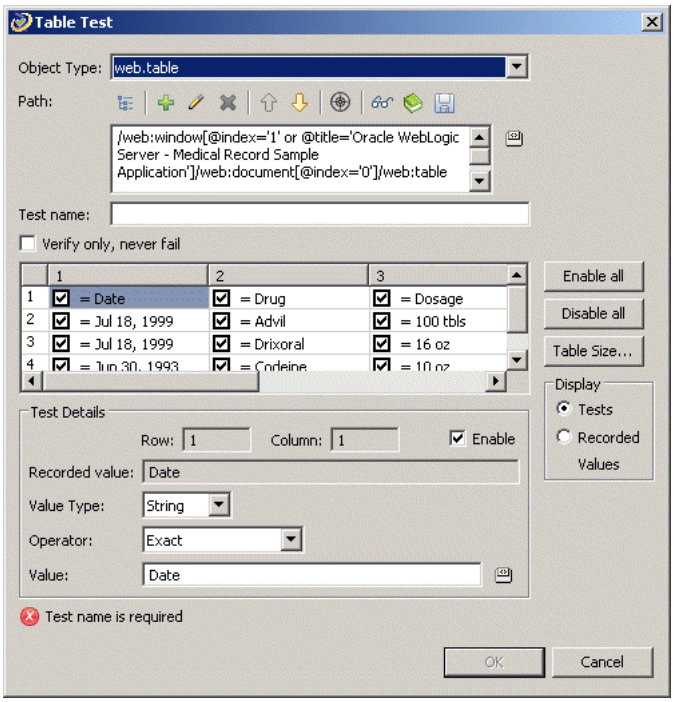
Figure 3–14 Table Test Properties Dialog Box



5. Click the select **Capture object in browser** button next to the **Path** field and select **Capture Object** from the menu. The capture mode starts.
6. Open the Medical Records Sample Application (http://*systemName*:7011/medrec/) in the browser window (which should still be open from the previous example).

7. Click **Start using MedRec!** to navigate to the page with the table data.
8. Click Login under the Patient section.
9. Enter `fred@golf.com` as the Email address.
10. Enter `weblogic` as the password.
11. Click **Submit**.
12. Click Successfully logged in! Click here to continue.
13. Move the mouse over the Prescriptions table so that the table is highlighted.
14. Press F10 to capture the object path in the Select Object dialog box.
15. Click **OK**. The object path for the Medical Records Sample Application web page table is added to the Table Test properties as follows:

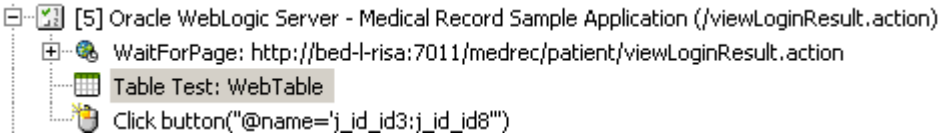
Figure 3–15 Table Test Properties Dialog Box with Captured Data



The Table Test properties lets you enable or disable testing for each table cell individually. The Test Details section shows the details for the currently selected table cell.

16. Clear the check boxes for all of the data items in rows 2, 3, and 4 (scroll the table as necessary). This will perform the testing on heading rows but not the data rows.
17. Enter WebTable as the Test name.
18. Click **OK** and view the test in the script. OpenScript adds the test to the script [6] (/viewLoginResult.action) Step Group as follows:

Figure 3–16 Table Test Added to the Script Tree



19. Save the script.
20. Close the Medical Records Sample Application browser windows.
21. Playback the script and verify that all tests pass.
22. Close the WebTutor script when finished.

3.7 Example 5: Creating an HTTP Test Script

This example explains how to create an HTTP test script project to use for load testing an application. HTTP test scripts record and playback navigations performed using the HTTP protocol. HTTP script are typically used for performing load tests on an application using the Oracle Load Testing application.

3.7.1 Creating an HTTP Script Project

To create an HTTP script project:

1. Select **New** from the **File** menu.
2. Select **Web/HTTP** under the **Load Testing (Protocol Automation)** folder.
3. Click **Next**.
4. Make sure the Repository is set to Default.
5. Enter HTTPTutor1 as the script name.
6. Click **Finish**. OpenScript creates the script project and shows the Initialize, Run, and Finish nodes in the Script view.

3.7.2 Recording an HTTP Script

To record an HTTP script:

1. Select **Record** from the **Script** menu or click the Record toolbar button. OpenScript opens a browser window and the OpenScript toolbar window.
2. If asked if you want to clear the cache, click **Yes**.
3. Enter `http://<machineName>:7011/medrec/` into the browser Address line and press **Enter**.
4. When the application loads, click **Start using MedRec!**. A second browser window opens with Avitek Medical Records Sample Application.
5. Click Login under the Administrator section.
6. Enter `admin@avitek.com` as the Email address.
7. Enter `weblogic` as the password.
8. Click **Submit**.
9. Click View Pending Requests.
10. Click Logout in the Avitek Medical Records application.

11. Close the Avitek Medical Records application browser window.
12. Close the Weblogic Server Avitek Medical Records Sample Application browser window. Recording automatically stops.
13. Playback the script and verify there are no errors.

3.8 Example 6: Creating an HTTP Test Script with Databanks

This example explains how to create an HTTP test script and use a databank to drive testing. This example also explains one way to use the Databanks with the Text Matching test case to verify Login results pages. The Databanks provides the capability to run iterative tests using data from a Databank file.

3.8.1 Creating an HTTP Script Project

To create an HTTP script project:

1. Select **New** from the **File** menu.
2. Select **Web/HTTP** under the **Load Testing (Protocol Automation)** folder.
3. Click **Next**.
4. Make sure the Repository is set to Default.
5. Enter HTTPTutor2 as the script name.
6. Click **Finish**. OpenScript creates the script project and shows the Initialize, Run, and Finish nodes in the Script view.

3.8.2 Recording an HTTP Script

To record an HTTP script:

1. Select **Record** from the **Script** menu or click the Record toolbar button. OpenScript opens a browser window and the OpenScript toolbar window.
2. Reload the Medical Records Sample Application (http://*systemName*:7011/medrec/) in the browser window.
3. When the application loads, click **Start using MedRec!**. A second browser window opens with Avitek Medical Records Sample Application.
4. Click Login under the Patient section.
5. Enter `fred@golf.com` as the Email address.
6. Enter `weblogic` as the password.
7. Click **Submit**.
8. Click Successfully logged in! Click here to continue.
9. Click Logout in the Avitek Medical Records application.
10. Close the Avitek Medical Records application browser window.
11. Close the Weblogic Server Avitek Medical Records Sample Application browser window. Recording automatically stops.

3.8.3 Viewing the Parameters in the Script

To view the script parameters:

1. Expand the [4] Oracle WebLogic Server - Medical Record Sample Application Step Group in the script tree. Notice the parameters under the Post Data node of the tree as follows:

Figure 3–17 Script Parameter Values for HTTP Post Data

The `usernameInput` and `passwordInput` parameters can be mapped to a databank file to pass data from an external file.

3.8.4 Configuring Databanks with OpenScript Scripts

Before you can map parameters to databank, you must configure the script with the databank file.

To configure a databank in OpenScript scripts:

1. Open or create a Script project.
2. Select **Script Properties** from the **Script** menu.
3. Select the **Script Assets** type.
4. Select **Databanks**.
5. Click **Add**.
6. Select **Databank** the select **CSV file**.
7. Select the Default Repository from the **My Repositories** tree.
8. Select the `avitek.csv` databank file from the DataBank repository folder.
9. Enter an alias name to use for the Databank or leave the default alias name. The default alias name is the name of the .CSV Databank file.
10. Click **OK**.
11. Click **OK** to add the Databank file.

Databank files are comma-separated value files. The field names are on the first line of the file separated by commas (no spaces). The field data is on subsequent lines separated by commas (different line for each record, no spaces around commas). The following shows an example:

```
FirstName,LastName,Mail,Phone
John,Smith,JohnS@company.com,x993
Mary,Ellen,MaryE@company.com,x742
```

If a data value contains a comma, place quotation marks around the value, as follows:

```
John,Smith,"Anytown, MA","(603) 993-0000"
```

3.8.5 Getting Databank Records

To get Databank records to use with a script:

1. Open or create a script project.
2. Configure the Databank to use with a script in the Assets Script Properties.
3. Select the script node where you want to use the Databank record.
4. Select the **Script** menu and then select **Other** from the **Add** sub menu.
5. Expand the General node and select Get Next Databank Record.
6. Click **OK**.
7. Select the Databank `avitek` alias to specify the Databank file to get the record from.
8. Click **OK**. A GetNextDatabankRecord: *databank alias* node will be added to the script.

   In the Java Code view, the `getDatabank("`*`databank alias`*`").getNextDataBankRecord()` method will be added to the script code:

   ```
   getDatabank("avitek").getNextDatabankRecord();
   ```

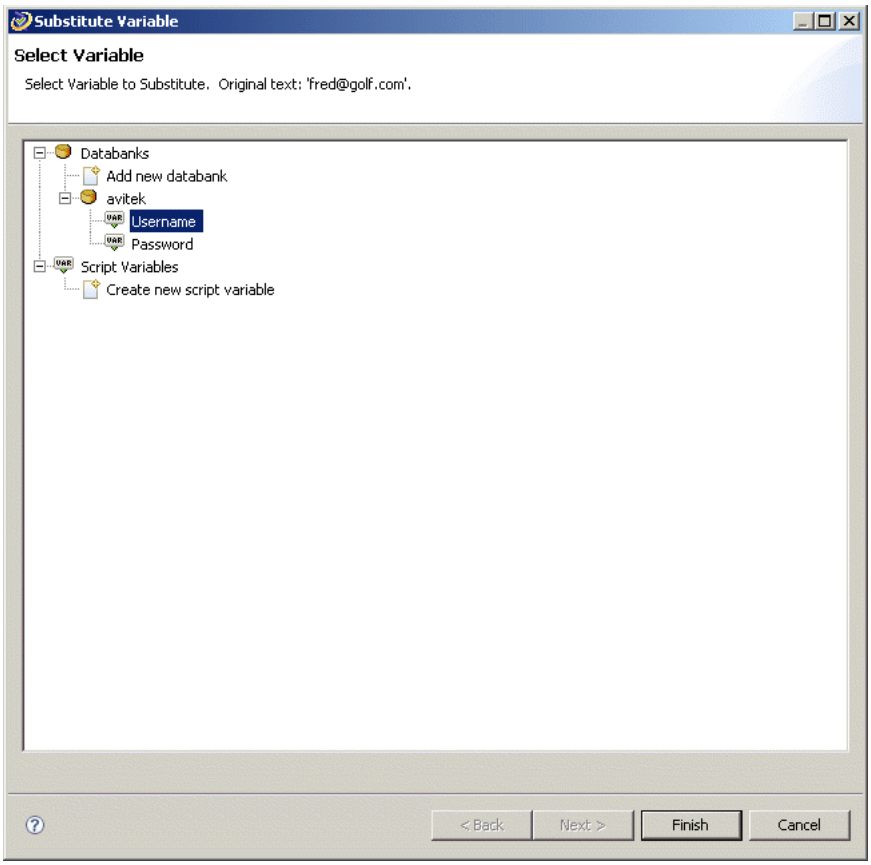
3.8.6 Mapping Script Parameters to Databank Fields

After configuring a databank in a script, you can map the databank fields to specific script parameters.

To map databank fields to script parameters:

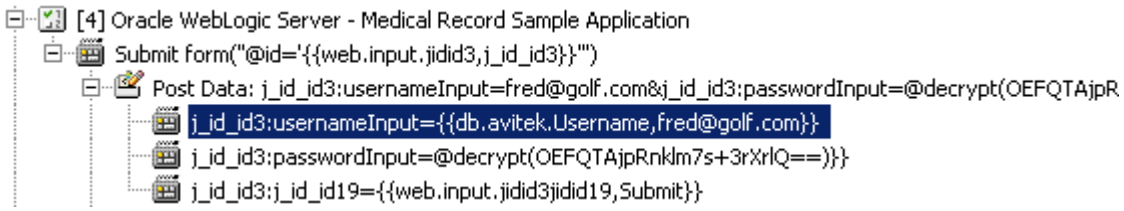
1. Expand the [4] Oracle WebLogic Server - Medical Record Sample Application script tree node.
2. Right-click the `usernameInput` parameter and select **Substitute Variable**. The Substitute Variable window opens with the databank field names listed as follows:

Figure 3–18 Substitute Variable Window



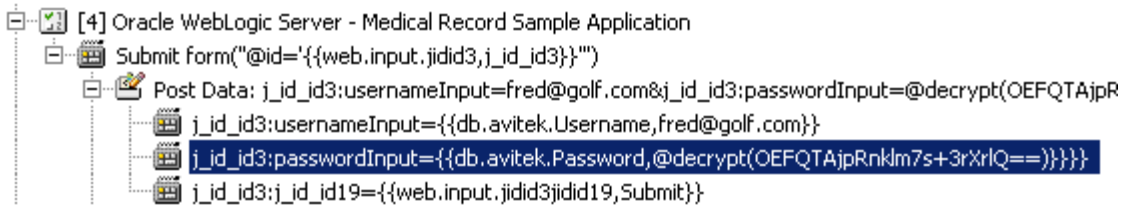
3. Select the Username field and click **Finish**. The parameter value changes to a databank variable in double braces{{db.avitek.Username,fred#@golf.com}} as follows:

Figure 3–19 Script Parameter with a Mapped Databank Variable



4. Right-click the `passwordInput` parameter and select **Substitute Variable**. The Substitute Variable window opens with the databank field names listed.
5. Select the Password field and click **Finish**. The parameter value changes to a databank variable in double braces{{db.avitek.Password,weblogic}} as follows:

Figure 3–20 Script Parameters with Multiple Mapped Databank Variables



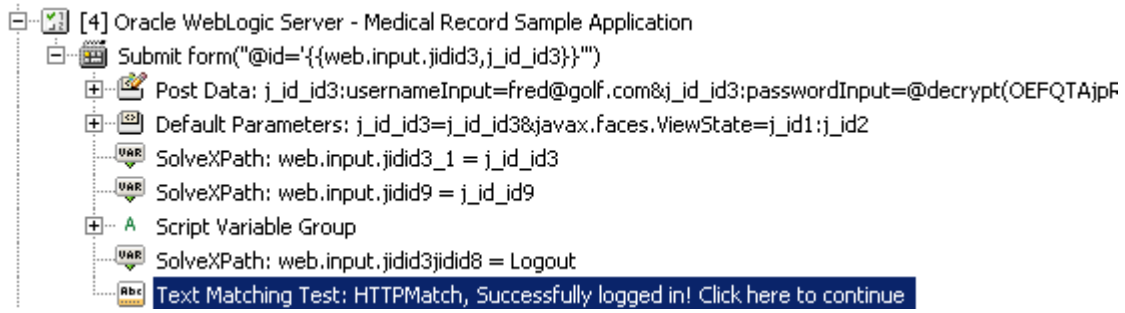
6. Save the script.

3.8.7 Inserting a Text Matching Test

Now we want to insert a Text Matching text case that verifies that the login results were successful.

1. Select the http://localhost:7011/medrec/loginPatient.action node under the [4] Oracle WebLogic Server - Medical Record Sample Application script tree node. The page appears in the Details view.
2. Select **Add** from the **Script** menu then select **Other**.
3. Select **Text Matching Test** in the HTTP Tests folder and click **OK**.
4. Enter HTTPMatch as the test name.
5. Enter `Successfully logged in! Click here to continue` as the Text to Match.
6. Make sure the **Source** options is **HTML Display Contents**.
7. Make sure the **Pass When** option is **Pass if Present**.
8. Make sure the **Match** option is **Exact**.
9. Click **OK**. The Text Matching test node appears in the script tree at the end of the Step Group as follows:

Figure 3–21 Script Tree with a Text Matching Test Node



10. Save the script.

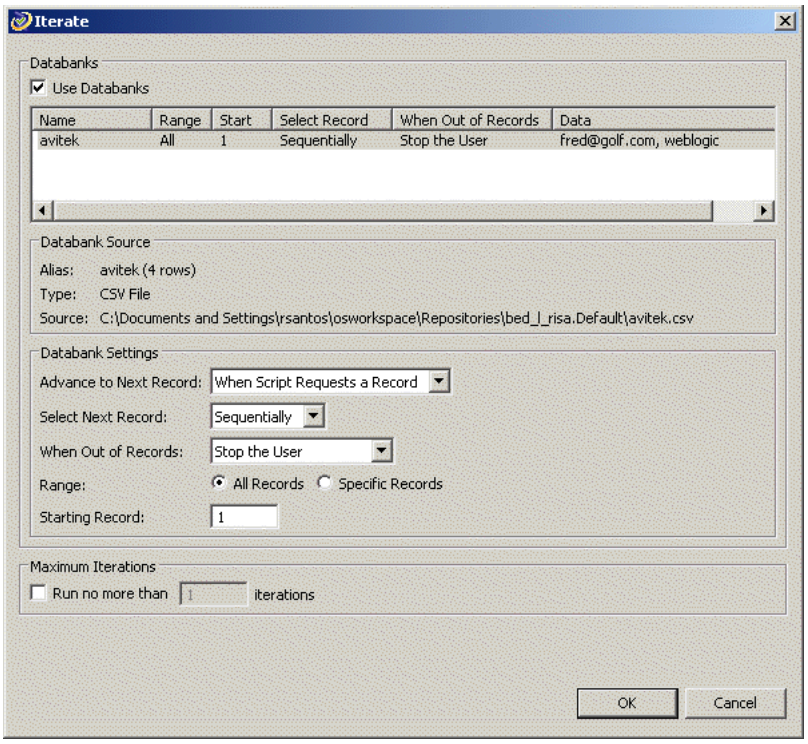
3.8.8 Playing Back the Script with Iterations

Now that the script has a databank mapped to the Post Data parameters, we can play back the script with multiple iterations to use all of the values in the databank file.

To play back the script with iterations:

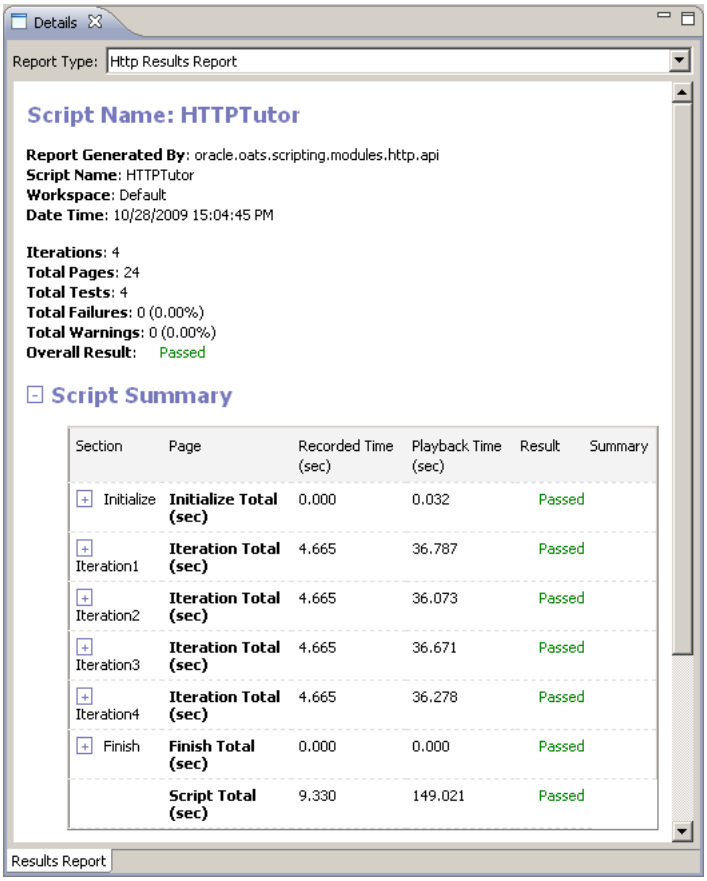
1. Select **Iterate** from the **Script** menu. The Iterations dialog box opens as follows:

Figure 3–22 Iterations Dialog Box



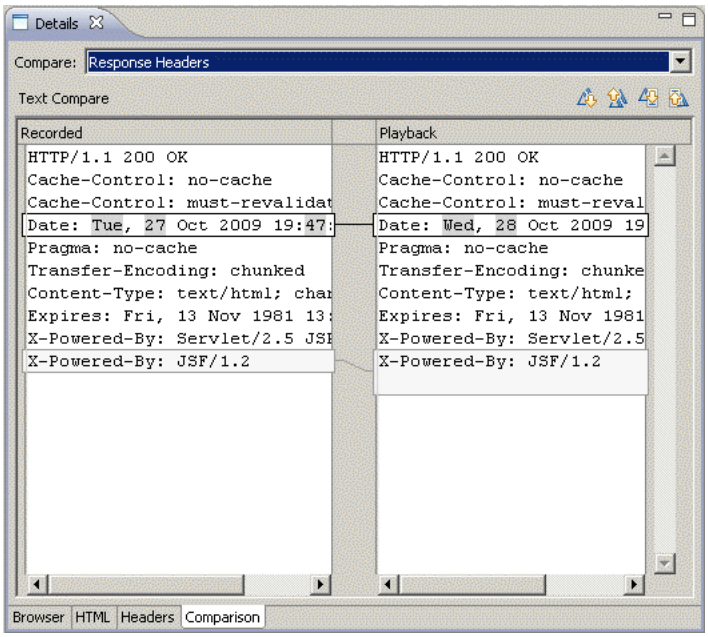
2. Make sure **Use Databanks** is selected.
3. Make sure **Range** is set to **All Records**.
4. Make sure the **Starting Record** is set to 1.
5. Make sure the **Run no more than [] iterations** is selected and set to 1.
6. Click the **OK** button to playback the script with multiple iterations.
7. Watch the Details view as the script plays back the script several times using a different data value for the login each time.
8. View the playback results in the Results view. At the end of playback the Details view shows the results report similar to the following:

Figure 3–23 Results Report for Multiple Iterations



9. Scroll the Results view and select one of the HTTP navigation nodes such as Get window[@index='0'] http://*systemName*:7011/medrec/index.action.
10. Click the Comparison tab in the Details view. The Comparison tab for HTTP scripts lets you compare the recorded HTML content, Request Headers, Response Headers, and Cookies to the playback values as follows:

Figure 3–24 Details View Showing the HTTP Test Comparison Tab



11. This completes the OpenScript tutorial. Save the script and close the OpenScript application.

3.9 Stopping the Avitek Medical Records Server

When finished with the tutorial be sure to stop the sample Medical Records Server. If you want to try using scripts recorded against the sample Medical Records Server application in Oracle Load Testing or Oracle Test Manager tutorials, leave the Avitek Medical Records Server running until after completing those trials.

To stop the Weblogic Server and Avitek Medical Records Server:

1. Select **Programs** from the **Start** menu and then select **Sample** from the **Oracle Application Testing Suite** menu.
2. Select **Stop MedRec Application**.
3. Wait until the Weblogic server stops.
4. Close the command windows after the server stops.

4

Oracle Load Testing Tutorial

This tutorial walks you through the main features of Oracle Load Testing. Oracle Load Testing is a separate product in the Oracle Application Testing Suite, which you may or may not have purchased. If you have the Oracle Load Testing version of the Oracle Application Testing Suite, you can follow the examples in this chapter to become familiar with the features and use of Oracle Load Testing.

The tutorial consists of the following examples:

- **Performing a Simple Load Test** - shows how to use Oracle Load Testing to run virtual users to simulate load on a Web application.
- **Adding Data Sources** - shows how to add data sources to the Oracle Load Testing ServerStats configuration to monitor server-side statistics, such as CPU usage, and available memory.
- **Editing Data Sources** - shows how to edit existing Oracle Load Testing ServerStats configurations to modify specific counters.
- **Creating a Scenario with Multiple Profiles** - shows how to add a new script to the Oracle Load Testing Scenario. This example also shows how to set the Reporting options and Session Start/Stop options to save data for use in post-run analysis.
- **Running Multiple Profiles** - shows how to use Oracle Load Testing to run multiple Scenario profiles with different amounts of virtual users and how to view statistical and performance information.
- **Controlling Virtual Users** - shows how to modify individual virtual user attributes, view actions, and stop and abort virtual users.
- **Generating Reports** - explains how to view the default reports and generate reports for post-run analysis.

The tutorial is designed to be followed sequentially from beginning to end and assumes you have completed the Oracle OpenScript tutorial in Chapter 3. The examples in this tutorial refer to scripts recorded in the previous tutorials.

4.1 Example 1: Performing a Simple Load Test

This example shows how to use Oracle Load Testing to run virtual users to simulate load on a Web application. The example illustrates how to run a previously recorded script to simulate multiple users accessing a Web application.

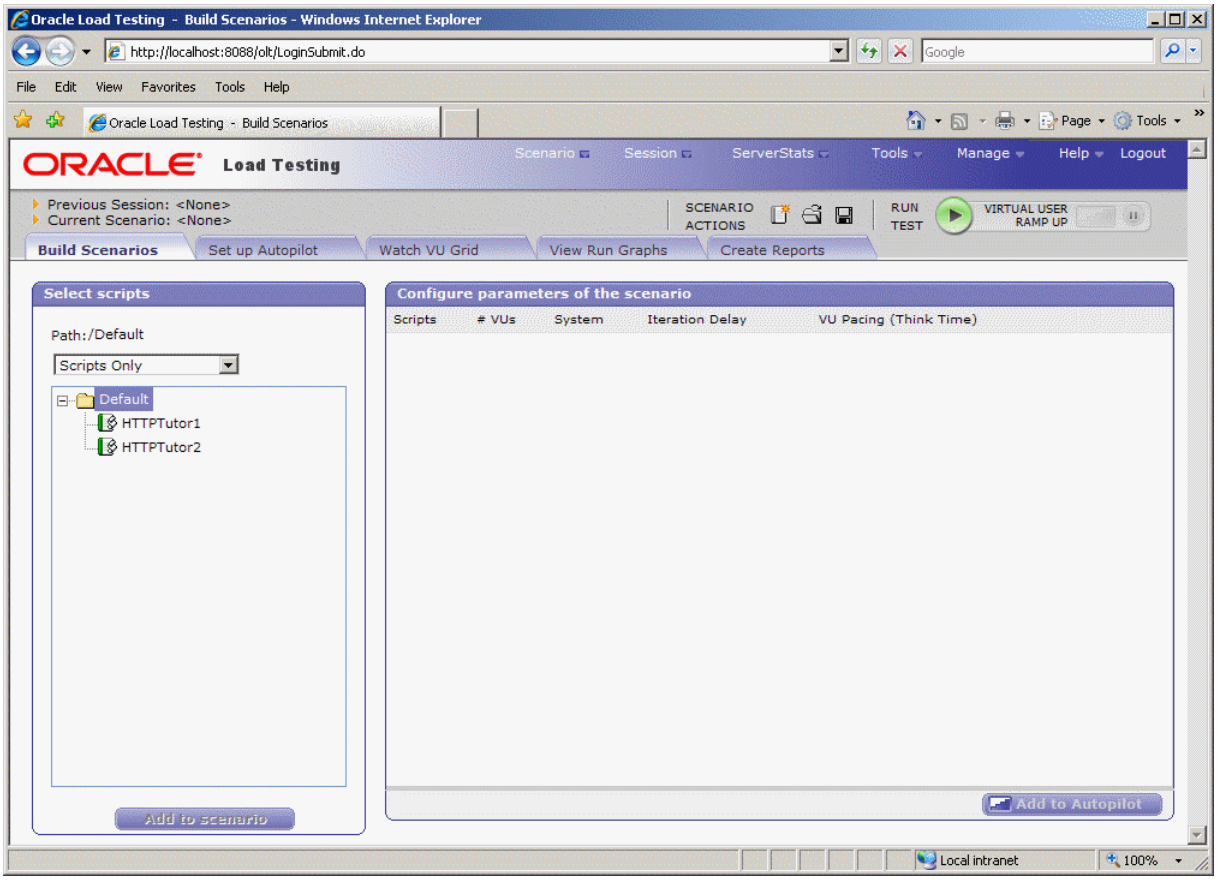
4.1.1 Starting Oracle Load Testing and Specifying the Workspace

> **Note:** This section uses the default login credentials from the Oracle Application Testing Suite installation.

To start Oracle Load Testing:

1. Select **Programs** from the **Start** menu and then select **Oracle Load Testing** from the **Oracle Application Testing Suite** Start menu.
2. Enter **administrator** as the user name.
3. Enter the password specified during the Oracle Application Testing Suite installation process.
4. Click **Login**. The main window appears, as follows:

Figure 4–1 Oracle Load Testing Main Window



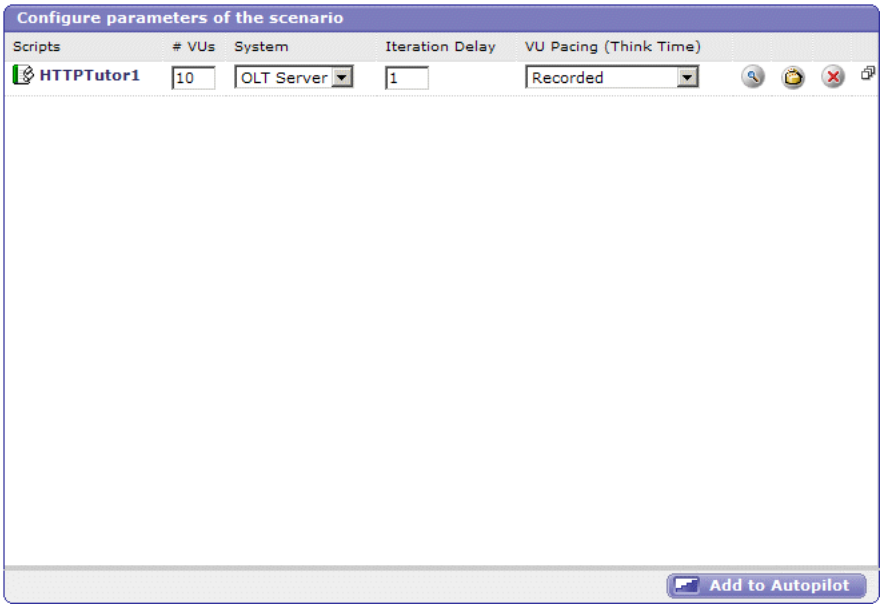
5. Select Default in the Workspace list.

4.1.2 Specifying a Scenario Profile

6. Make sure the **Build Scenario** tab is displayed in Oracle Load Testing.
7. Select HTTPTutor1 in the **Select scripts** list. These are the scripts that you record using Oracle OpenScript. For Oracle OpenScript scripts, only load testing and general scripts appear in the list. Functional test scripts do not.

8. Click the **Add to scenario** button to add HTTPTutor1 to the **Configure Parameters** list. You can also double-click the script name to add it to the **Configure Parameters** list.

Figure 4–2 Configure Parameters Pane

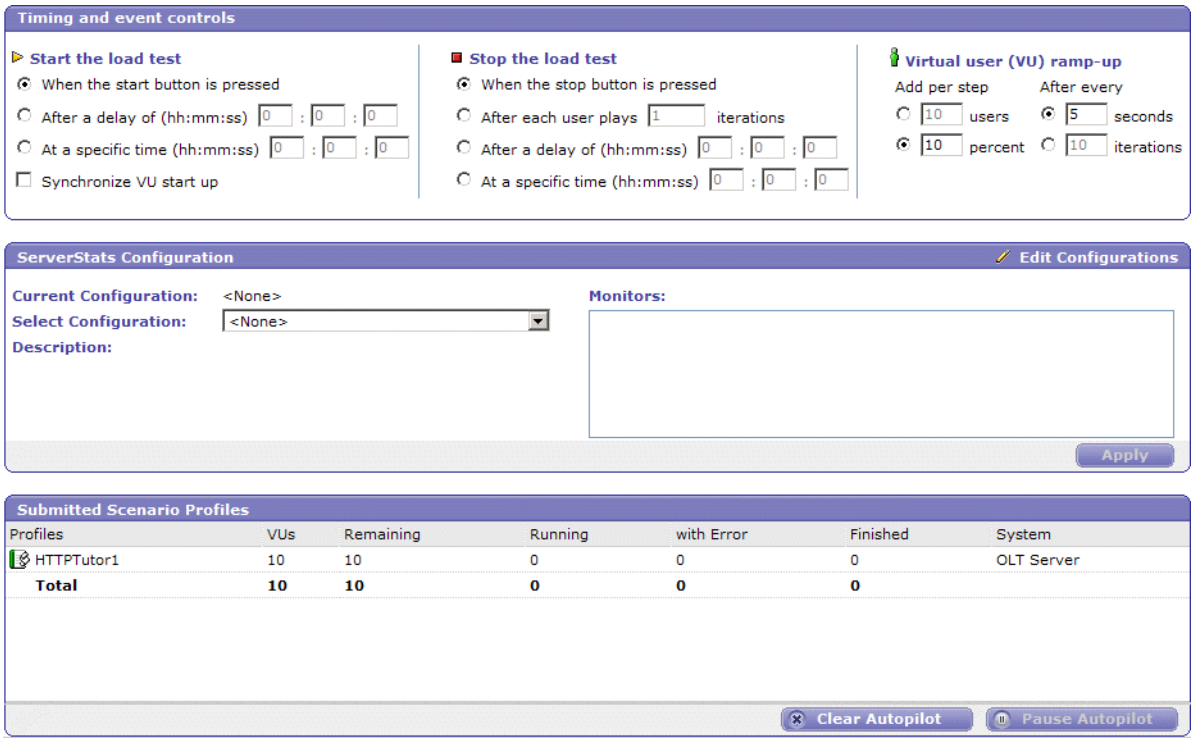


Oracle Load Testing automatically specifies a set of default virtual user attributes for the Scenario Profile in the Scenario tab. For this example, we'll use the default attributes.

9. Click the **Add to Autopilot** button on the Build Scenario tab.

   Oracle Load Testing automatically opens the Set up Autopilot tab with the HTTPTutor1 Scenario Profile listed in **Submitted Scenario Profiles** list.

Figure 4–3 Autopilot Window

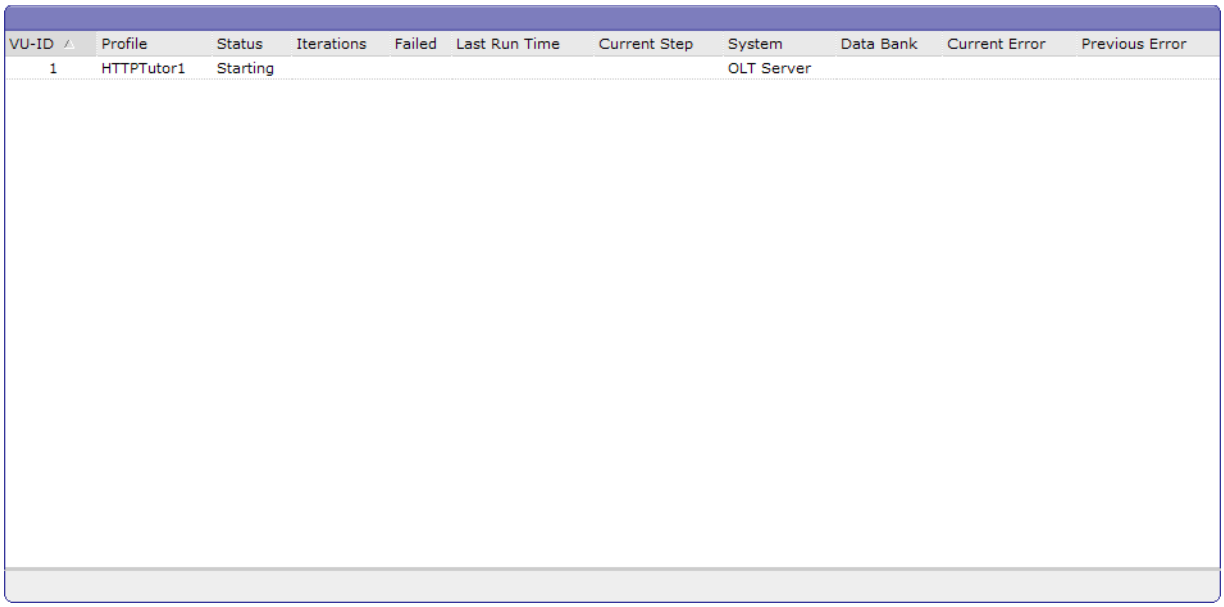


4.1.3 Running the Scenario Profile Using Autopilot

10. Select **After each user plays [#] iteration** option from the **Stop the load test** group of the Autopilot tab.
11. Enter 5 in the **After each user plays** edit box.
12. Select [#] users below **Add per step** in the Virtual User ramp up group.
13. Enter 1 in the **Add per step [#] users** edit box.
14. Select [#] seconds below **After every** in the Virtual User ramp up group.
15. Enter 10 in the **[#] seconds** edit box.
16. Click the **Run Test** button on the Autopilot tab or the toolbar.
17. Select **Yes** when asked to record session data and click **OK**.

    Oracle Load Testing starts running the virtual users in the Virtual User Grid. Watch as the Autopilot starts running the HTTPTutor1 script as ten virtual users.

Figure 4–4 Virtual User Status Grid Window



18. Allow the virtual users to continue running until all of them indicate Finished in the **Status** column of the virtual user grid.

    Congratulations. You have just performed a simple load test on the Demo Web application. Oracle Load Testing performs the virtual user Web interaction in the background. You can monitor the virtual users in the grid as they are running. In the later examples of this tutorial, you'll see how to use Oracle Load Testing to view statistical and performance information, and how to view virtual user actions.

4.2 Example 2: Adding Data Sources

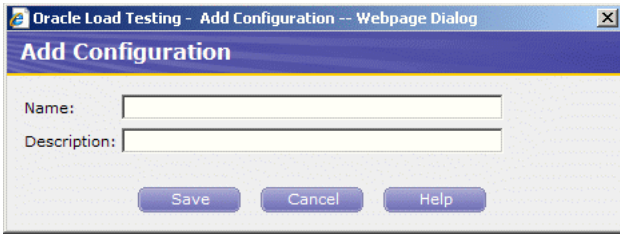
This example shows how to add data sources to the Oracle Load Testing ServerStats configuration to monitor server-side statistics, such as CPU usage, and available memory.

Oracle Load Testing ServerStats can monitor statistics from a variety of systems and server types. This tutorial adds counters from your local Windows 200x/XP system to demonstrate the features of Oracle Load Testing ServerStats. If you are not running the Oracle Application Testing Suite on a Windows 200x/XP machine, you should skip examples two and three and continue the tutorial with example 4.

To configure counters from a Windows 200x/XP data source, do the following:

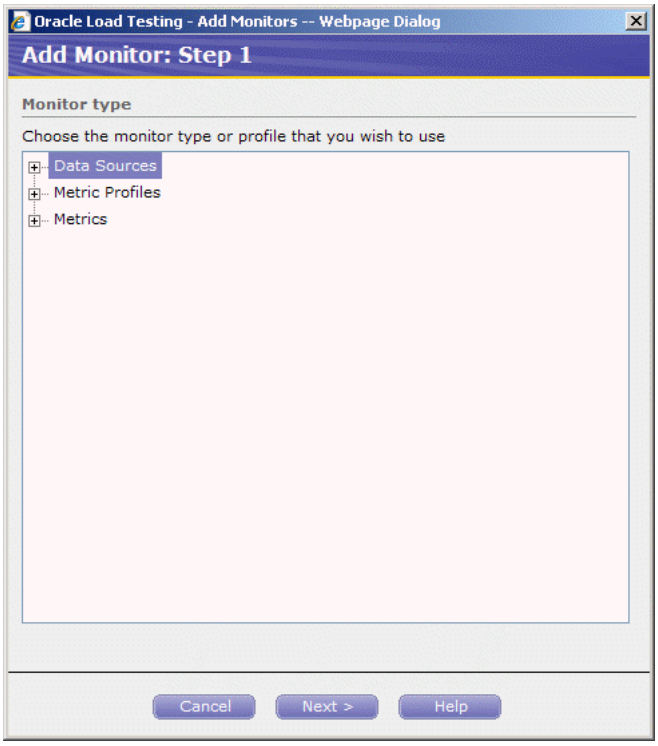
1. Select **Configurations** from the **ServerStats** menu. Oracle Load Testing opens the ServerStats Configurations window.
2. Click **New** to add a new configuration.

Figure 4–5 Add Configuration Dialog Box



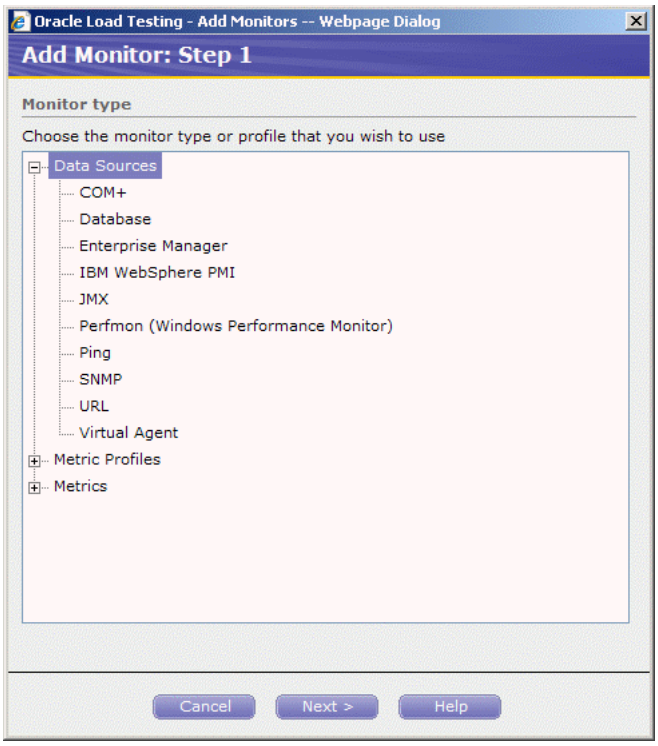
3. Type Tutorial for the Name and the Description.
4. Click **Save**. The Configuration window changes to include the Monitors configuration options.
5. Click **New** in the Monitors pane to open the Add Monitor Step 1 window.

Figure 4–6 Add Monitors Step 1 Dialog Box



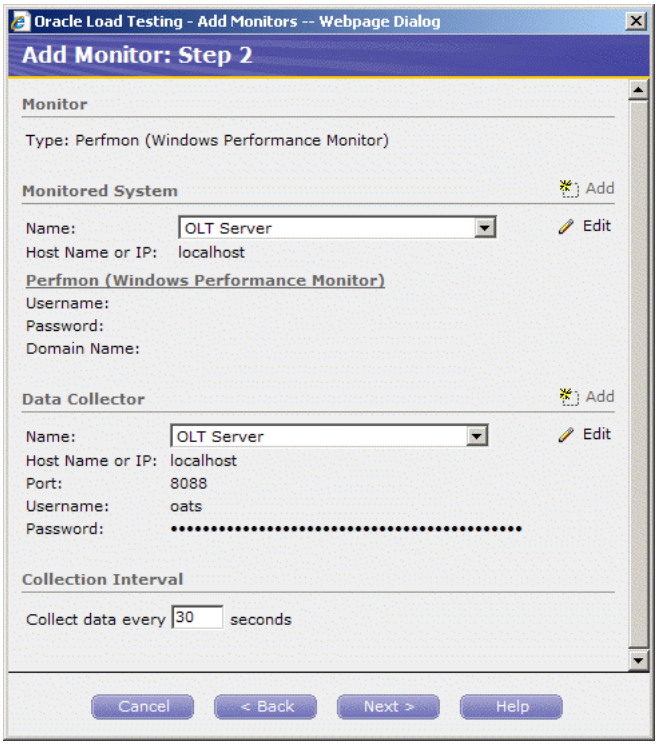
6. Click the Plus icon next to Data Sources to expand the list of data sources.

Figure 4–7 Add Monitors Step 1 with Data Sources Expanded



7. Select Perfmon (Windows Performance Monitor) and click **Next**.

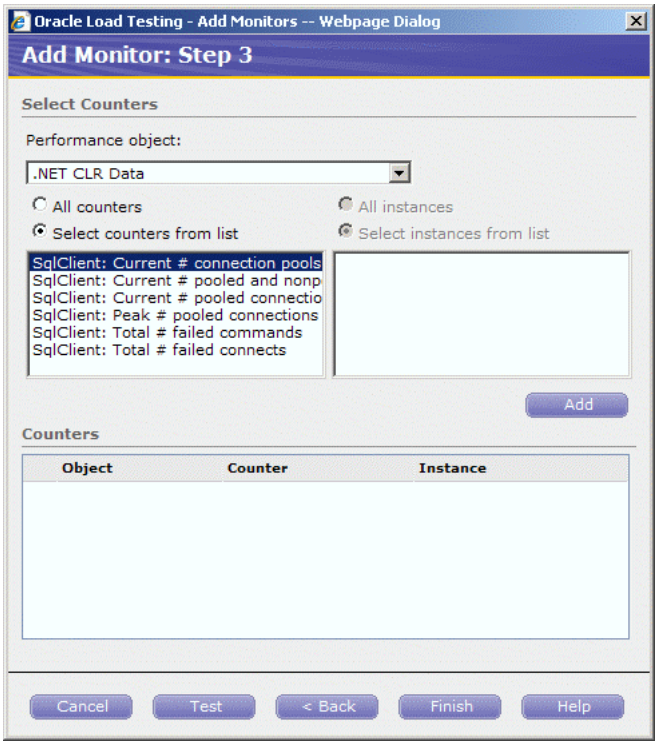
Figure 4–8 Add Monitors Step 2 Dialog Box



This step lets you specify which system to monitor and which system to use for the data collector.

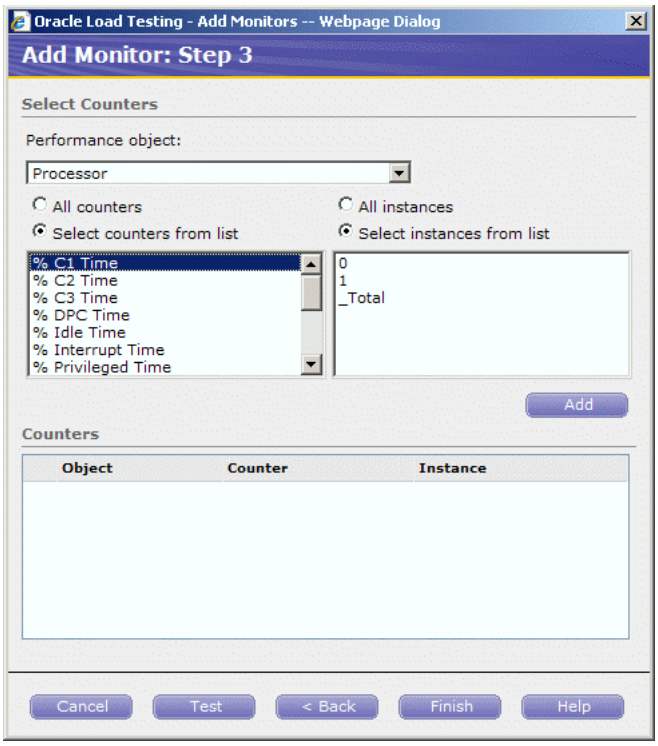
8. Leave the default settings for **Monitored System** and **Data Collector**.
9. Click **Next** to select the specific counters to monitor.

Figure 4–9 Add Monitors Step 3 Dialog Box



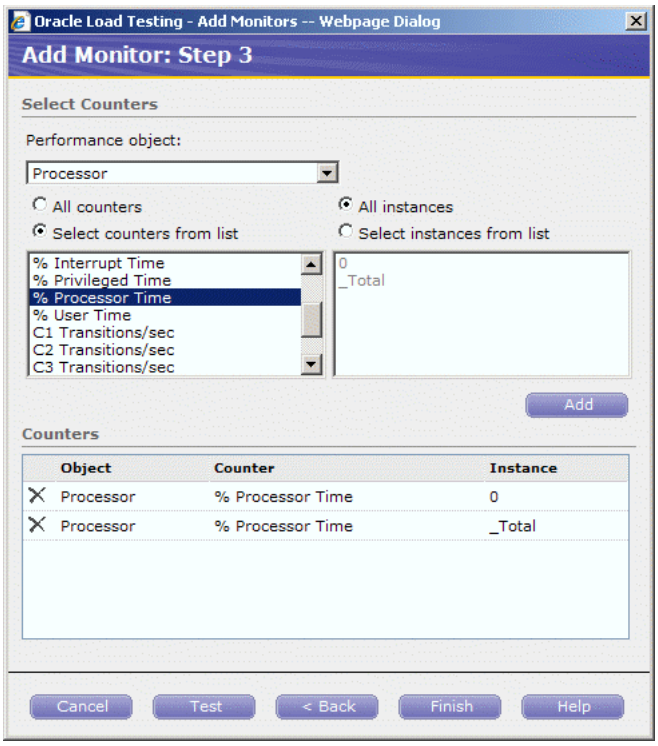
10. Select Processor in the **Performance object** list.

Figure 4–10 Add Monitors Step 3 with Processors Listed



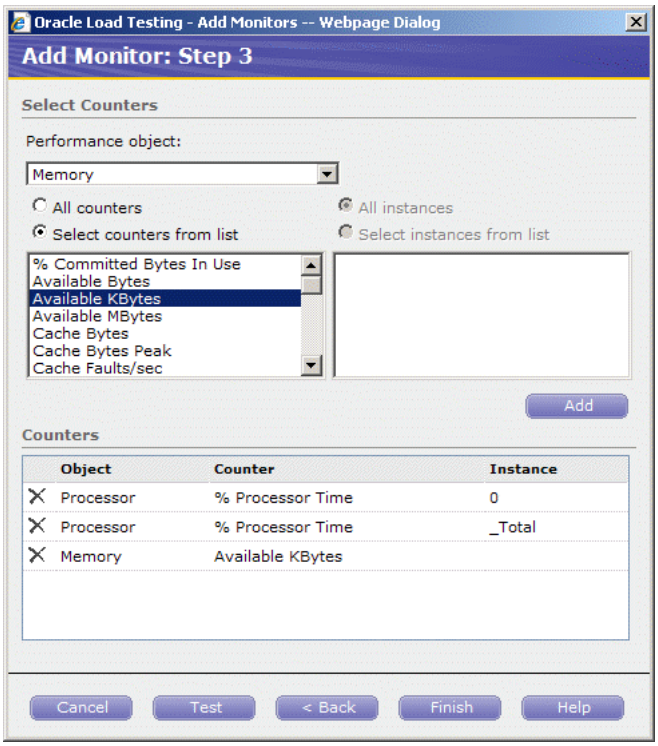
11. Select % Processor Time in the **Select counters from list** group.
12. Select the **All instances** option.
13. Click **Add**. The counters are added to the **Counters** list.

Figure 4–11 Add Monitors Step 3 with Processors Selected



14. Select Memory in the **Performance object** list.
15. Select Available Kbytes and click **Add**.

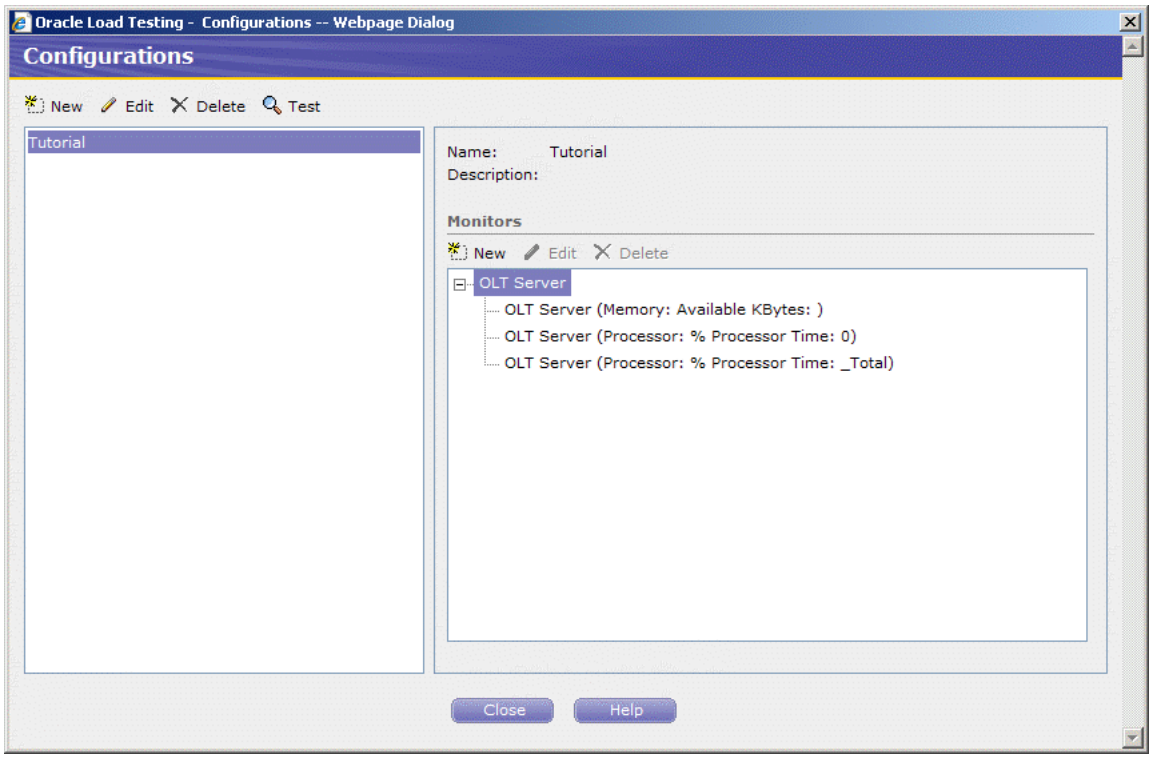
Figure 4–12 Add Monitors Step 3 with Processors and Memory Selected



16. Click **Finish** to complete adding monitors. ServerStats display a status while it verifies the counters (this may take a few moments).

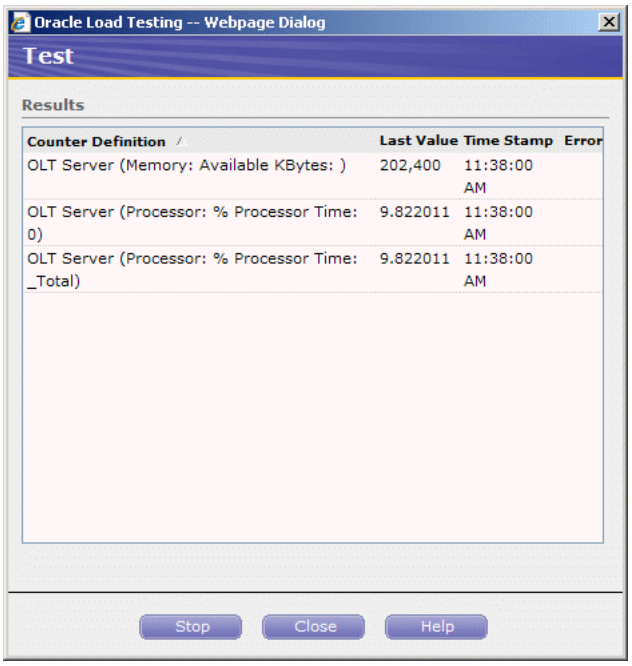
    When the verification is complete, the Configuration window is updated with the list of monitors.

Figure 4–13 Configurations Dialog Box with Defined Monitors



17. Click **Test** to test the counters.

Figure 4–14 Test Monitors Dialog Box



18. Review the results to verify the counters are working properly.
19. Click **Close** to exit the test results.

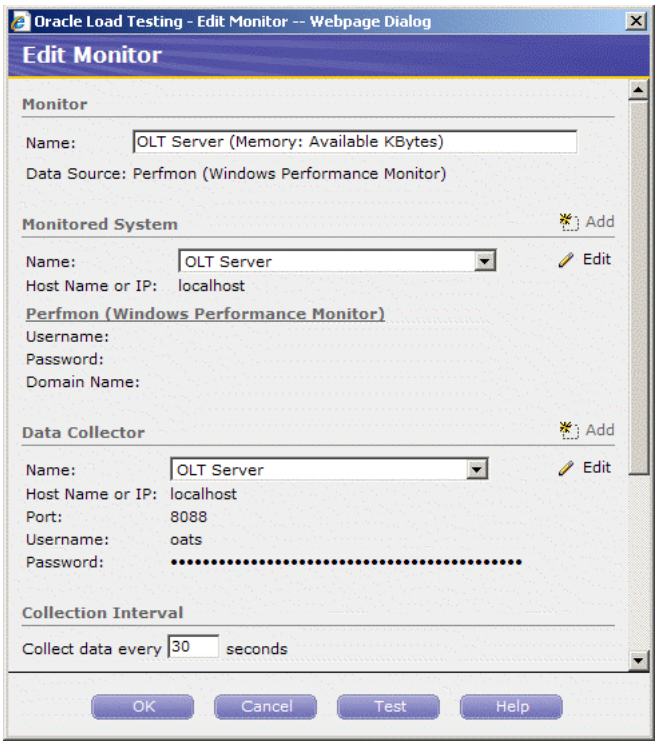
    The next example explains the procedures for editing existing ServerStats configurations.

4.3 Example 3: Editing Data Sources

This example shows how to edit existing Oracle Load Testing ServerStats configurations to modify specific counters. The steps in this example are based upon steps completed in the previous example.

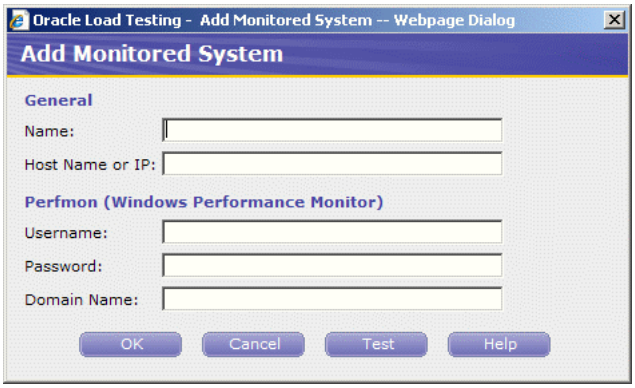
1. If not already open, select **Configurations** from the **ServerStats** menu to open the ServerStats Configurations window.
2. Select the Tutorial configuration.
3. Click the Memory (Available Kbytes) monitor and click **Edit**.

Figure 4–15 Edit Monitor Dialog Box



4. Click **Add** next to **Monitored System**.

Figure 4–16 Add Monitored System Dialog Box



If you have additional systems to monitor you can specify the information here to add the system to the ServerStats Configuration.

5. Click **Cancel**.
6. Change the **Collection Interval** to 45 seconds.
7. Click **OK**.
8. Click **Close** to close the ServerStats Configurations window.

See the *Oracle Load Testing ServerStats Guide* for additional information about using the features and options of Oracle Load Testing ServerStats.

4.4 Example 4: Creating a Scenario with Multiple Profiles

This example shows how to create scenarios with multiple virtual user profiles and how to set the attributes for each scenario. It also shows how to specify the reporting options.

4.4.1 Adding a Virtual User Profile to the Scenario

1. Click the **Build Scenarios** tab.
2. Double-click HTTPTutor2 in the **Default Profiles** list to add it to the **Configure parameters of the scenario** list.
3. Click the **Configure all parameters** button on the HTTPTutor2 line to display the Edit Scenario Details dialog box.
4. Change the **#VUs** value to 3.
5. Make sure the **Virtual User Pacing** is set to Recorded and the **Maximum** value is set to 10 seconds.
6. Change the **Caching Emulation** to Repeat User.
7. Make sure the **Use Databanks** field is True.
8. Leave the default settings for remainder of the attributes and click **OK**.
9. Click the **Configure all parameters** button on the HTTPTutor1 line to display the Edit Scenario Details dialog box.
10. Change the **# VUs** value to 6.
11. Make sure the **Virtual User Pacing** is set to Recorded and the **Maximum** value is set to 10 seconds and click **OK**.

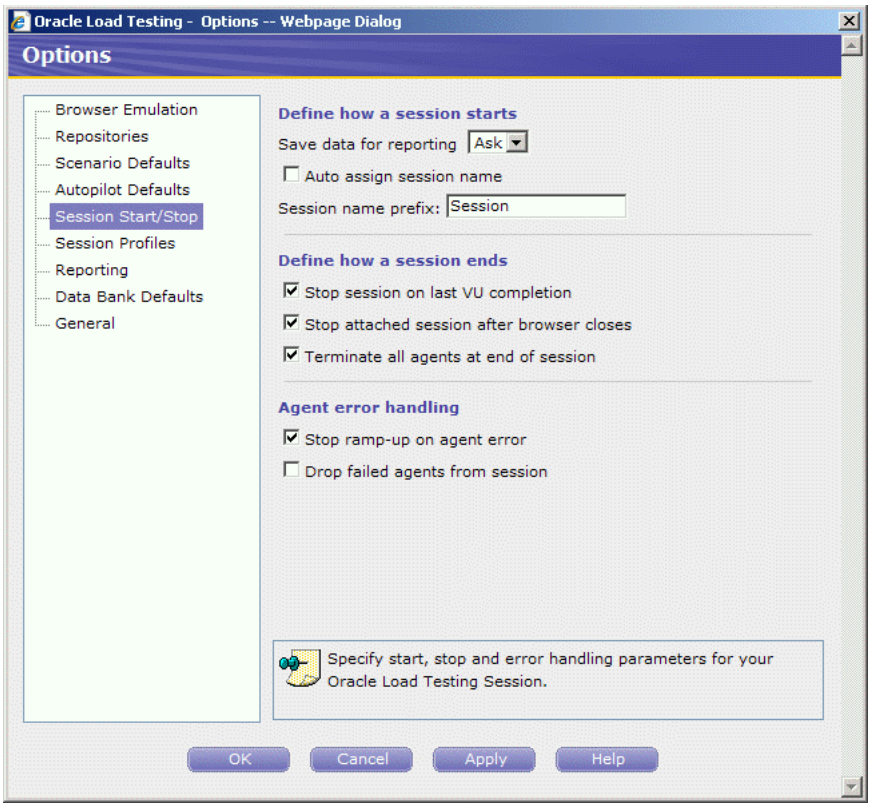
    Notice that each profile in the **Scenario Profiles** list can have a different set of attributes.

4.4.2 Saving Data for Reporting

The data generated by a Oracle Load Testing Autopilot session can be saved to the Oracle Load Testing database for post-session analysis. The Session Start/Stop options let you specify if Oracle Load Testing should save the data.

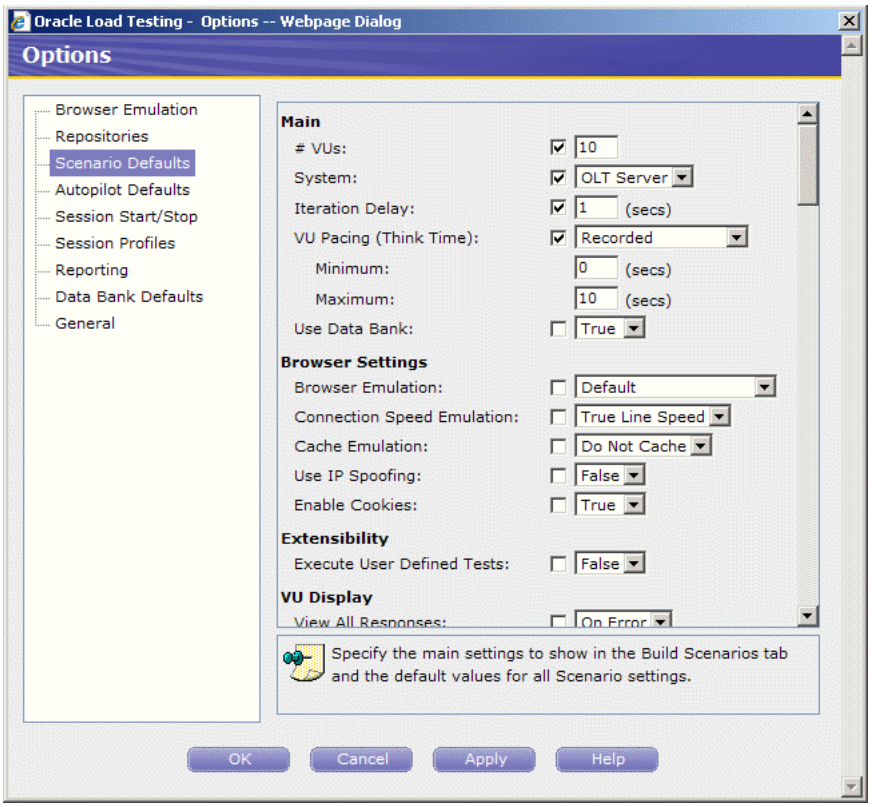
12. Select **Options** from the **Tools** menu and then select **Session Start/Stop**.

Figure 4–17 Session Start/Stop Options Dialog Box



13. Set the **Save data for reporting** option to Ask.
14. Select the **Terminate all agents at end of session** checkbox.
15. Select **Scenario Defaults**.

Figure 4–18 Scenario Defaults Options Dialog Box



16. Set **View All Responses** in the VU Display section to Always.
17. Scroll down the screen and set **Auto generate timers for all resources** in the Reporting section to True.
18. Click **OK**.

4.4.3 Saving the Scenario

19. Select **Save As** from the **Scenario** menu.
20. Specify the filename as LoadTest1 and click **OK**.

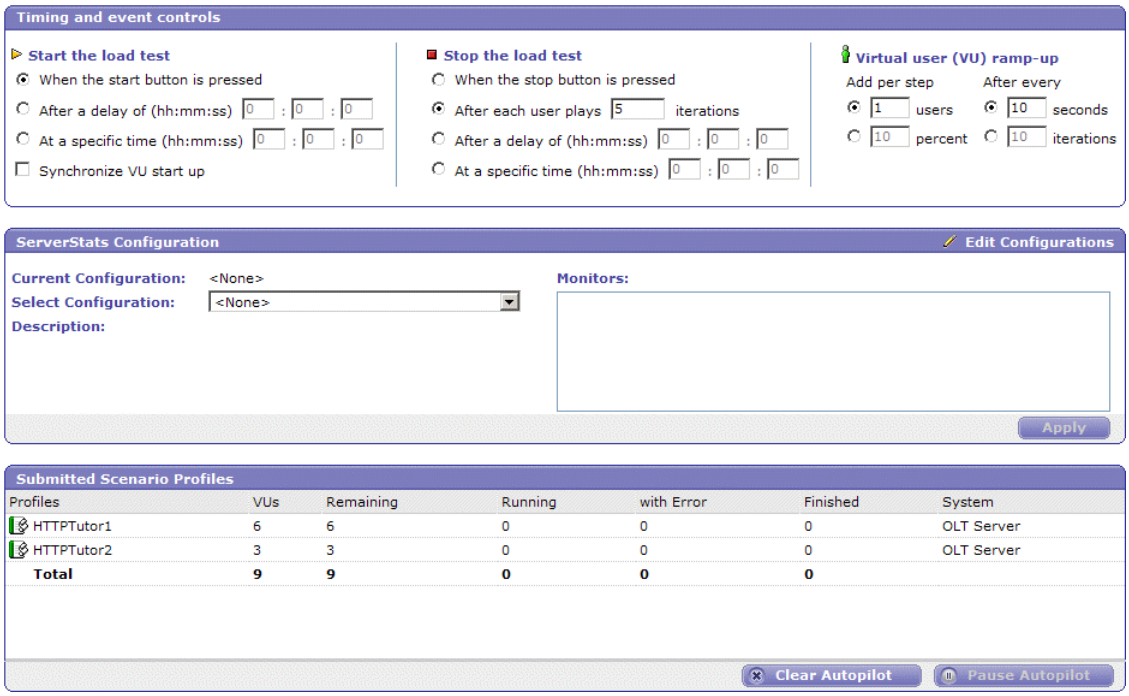
4.5 Example 5: Running Multiple Profiles

This example shows how to use Oracle Load Testing to run multiple Scenario profiles with different amounts of virtual users and how to view statistical and performance information.

4.5.1 Running the Scenario Profiles Using Autopilot

1. Make sure the Scenario from the previous example is still shown in the Build Scenarios tab.
2. Click the **Add to Autopilot** button on the Scenario tab or the toolbar.
3. Oracle Load Testing automatically opens the Set Up Autopilot tab with the HTTPTutor1 and HTTPTutor2 Scenario Profiles listed in **Submitted Scenario Profiles** list.

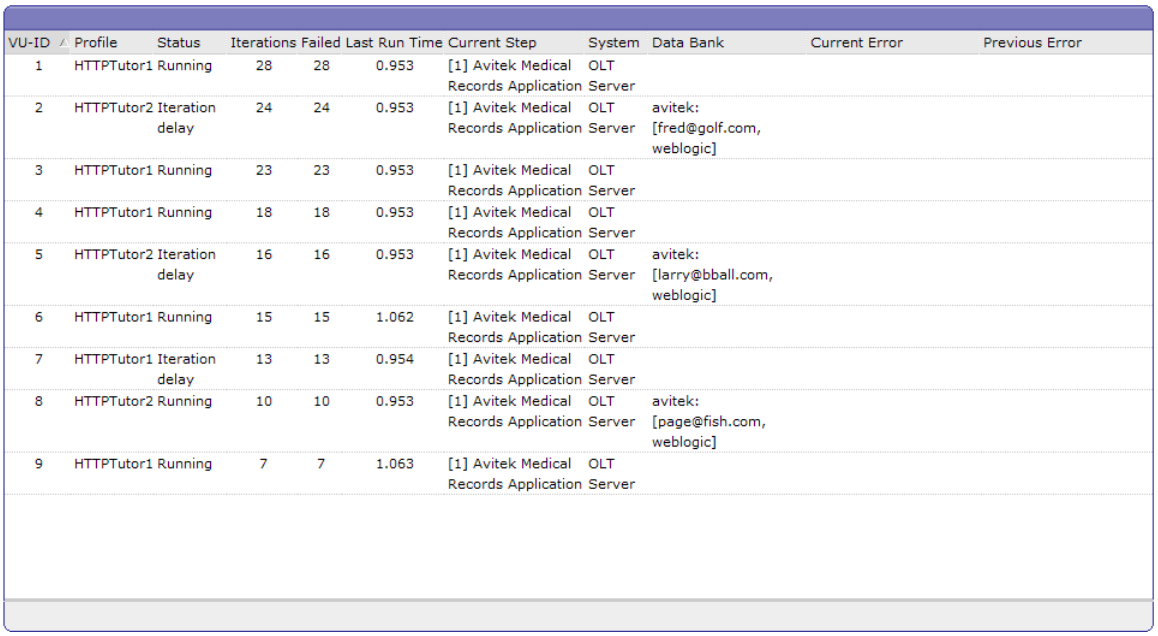
Figure 4–19 Set Up Autopilot Options



4. Select When the stop button is pressed in the **Stop the load test** group of the Set Up Autopilot tab.
5. Enter 3 in the edit box next to **# users** under **Add per step** in the **Virtual User (VU) Ramp-up** section.
6. Enter 5 in the edit box next to **# iterations** under **After every** in the **Virtual User (VU) Ramp-up** section.
7. In **ServerStats Configuration** section, select Tutorial from the Configuration drop down list to add the configuration to the load test.
8. Click **Save** to save the Rampup Specification in the Scenario file.
9. Click the **Run test** button on the Oracle Load Testing toolbar.
10. Oracle Load Testing opens the Save Session data dialog box.
11. Click **Ok**.

    The Save Session data dialog box appears because we used the Ask setting in the **Session Start/Stop** options (select **Options** from the **Tools** menu). You can bypass this dialog box and use automatic or default values when running virtual users under routine testing conditions by changing the **Session Start/Stop** options.
12. Watch as the Autopilot starts running the HTTPTutor1 and HTTPTutor2 scripts as virtual users. Notice also that HTTPTutor2 is playing back records from the Data Bank.

Figure 4–20 Virtual User Grid



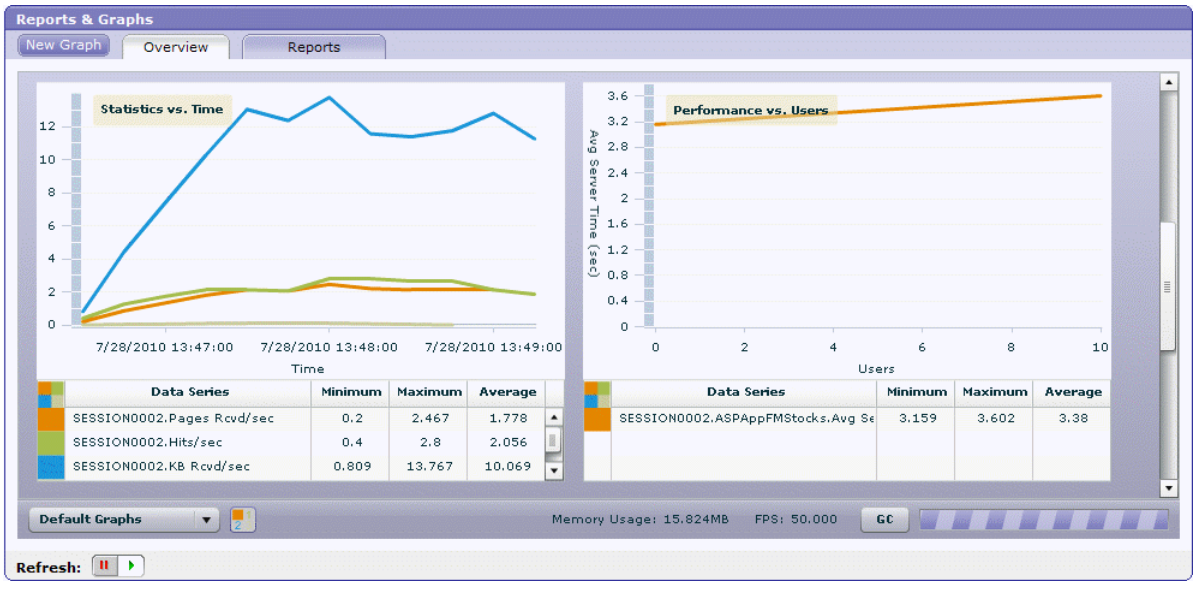
VU-ID	Profile	Status	Iterations	Failed	Last Run Time	Current Step	System	Data Bank	Current Error	Previous Error
1	HTTPTutor1	Running	28	28	0.953	[1] Avitek Medical Records Application	OLT Server			
2	HTTPTutor2	Iteration delay	24	24	0.953	[1] Avitek Medical Records Application	OLT Server	avitek: [fred@golf.com, weblogic]		
3	HTTPTutor1	Running	23	23	0.953	[1] Avitek Medical Records Application	OLT Server			
4	HTTPTutor1	Running	18	18	0.953	[1] Avitek Medical Records Application	OLT Server			
5	HTTPTutor2	Iteration delay	16	16	0.953	[1] Avitek Medical Records Application	OLT Server	avitek: [larry@bball.com, weblogic]		
6	HTTPTutor1	Running	15	15	1.062	[1] Avitek Medical Records Application	OLT Server			
7	HTTPTutor1	Iteration delay	13	13	0.954	[1] Avitek Medical Records Application	OLT Server			
8	HTTPTutor2	Running	10	10	0.953	[1] Avitek Medical Records Application	OLT Server	avitek: [page@fish.com, weblogic]		
9	HTTPTutor1	Running	7	7	1.063	[1] Avitek Medical Records Application	OLT Server			

Initially, the Autopilot starts only three virtual users. After the first three have completed five iterations, the Autopilot starts another three virtual users. Once the second three virtual users have completed five iterations, the remaining three virtual users start. The **Virtual User (VU) Ramp-up** options of the Autopilot let you control the rate at which virtual users start running.

4.5.2 Viewing Performance Statistics

Oracle Load Testing automatically displays run time graphs in the View Run Graphs tab. Click the View Run Graphs tab to view the graphs. The graphs update continuously while the session is running.

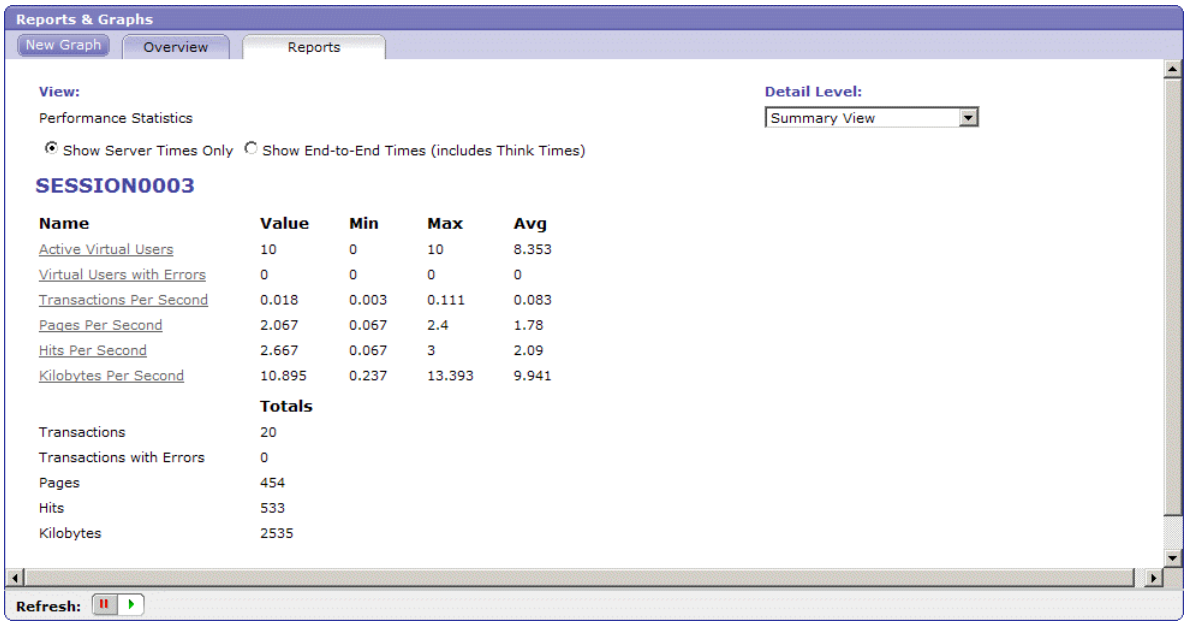
Figure 4–21 View Run Graphs Tab



If necessary, scroll to the bottom of the window until you see the Default Graphs pulldown menu. The pulldown menu lets you select specific graphs to view full scale.

Click on the Reports tab to view the Performance Statistics report. The Performance Statistics report shows a summary of the performance data for the running virtual users.

Figure 4–22 Performance Statistics Report



The statistics show the values for the following performance categories:

<Session Name> Current

- **Active Virtual Users** - the number of virtual users currently running in the Autopilot.
- **Virtual Users with Errors** - the number of virtual users with errors.
- **Transactions Per Second** - the number of times the virtual user played back the script per second.
- **Pages Per Second** - the number of pages returned by the server per second. A "page" consists of all of the resources (i.e. page HTML, all images, and all frames) that make up a Web page.
- **Hits Per Second** - the number of resource requests to the server per second. Each request for a page, individual images, and individual frames is counted as a "hit" by Oracle Load Testing. If Oracle Load Testing does not request images from the server (as specified in the Download Manager), images are not included in the hit count. The **Hits Per Second** and **Pages Per Second** counts will be the same if images are not requested and there are no frames in the page.
- **Kilobytes Per Second** - the number of kilobytes transferred between the server and browser client per second.

<Session Name> Totals

- **Transactions** - the total number of times the virtual user played back the virtual user profile.

- **Transactions with Errors** - the total number of virtual user profile iterations that had errors.
- **Pages** - the total number of number of pages returned by the server.
- **Hits** - the total number of resource requests to the server.
- **Kilobytes** - the total number of kilobytes transferred between the server and browser client.

Performance by Profile and Timer

- **<Profile Name>** - the latest, minimum, maximum, and average performance for the virtual user profile in seconds.
- **<Timer Name>** - the latest, minimum, maximum, and average performance for the server response timers in seconds. Server Response timers are added to scripts using Oracle OpenScript.

Performance by Profile and VUs

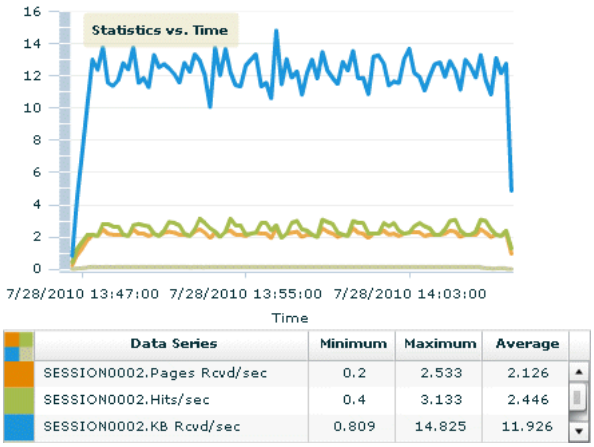
- **<Profile Name> # VUs** - shows the time it took to run the virtual user profile with the indicated number of virtual users running. When ramping up virtual users, Performance by Profile and VUs values are added when additional virtual users start running. Once additional Performance by Profile and VUs values are added, the previous Performance by Profile and VUs values are no longer updated. For example, the statistics show elapsed time values for each profile for three, six, and nine virtual users. The <profile name> 3 VUs values are updated only while three virtual users are running. Once the Autopilot ramps up to run six virtual users, the <profile name> 3 VUs values stop updating and the <profile name> 6 VUs values are added and are updated while six virtual users are running. Once the Autopilot ramps up to run nine virtual users, the <profile name> 6 VUs values stop updating and the <profile name> 9 VUs values are added and are updated while nine virtual users are running.

4.5.3 Viewing Graphs

1. Click the Overview tab in the View Run Graphs tab.

   Oracle Load Testing provides several types of graphs that show performance, error, and statistical information for the running virtual users. The Overview tab shows a summary of the default graphs on in a single view. Clicking the Show Chart Statistics button at the bottom of the Overview tab shows and hides the data series tables for each graph.

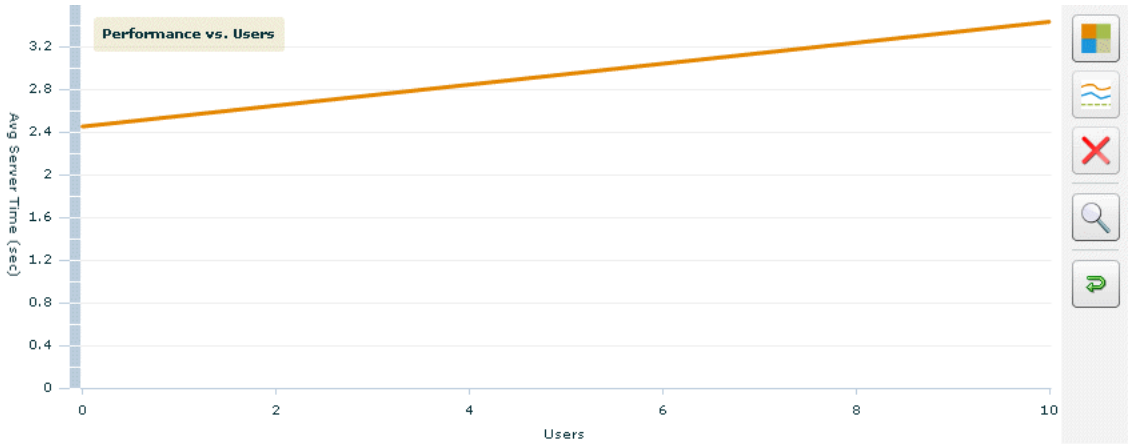
Figure 4–23 Statistics vs. Time Report with Chart Statistics Table Visible



Click on a graph in the Overview tab to view a larger image. Click the Back to Overview button on the right side of the graph to return to the Overview view of the graphs.

2. Select the Performance Vs. Users in the **Default Graphs** dropdown of the Overview tab.

Figure 4–24 Performance Vs. Users Report



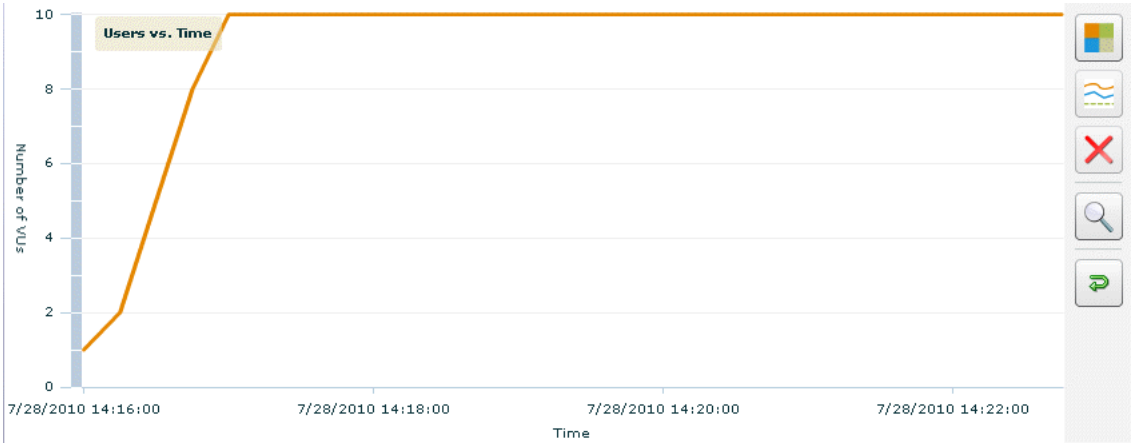
This graph shows the average run time for the number of running virtual users in each profile. The plot points represent the Autopilot rampup of virtual users. In the above graph, the first plot points for each profile shows the average run time while three virtual users were running. Once the Autopilot ramps up to run six virtual users, the plot points for three virtual users are no longer updated.

The second plot points show the average run time while six virtual users were running. Once the Autopilot ramps up to run nine virtual users, the plot points six virtual users are no longer updated.

The third plot points show the average run time while nine virtual users are running. In this example, nine virtual users is the total number of virtual users the Autopilot ramps up to run. The third plot points will be updated continuously while the nine virtual users are running.

3. Select the Users Vs. Time in the **Default Graphs** dropdown of the Overview tab.

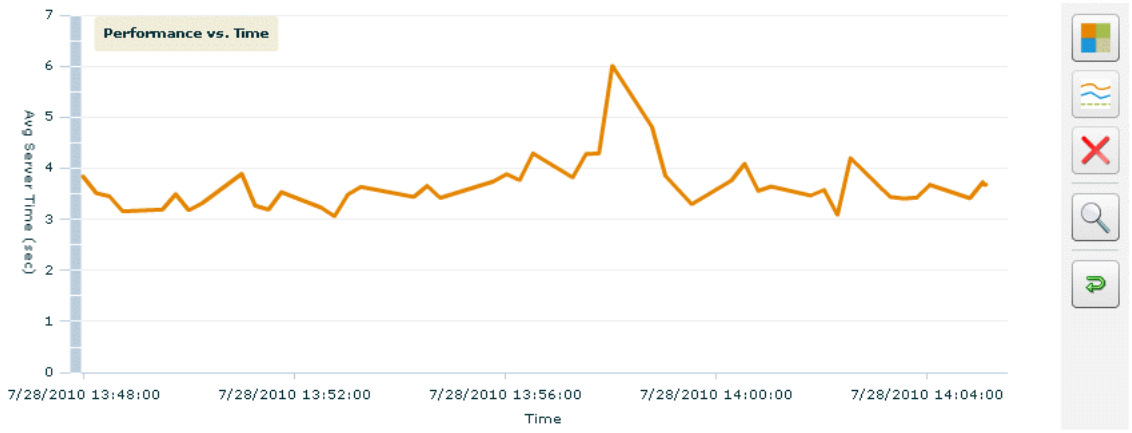
Figure 4–25 Users Vs. Time Report



This graph shows the relative time when the virtual users for each profile started running. The graph represents the Autopilot ramp up times and the number of virtual users ramped up for each profile.

4. Select the Performance Vs. Time in the **Default Graphs** dropdown of the Overview tab.

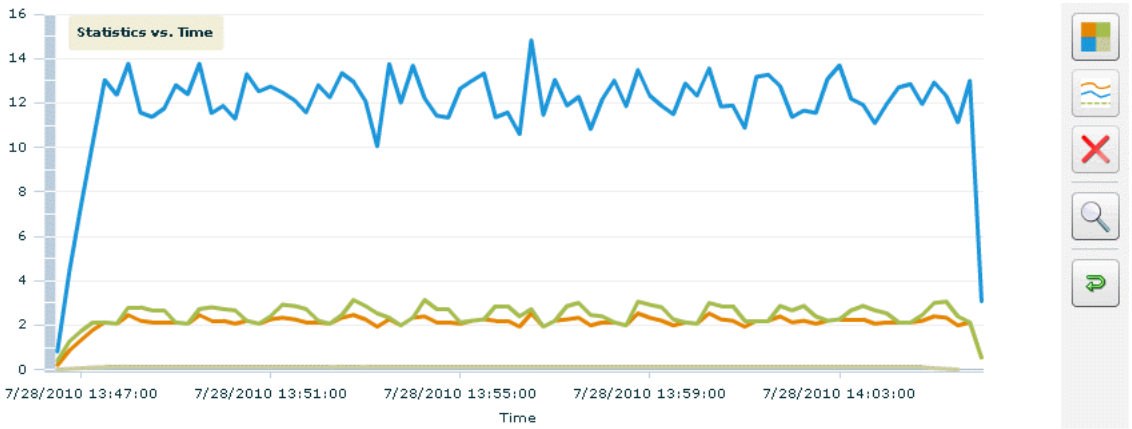
Figure 4–26 Performance Vs. Time Report



This graph shows the average run time for the active virtual users running each profile over time.

5. Select the Statistics Vs. Time in the **Default Graphs** dropdown of the Overview tab.

Figure 4–27 Statistics Vs. Time Report



This graph shows averages for virtual user hits, pages, transactions, and Kilobytes per second over time.

The error graphs show percentages of errors vs. virtual users over time.

4.6 Example 6: Controlling Virtual Users

This example shows how to modify individual virtual user attributes, view actions, and stop and abort virtual users in Oracle Load Testing.

1. Make sure the virtual users from Example 3 are still running.

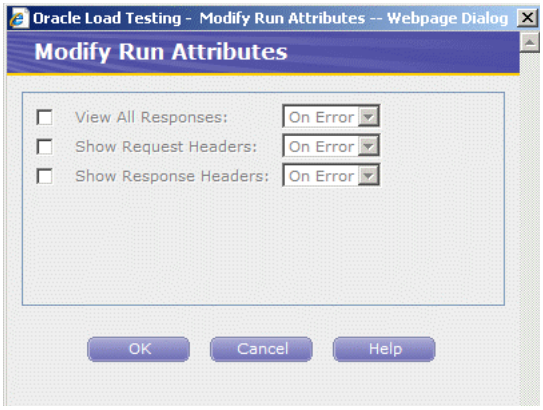
4.6.1 Modifying the Run Attributes

2. Click on any virtual user in the virtual user grid.
3. Click the right mouse-button to open the popup menu.

Figure 4–28 Virtual User Shortcut Menu

4. Select **Modify Run Attributes**. Oracle Load Testing opens a dialog box for changing the run attributes for the selected virtual user.

Figure 4–29 Modify Run Attributes Dialog Box



You can change the attributes of each virtual user individually.

5. Click **Cancel** to close the dialog box.

4.6.2 Viewing Virtual User Actions

6. Select **VU Display** from the **Tools** menu or you can also use the right-click popup menu from the virtual user grid. Oracle Load Testing opens a browser window in which you can view the actions of the virtual user.
7. Click the Navigate to Previous Page toolbar button. The viewer shows only the previous page.
8. Click the Navigate to Next Page toolbar button. The viewer shows only the next page.
9. Click the Auto Mode toolbar button. The view shows new pages accessed by the virtual user as they arrive to the viewer.
10. Click the Stop Accepting New Pages toolbar button. The viewer stops accepting pages from the virtual user.

    Note: Because of the speed at which new pages arrive in the viewer, it may take a few moments for cached pages to stop appearing.
11. Close the window to exit the viewer.

4.6.3 Stopping an Individual Virtual User

12. Click on any virtual user in the virtual user grid.
13. Click the right mouse-button to open the popup menu.
14. Select **Stop**. Oracle Load Testing stops running the selected virtual user. The virtual user will complete the current script iteration and then stop.

4.6.4 Aborting an Individual Virtual User

15. Click on any virtual user in the virtual user grid.
16. Click the right mouse-button to open the popup menu.
17. Select **Abort**. Oracle Load Testing aborts running the selected virtual user without completing the current script iteration.

4.6.5 Stopping All Virtual Users

18. Click the Stop toolbar button to stop all virtual users. The virtual users will complete the current script iteration and then stop.

4.6.6 Aborting All Virtual Users

19. Click the Abort toolbar button to abort all virtual users. The virtual users will abort the virtual user without completing the current script iteration.

4.7 Example 7: Generating Reports

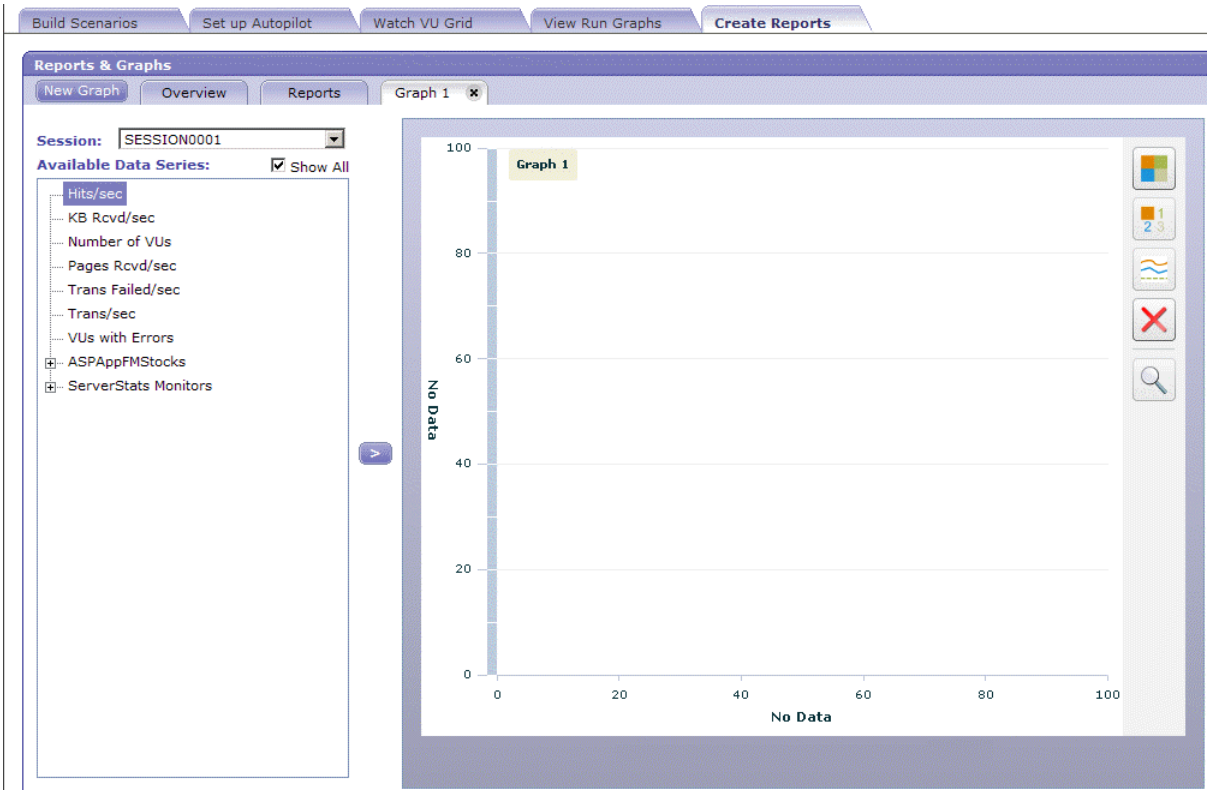
This example explains the automatic report generation features of Oracle Load Testing. The data collected by Oracle Load Testing and Oracle Load Testing ServerStats while the Autopilot is running virtual users is saved to a database when the **Save Data for Reporting** option in the Oracle Load Testing Session Start/Stop options is set to Yes or Ask. You can use Oracle Load Testing to analyze the data and generate a variety of graphs and reports.

4.7.1 Generating Reports from Oracle Load Testing

This example shows how to use previously saved data to create custom reports and to view Session and Scenario reports. The Create Reports tab is used to generate session run reports after a session is finished.

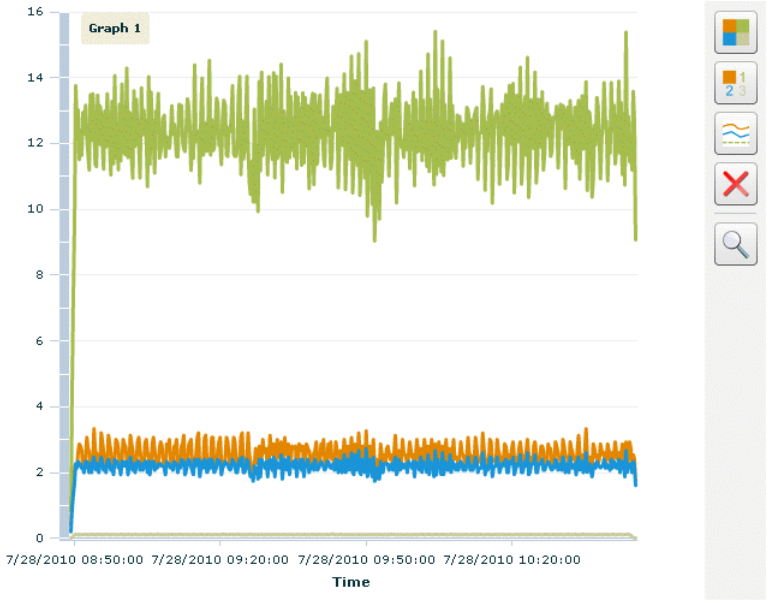
1. Select the **Create Reports** tab.
2. The Graph 1 tab is displayed with a blank graph. Select Session0001 from the **Session** list. The data categories appear in the Available Data Series list.
3. Click **Show All**.

Figure 4–30 Create Reports Tab



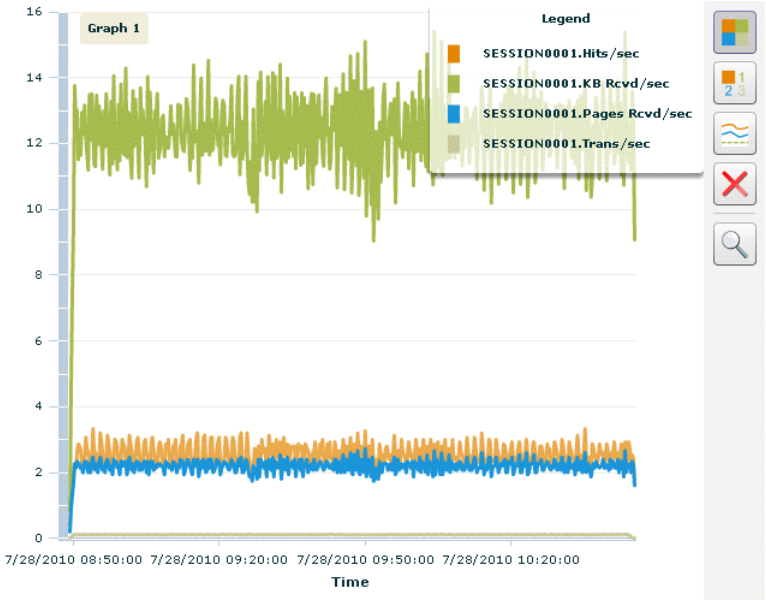
4. Double-click Hits/sec, KB Rcvd/sec, Pages Rcvd/sec, and Trans/sec in the **Available Data Series** field to add the counters to the graph. These are the overall counters. You can also select them and click the **Add Data Series** button.
5. Expand the ServerStats Monitors node then double-click the OLT Server (Processor: % Processor Time: 0) node to add it to the graph.

Figure 4–31 Sample Session Report Graph



6. Click the **Show Legend** button on the right side of the graph. The chart legend view opens over the graph.

Figure 4–32 Sample Session Report Graph Showing the Legend View



The legends show which color line represents which virtual user profile, script page, and Oracle Load Testing ServerStats counter. The legends for Oracle Load Testing data show the session, the virtual user profile, and the scripts page in the form session.profile.page[#]. The legends for ServerStats data show the session, counter object, counter instance and counter in the form session.object.instance.counter.

You can export the graphs to .png and .jpg image files or the data to comma separated value files and Microsoft Excel files.

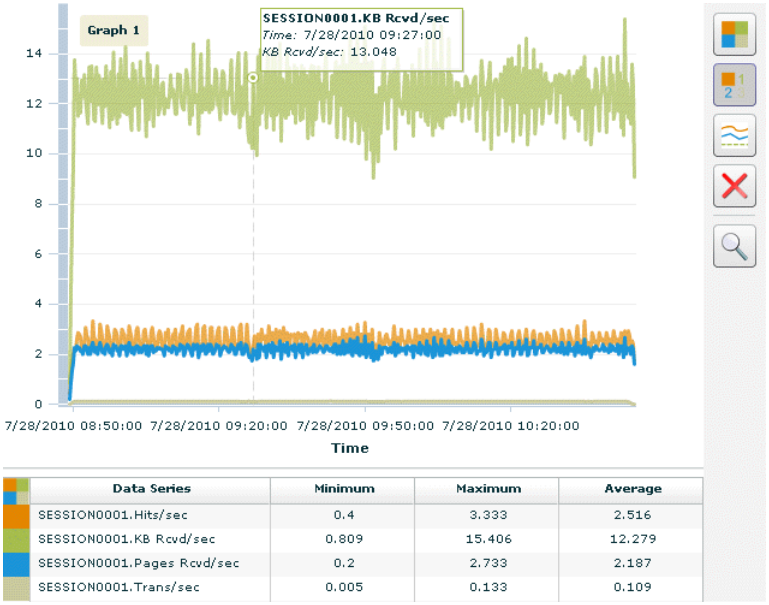
7. Click on a data series once in the legend to hide the data series plot line in the graph. The data series will be dimmed in the legend.
8. Click a hidden data series once in the legend to show the data series plot line in the graph.
9. Click the **Show Legend** button again to hide the legend.
10. Click the **Show Chart Statistics** button on the right side of the graph. The chart statistics table view opens at the bottom of the graph.

Figure 4–33 Sample Session Report Graph Showing the Chart Statistics

Data Series	Minimum	Maximum	Average
SESSION0001.Hits/sec	0.4	3.333	2.516
SESSION0001.KB Rcvd/sec	0.809	15.406	12.279
SESSION0001.Pages Rcvd/sec	0.2	2.733	2.187
SESSION0001.Trans/sec	0.005	0.133	0.109

11. Move the mouse cursor over a data series line in the graph. The data point information popup appears over the graph.

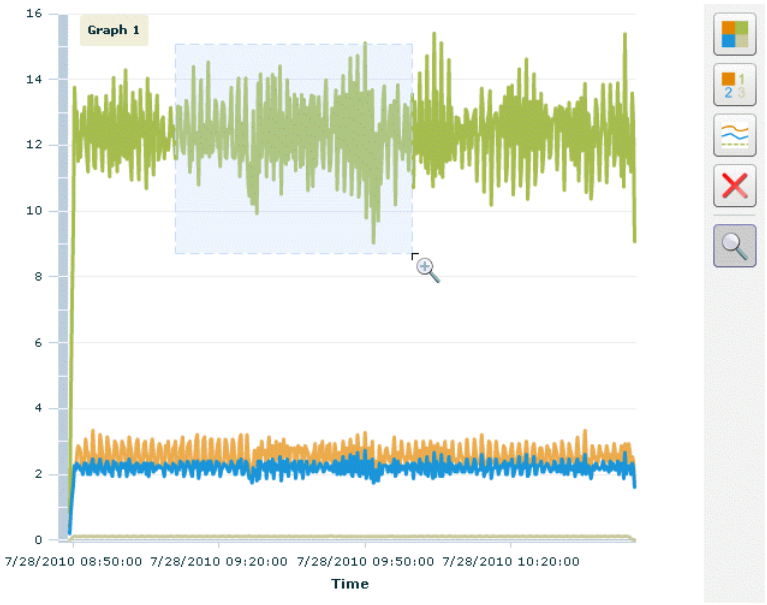
Figure 4–34 Sample Session Report Graph Showing Data Point Information



Data Series	Minimum	Maximum	Average
SESSION0001.Hits/sec	0.4	3.333	2.516
SESSION0001.KB Rcvd/sec	0.809	15.406	12.279
SESSION0001.Pages Rcvd/sec	0.2	2.733	2.187
SESSION0001.Trans/sec	0.005	0.133	0.109

12. Click the **Show Chart Statistics** button again to hide the statistics table.
13. Click the **Zoom** button on the right side of the graph. The cursor changes to a magnifying glass.
14. Click and drag the mouse over a section of the graph to zoom in on that section.

Figure 4–35 Sample Session Report Graph Showing the Zoom Tool



You can zoom in multiple steps by clicking and dragging the Zoom tool repeatedly.

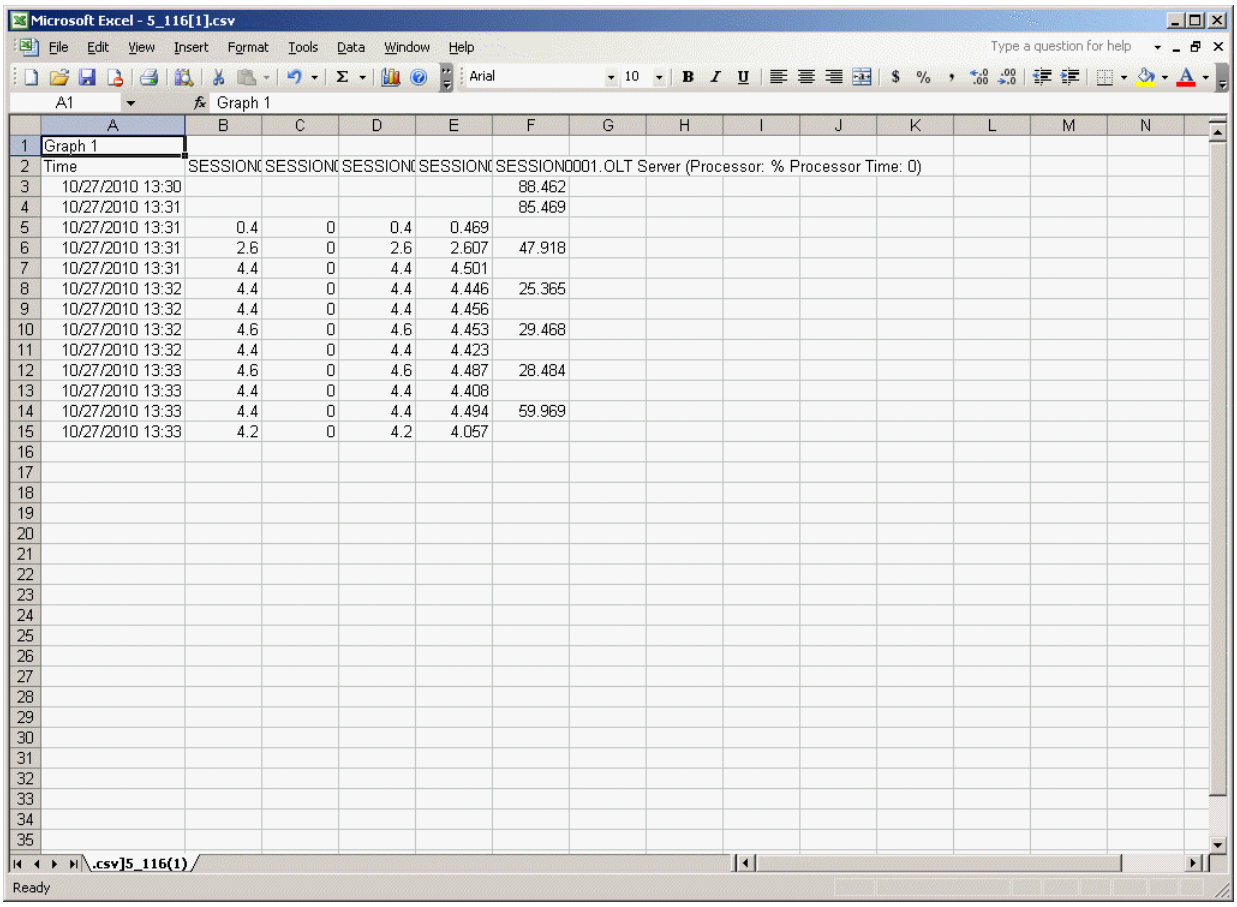
15. Press and hold the Ctrl key and click the left mouse button to zoom out one step. Press and hold the Ctrl key and double-click the left mouse to zoom out to the full graph size.
16. Click the **Zoom** button again to turn off the zoom tool.

4.7.2 Exporting Charts

Note: Skip this section if you do not have Microsoft Excel installed on your system.

17. Click the **Graph Options** button on the right side of the graph. The graph options window opens.
18. Click the **Export** tab.
19. Select the CSV Format option.
20. Click **Export**.
21. The Save As dialog box is displayed. Select a location to save the report and click **Save**.
22. The Download Complete dialog box is displayed. Click **Open**.

Figure 4–36 Sample Excel Report



You can use the Other Export formats and the features and capabilities of Microsoft Excel to work with exported data or charts.

23. Row two contains the counter names. Subsequent rows contain the actual data values for the chart.

24. Select **Exit** from the **File** menu to close Microsoft Excel.

4.7.3 Viewing Scenario and Session Reports

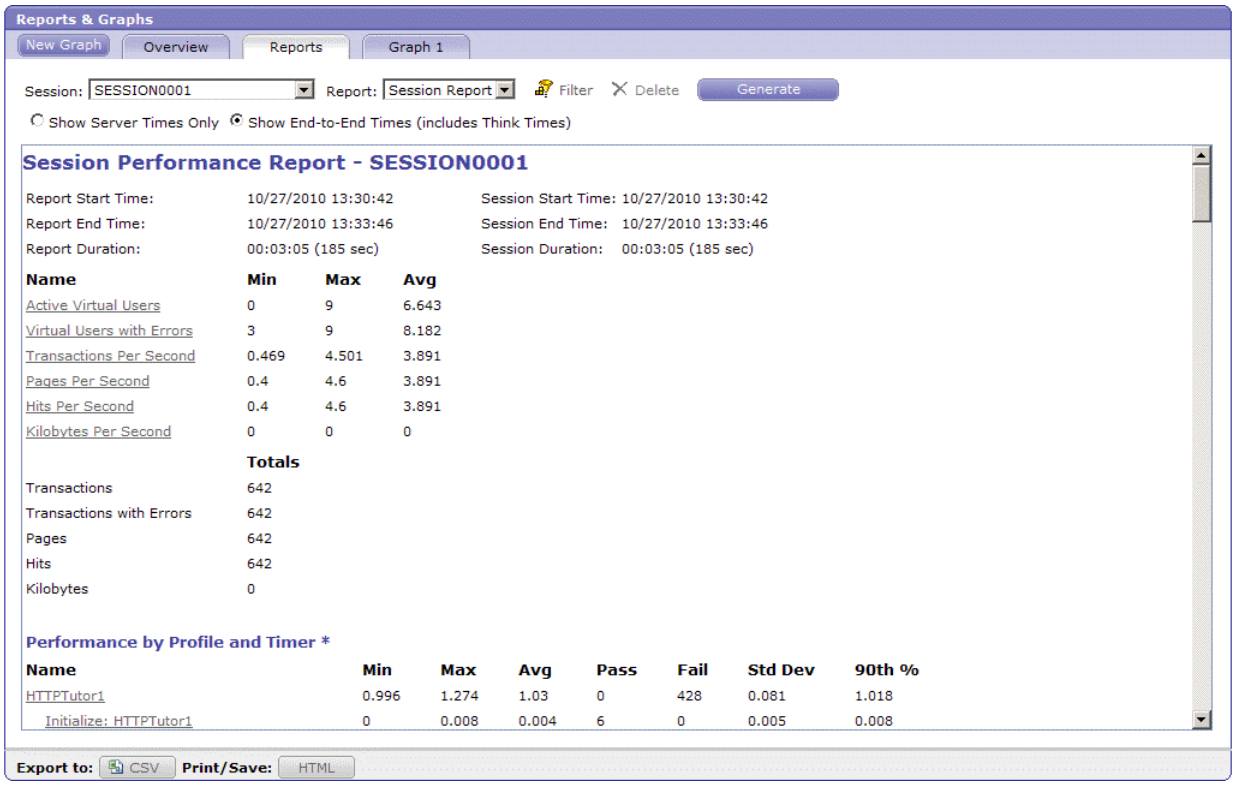
The report shows the current and total performance over time for the Oracle Load Testing scenario. The report also shows the Oracle Load Testing scenario settings used for the session.

25. Oracle Load Testing also generates textual reports for Oracle Load Testing Scenario settings and Oracle Load Testing and Oracle Load Testing ServerStats session data.

26. Select the **Reports** tab.

27. Select the session for which you want to view the report from the **Session** dropdown list and click **Generate**. Oracle Load Testing displays the report.

Figure 4–37 Sample Session Performance Report



28. Scroll the Session Performance Report to view the Oracle Load Testing Scenario Report details.
29. You can print the report by clicking the **Print/Save [HTML]** button and selecting **Print** from the **File** menu in the browser window.

This completes the Oracle Load Testing tutorial. See the *Oracle Load Testing User's Guide* for additional information about load testing and using Oracle Load Testing.

5

Oracle Test Manager Tutorial

This tutorial walks you through the main features of Oracle Test Manager. It consists of the following examples.

- **Adding a Requirement** - describes how to add a requirement.
- **Adding a Test** - describes how to add a both a manual and automated test and how to associate a requirement with a test.
- **Running a Test** - explains how to run both a manual test and an automated test and how to view the results of the automated test.
- **Adding an Issue** - describes how to find and issue, add an issue and associate it with a test.
- **Creating Reports** - explains how to view reports and charts.
- **Using Dashboards** - explains how to view default chart Dashboards and create custom Dashboards.

The tutorial is designed to be followed sequentially from beginning to end. Many of the examples are interrelated and build upon the steps in previous examples.

5.1 Starting Oracle Test Manager

> **Note:** This section uses the default login credentials from the Oracle Application Testing Suite installation.

To start Oracle Test Manager:

1. Select **Programs** from the **Start** menu and then select **Oracle Test Manager** from the **Oracle Application Testing Suite** Start menu.
2. Enter **administrator** as the user name.
3. Enter the password specified during the Oracle Application Testing Suite installation process.
4. Make sure the **Database** is set to Default OTM.
5. Make sure the **View Type** is set to All modules.
6. Click **Login**.

5.2 Opening the Sample Project

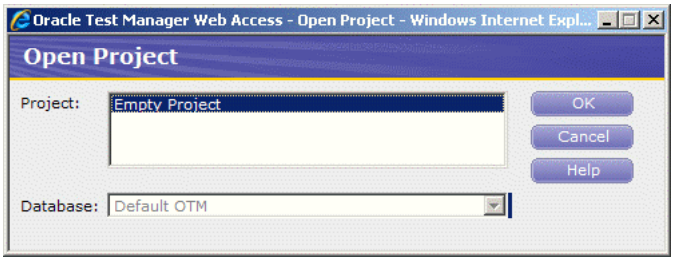
A demonstration project is installed with Oracle Test Manager for testing the sample stock brokerage application Fitch & Mather. This application can be viewed at `http://demo.fmstocks.com`. To open the sample project:

1. Start Oracle Test Manager and log in.

   The default installation user names are **administrator** and **default** unless changed by an administrator. The password is the password specified during the installation procedure unless changed by an administrator.

2. Click **Open** from the **Project** menu to display the Open Project dialog box.

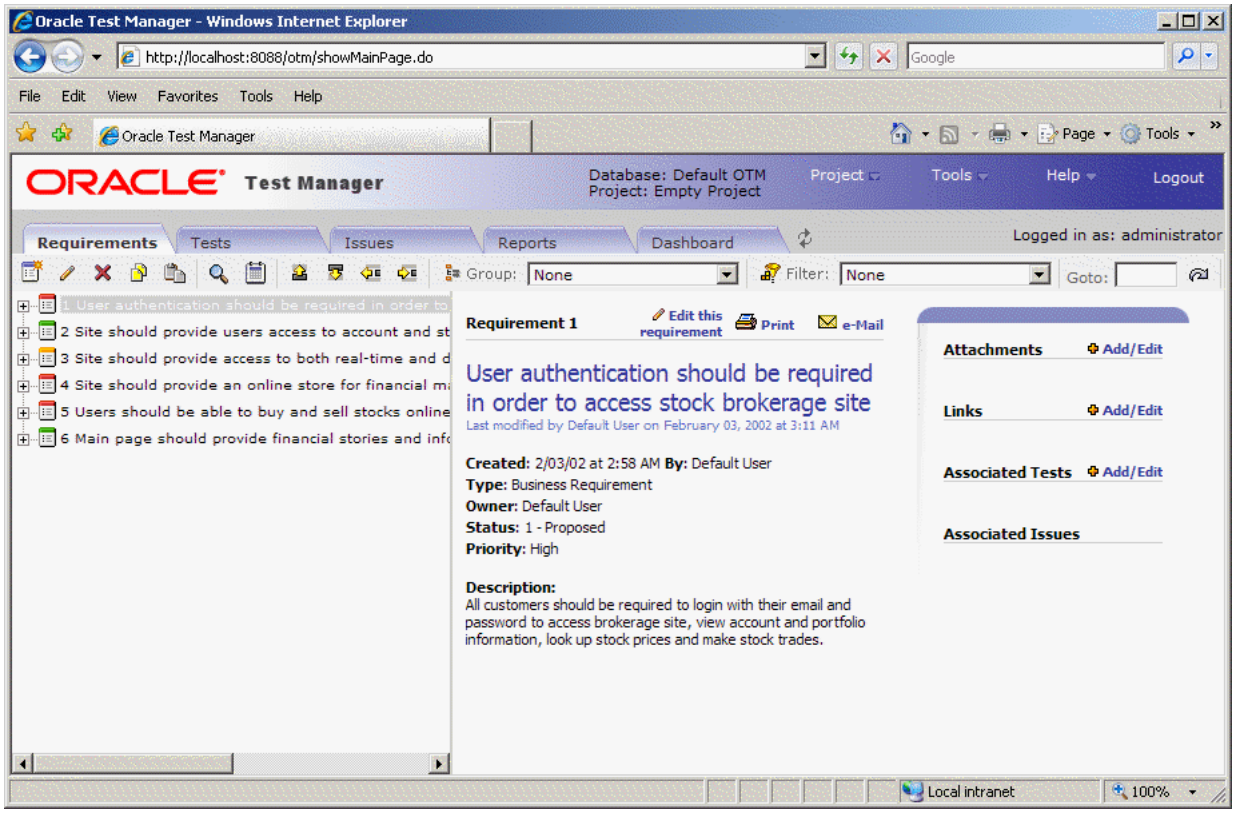
Figure 5–1 Open Project Dialog Box



3. Make sure the **Default OTM** database is selected. This is the database that was created when you installed Oracle Test Manager.

4. Click **OK**. The main window appears as follows:.

Figure 5–2 Oracle Test Manager Main Window



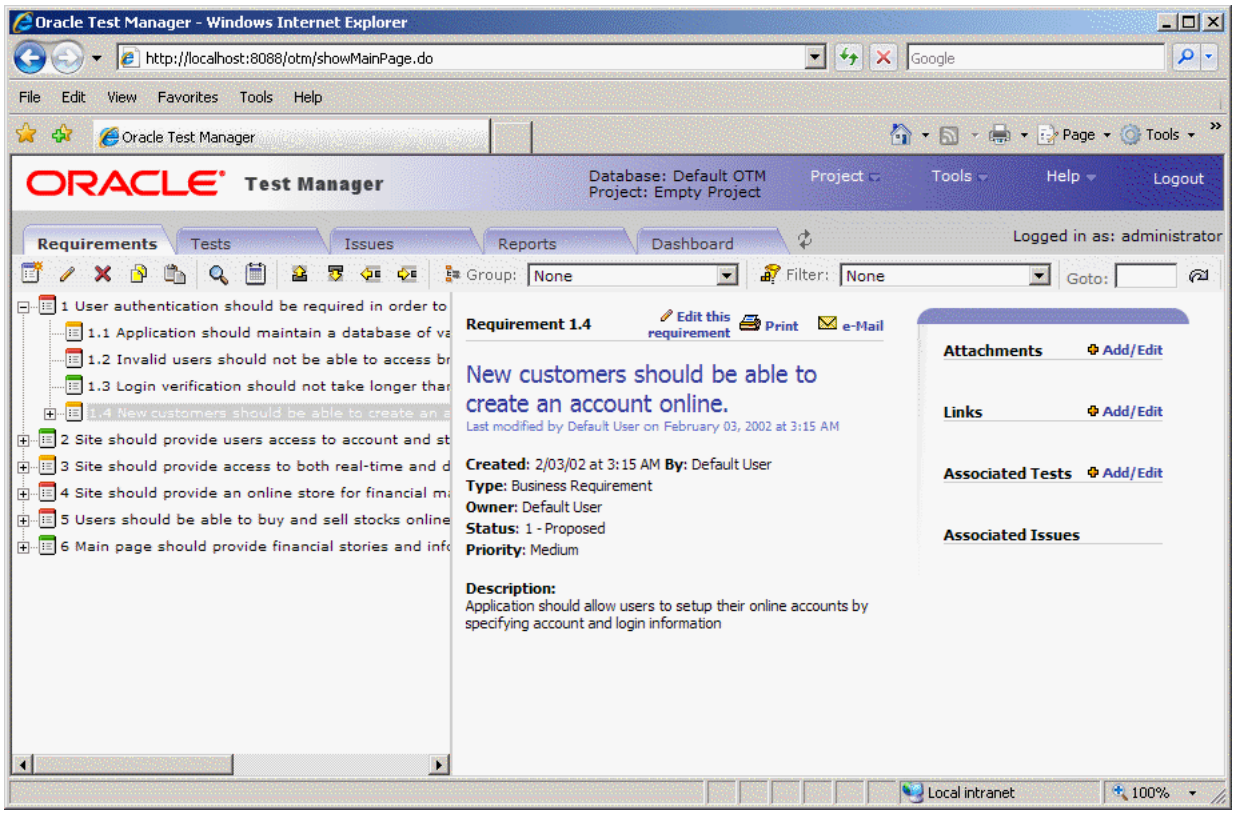
The left pane contains tabs for the modules that comprise the application. They are requirements, tests, issues, reports, and dashboard. The detailed information for the selected item is displayed in the right pane. The information displayed may be different from the examples shown in this tutorial if your system administrator has customized the database to add new fields or disable default fields.

5.3 Example 1: Adding a Requirement

This example explains how to add a requirement.

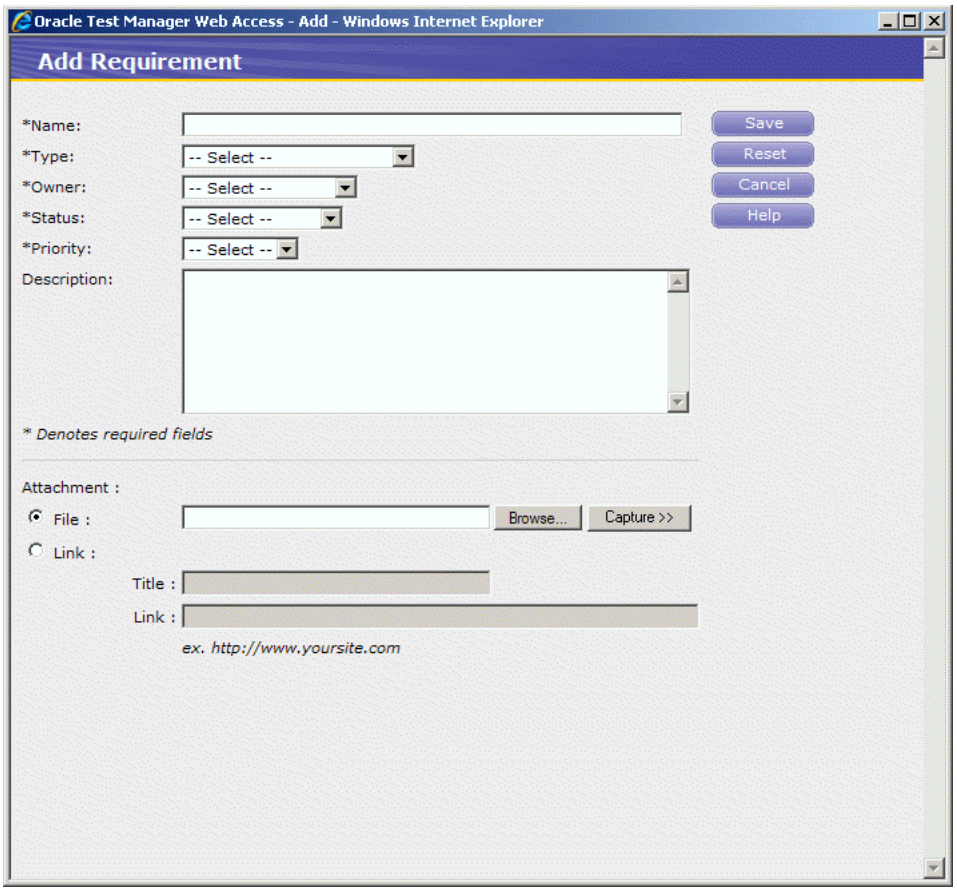
1. Click the **Requirements** tab.
2. Expand item 1 and click item 1.4. The new requirement will be added as item 1.5.

Figure 5–3 Requirements Tab



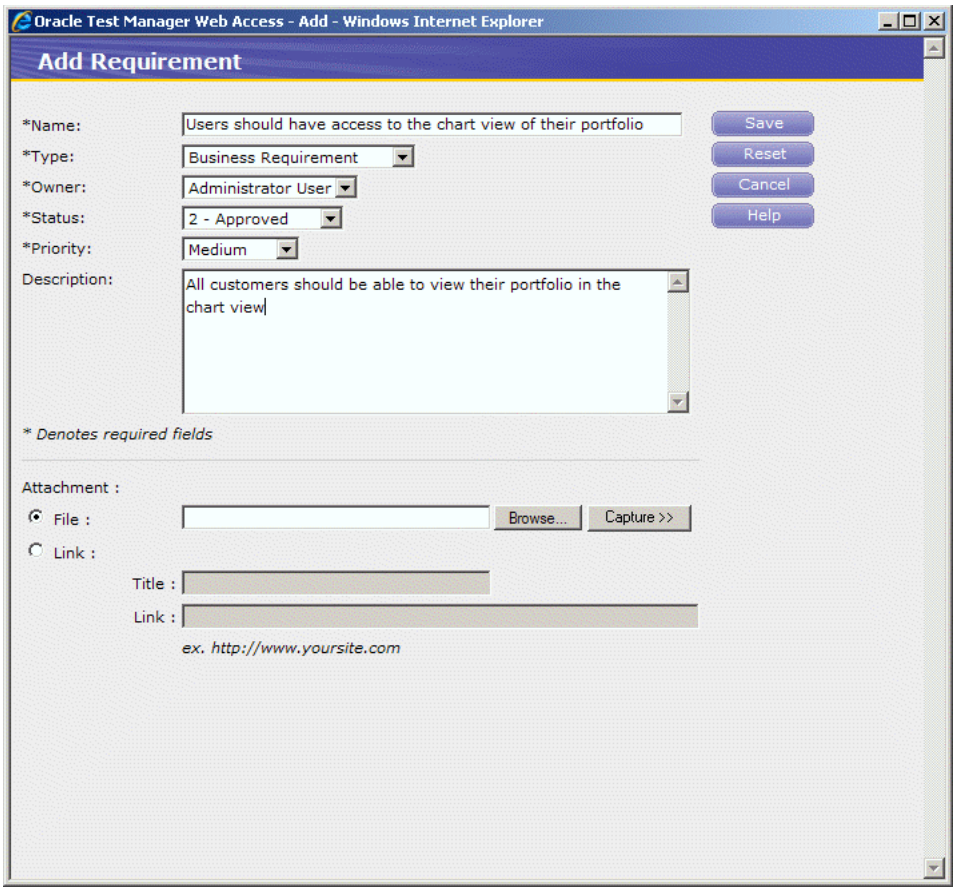
3. Click **Add** to display the Add Requirement dialog box.

Figure 5–4 Add Requirement Window



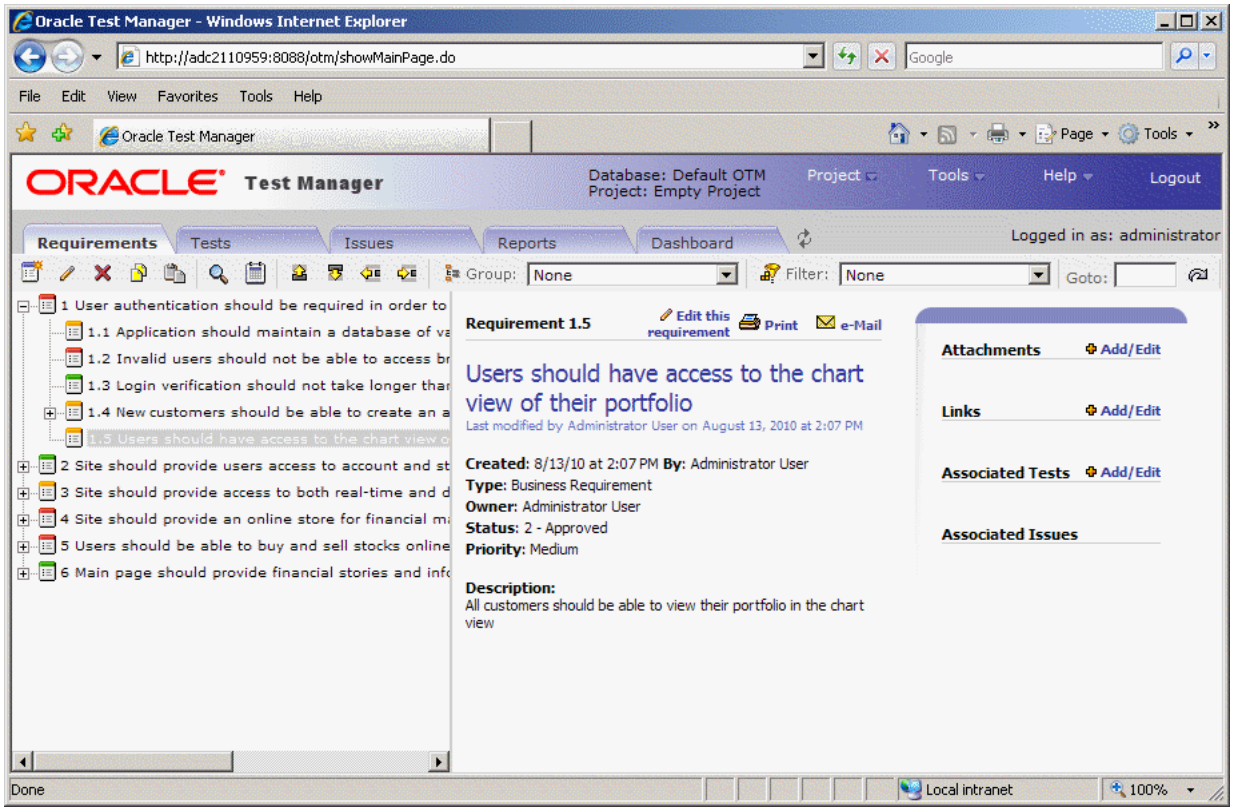
4. Enter "Users should have access to the chart view of their portfolio" in the Name field.
5. Select a type, owner, and status.
6. Select Medium from the Priority list.
7. Enter "All customers should be able to view their portfolio in the chart view" in the Description field.

Figure 5–5 Add Requirement Window with Sample Data



8. Click **Save**. The new requirement appears in the tree.

Figure 5–6 Requirements Tab with New Requirement Added



5.4 Example 2: Adding a Test

You can add two types of test cases to Oracle Test Manager:

Manual tests - these are tests that allow you to organize your test cases. For each step in the test, you enter the action, expected result, and pertinent comments. When you run the test, the Oracle Test Manager Manual Test Wizard takes you step by step through the test, allowing you to manually execute each test and enter the result.

Automated tests - these are Oracle OpenScript tests that can be run automatically without manual intervention. You can run one test, an entire branch of tests, or all automatic tests in the project.

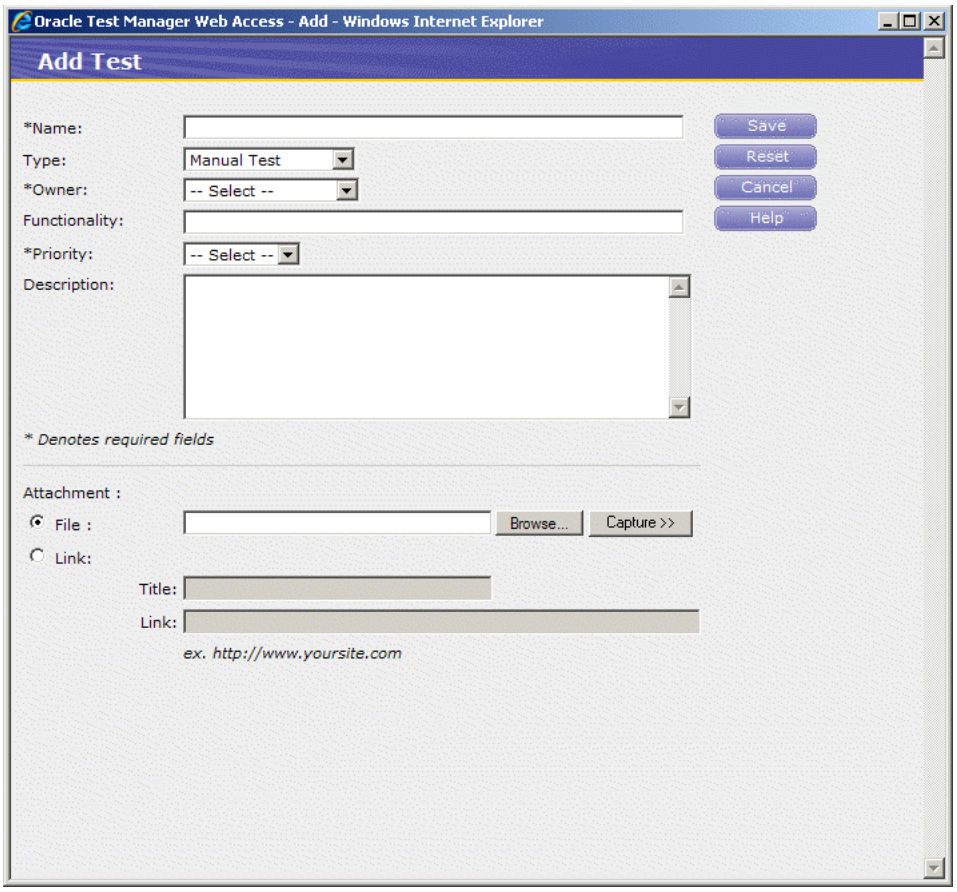
This example has two parts. The first part explains how to add a manual test. The second part explains how to create the same test using Oracle OpenScript and add it as an automated test.

5.4.1 Adding a Manual Test

This example explains how to add a manual test.

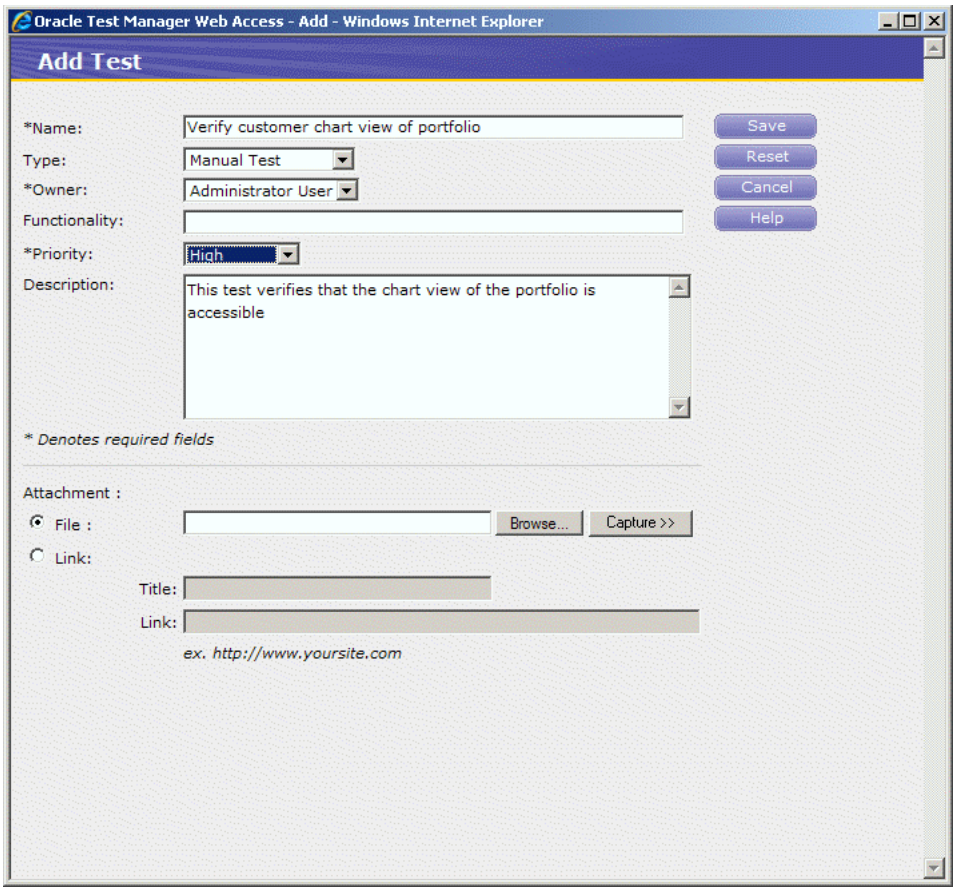
1. Click the **Tests** tab.
2. Click item 1, Login Tests.
3. Click **Add** to display the Add Test dialog box.

Figure 5–7 Add Test Window



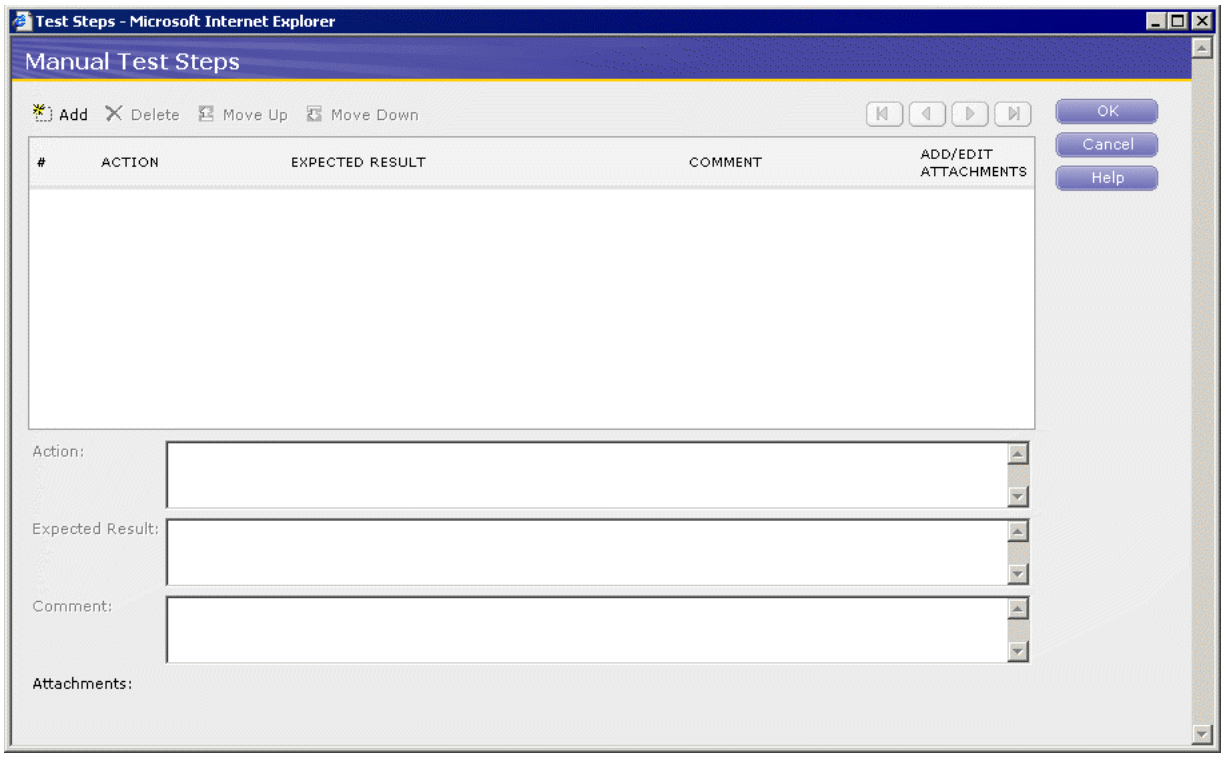
4. Enter "Verify customer chart view of portfolio" in the **Name** field.
5. Select **Manual Test** in the **Type** field.
6. Enter "This test verifies that the chart view of the portfolio is accessible" in the **Description** field.
7. Select an owner and priority.

Figure 5–8 Add Test Window with Sample Data



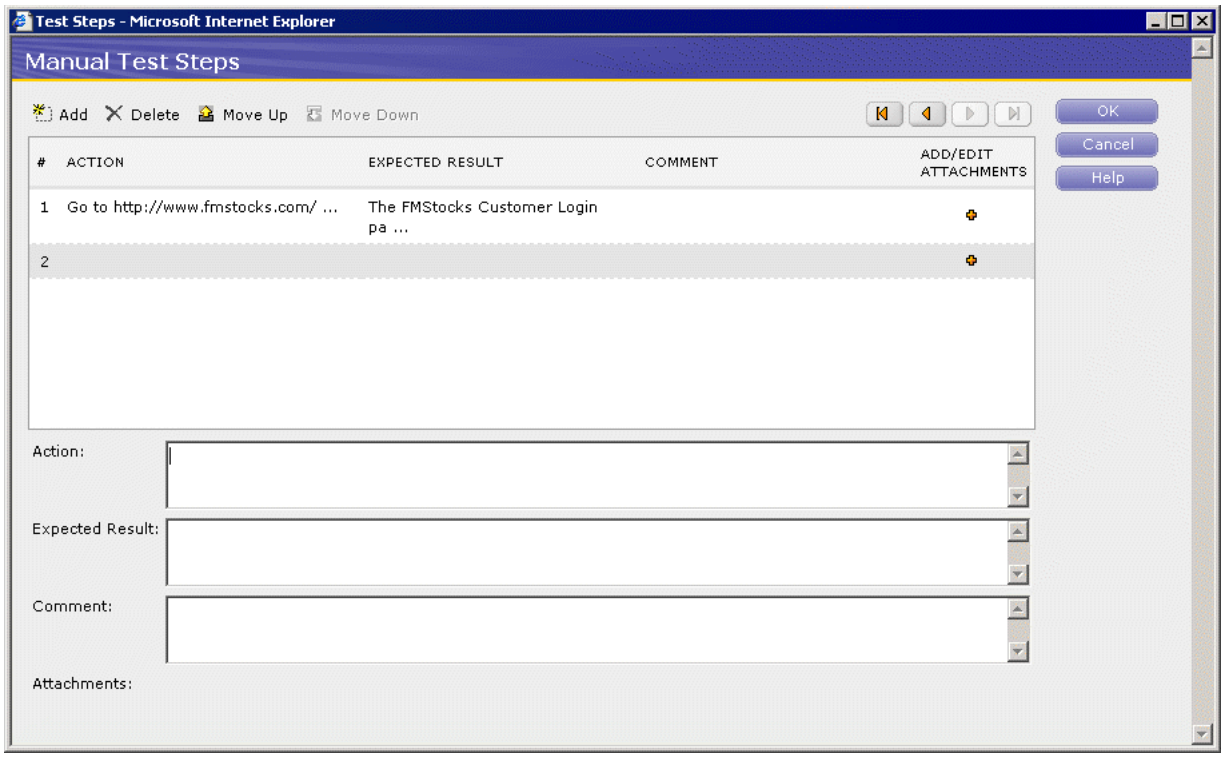
8. Click **Save**. The test is added as item number 2. You are now ready to add test steps.
9. Click **Add/Edit** in the Test Steps section of the right-hand pane to display the Test Steps dialog box.

Figure 5–9 Manual Test Steps Window



10. Click **Add**.
11. Enter, "Go to http://www.fmstocks.com/fmstocks" in the **Action** field.
12. Enter, "The FMStocks Customer Login page should be displayed" in the **Expected Results** field.
13. Click **Add** to update the top of the dialog box and to go to the next step.

Figure 5–10 Manual Test Steps Window with Sample Data



14. Enter the following steps, clicking **Add** to start each step:

Step	Action	Expected Result
2	Click Login.	The Welcome to FMStocks page should be displayed.
3	Click the Chart Your Portfolio link	The Your Portfolio page should be displayed with a spreadsheet and graphs.

The Manual Test Steps dialog box shows the entries as entered.

15. Click **OK**.

5.4.2 Adding an Automated Test

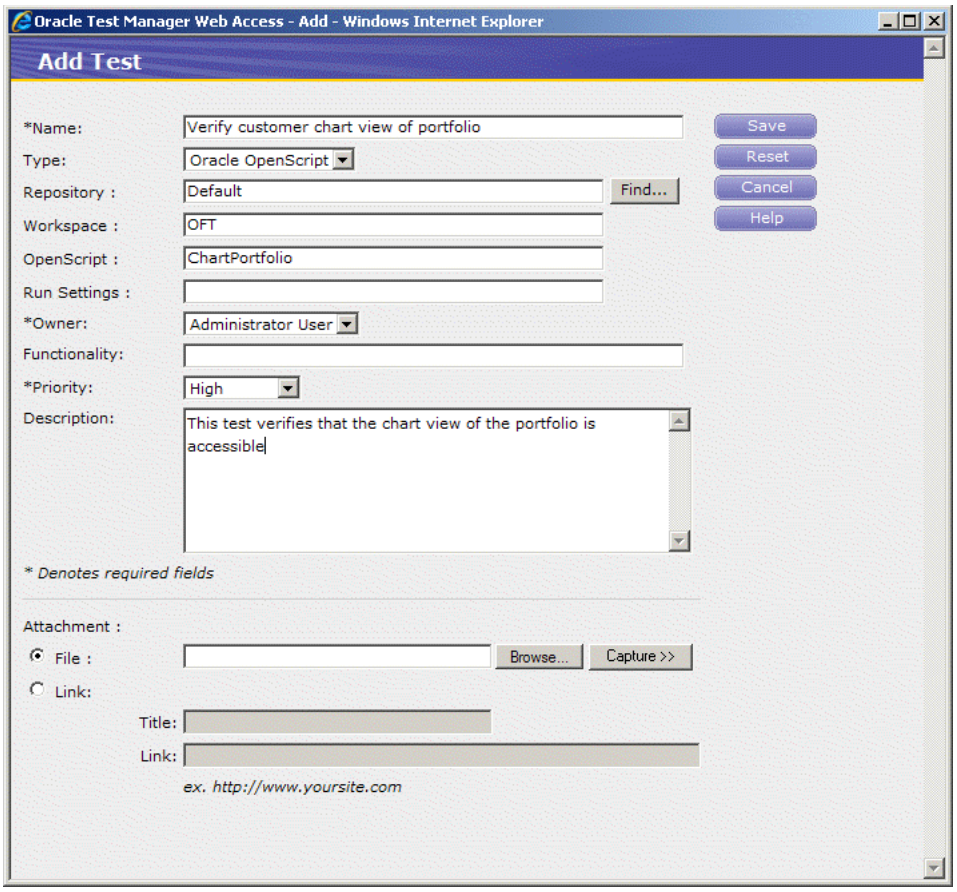
Oracle Test Manager can run Oracle OpenScript functional test scripts as automated tests. This example explains how to add a script created in Oracle OpenScript to the Tests in Oracle Test Manager and associate it with the requirement just added.

1. Select **Programs** from the **Start** menu and then select **OpenScript** from the **Oracle Application Testing Suite** menu.
2. Select **New** from the **File** menu.
3. Expand the Functional Testing (Browser/GUI Automation) folder.
4. Select "Web" and click **Next**.
5. Select the default repository.
6. Enter `ChartPortfolio` as the name of the script and click **Finish**.

OpenScript creates the script project and adds the Initialize, Run, and Finish sections to the script tree.

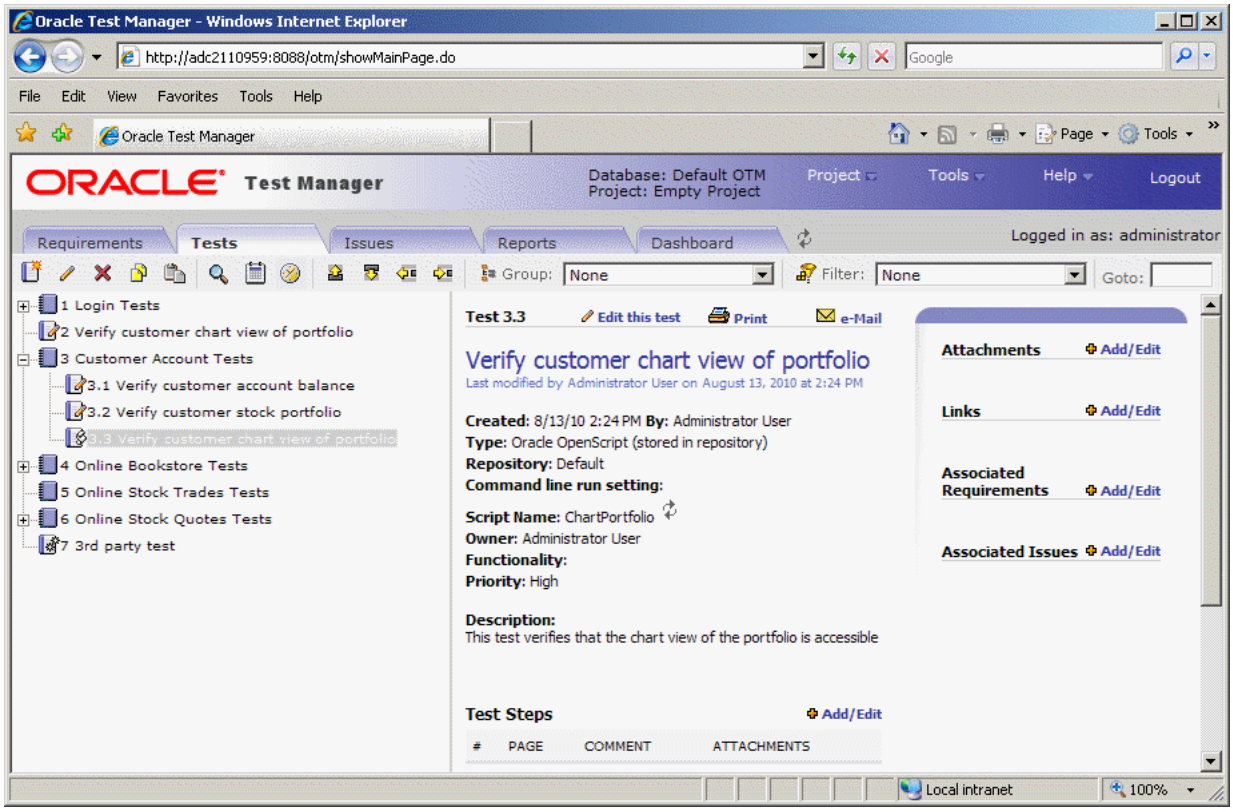
7. Select **Record** from the **Script** menu or click the toolbar button. A new browser window opens.
8. Go to `http://www.fmstocks.com/fmstocks`.
9. Click **Login**.
10. Click the **Chart Your Portfolio** link.
11. Select **Stop** from the **Script** menu or click the toolbar button.
12. Select **Save** from the **File** menu.
13. Select **Playback** from the **Script** menu or click the toolbar button and verify that the script plays back correctly and passes. You can view the results in the OpenScript **Results** view or the **Details** view.
14. Select **Exit** from the **File** menu to exit Oracle OpenScript and return to Oracle Test Manager.
15. Click the **Tests** tab.
16. Expand Customer Account Tests and click item 3.2 Verify customer stock portfolio.
17. Click **Add** to display the Add Test dialog box.
18. Enter "Verify customer chart view of portfolio" in the Name field.
19. Select **Oracle OpenScript** in the Type field.
20. Click **Find** in the Repository field.
21. Expand the Default repository, select the ChartPortfolio script, and click **OK**.
22. Select an owner and priority.
23. Enter "This test verifies that the chart view of the portfolio is accessible" in the Description field.

Figure 5–11 Add Test Window with Sample Automated Test



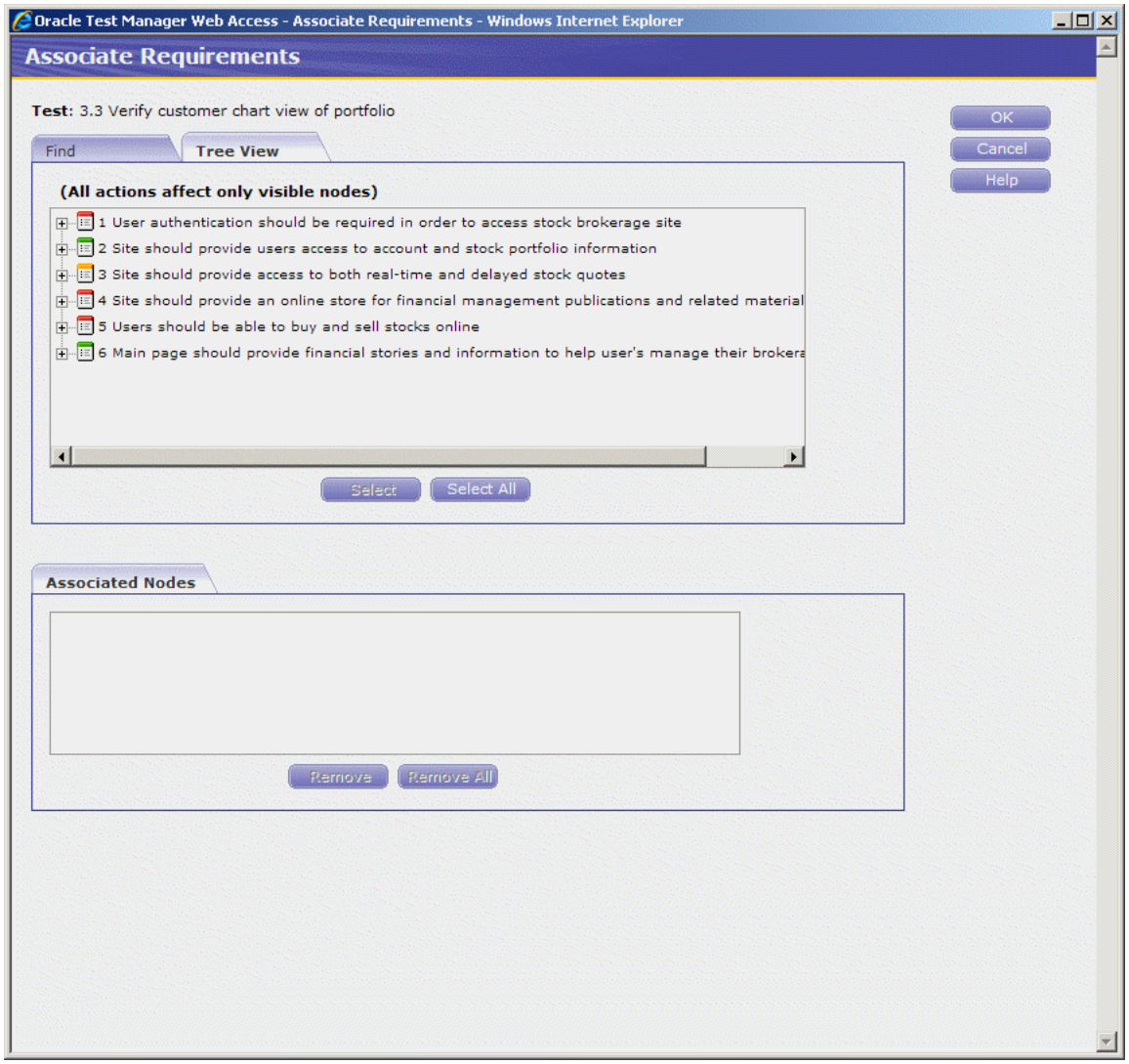
24. Click **Save**.
25. The test is added as item 3.3.

Figure 5–12 Tests Tab with New Test Added



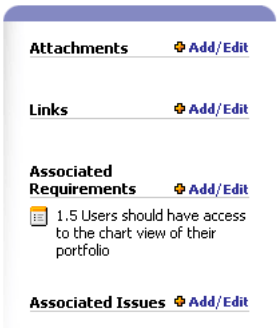
26. Click **Add/Edit** next to Associated Requirements in the right pane, to display the Associate Requirement dialog box.
27. Click the **Tree View** tab.

Figure 5–13 Associate Requirements Window



28. Expand item 1 and click item 1.5.
29. Click **Select**.
30. Click **OK**. The test is now linked to the requirement and is listed on the right-hand side of the screen. The test is also listed in the Associated Tests section of the requirement.

Figure 5–14 Test Associated with Requirement



5.5 Example 3: Running a Test

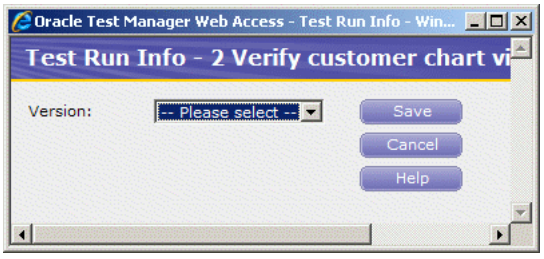
This example has two parts. The first part explains how to run the manual test created in the previous example. The second part explains how to run the automated test created in the previous example, then how to view the detailed Results Report.

5.5.1 Running a Manual Test

This example explains how to run the manual test entered in the previous example.

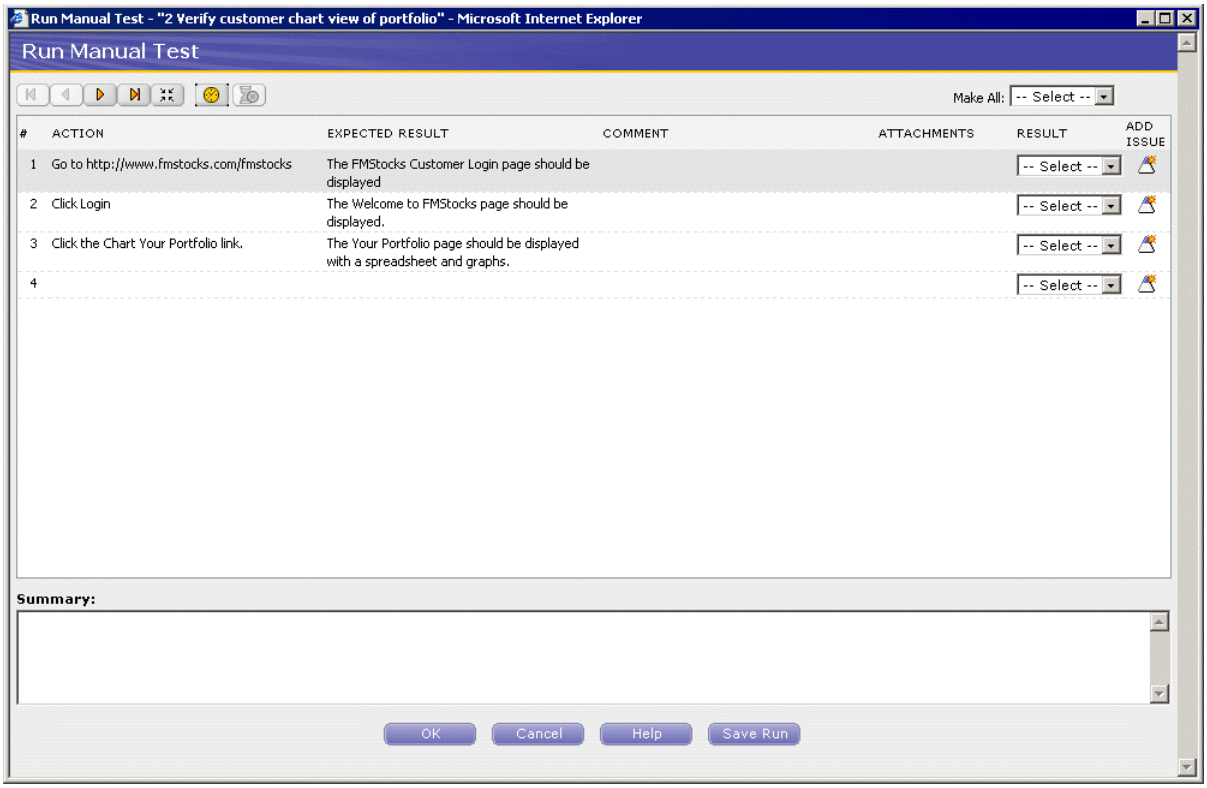
1. Click the Tests tab and select number 2, Verify customer chart view of portfolio.
2. Click **Run this test** in the Run History section of the right-hand pane.

Figure 5–15 Run Test Info Window



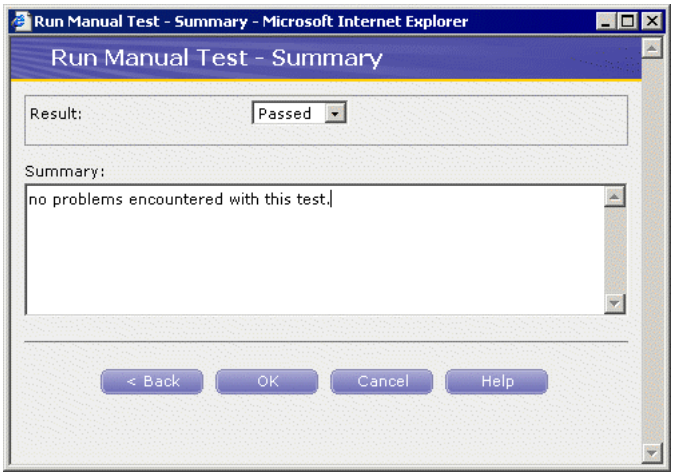
3. Select 1.0 in the **Version** field. This is the version number of the application you are testing.
4. Click **Save**.

Figure 5–16 Run Manual Test Window



5. This dialog box tells you the action to take and the expected result. Open your browser and perform the first action. Select the result **Passed**. Enter comments, when needed, in the Summary field.
6. Repeat for steps 2 and 3.
7. Click **OK** to display the Run Manual Test - Summary window.

Figure 5–17 Run Manual Test Summary Window

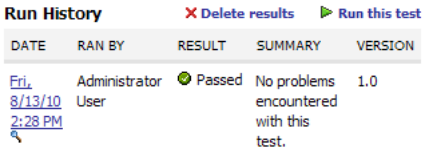


8. Select **Passed**.
9. Enter, "No problems encountered with this test." in the Summary field.

10. Click **OK**.

The Run History and Result Detail are displayed in the right pane.

Figure 5–18 Run History of Manual Test



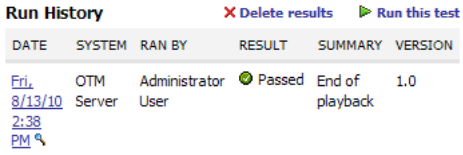
5.5.2 Running an Automated Test

This example explains how to run an the automated test created in the previous example and how to view the results detail and the detailed Oracle OpenScript Results Report.

1. Click the **Tests** tab.
2. Expand item number 3 and click item 3.3 Verify customer chart view of portfolio.
3. Click **Run this Test** in the Run History section of the right-hand pane to display the Test run Info dialog box.
4. Select **OTM Server** in the **System** field. This is the machine on which the test will be run.
5. Select 1.0 in the **Version** field. This is the version number of the application you are testing.

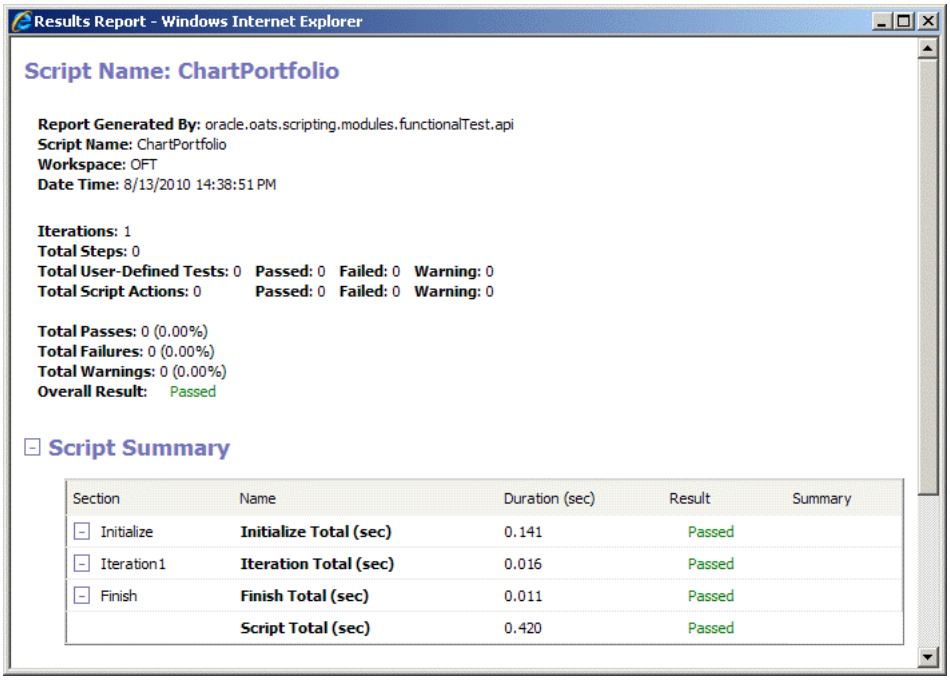
   The **Run Settings** field is an advanced option used to pass additional settings to the OpenScript script at run time. We will leave this field blank.
6. Click **Save**. Oracle OpenScript executes the script (this may take a few moments).
7. Click the Refresh arrows next to the Script Name in the right pane to update the Run History. Oracle Test Manager displays the Result Detail Summary when the test is finished.

Figure 5–19 Run History of Automated Test



8. Click the date of the test in the Run History section of the right pane.
9. Click **View Report** in the Result Summary to display the detailed Oracle OpenScript Results Report in a separate browser window.

Figure 5–20 Results Report Window



10. Review the report and close the Result Report window when finished.

5.6 Example 4: Adding an Issue

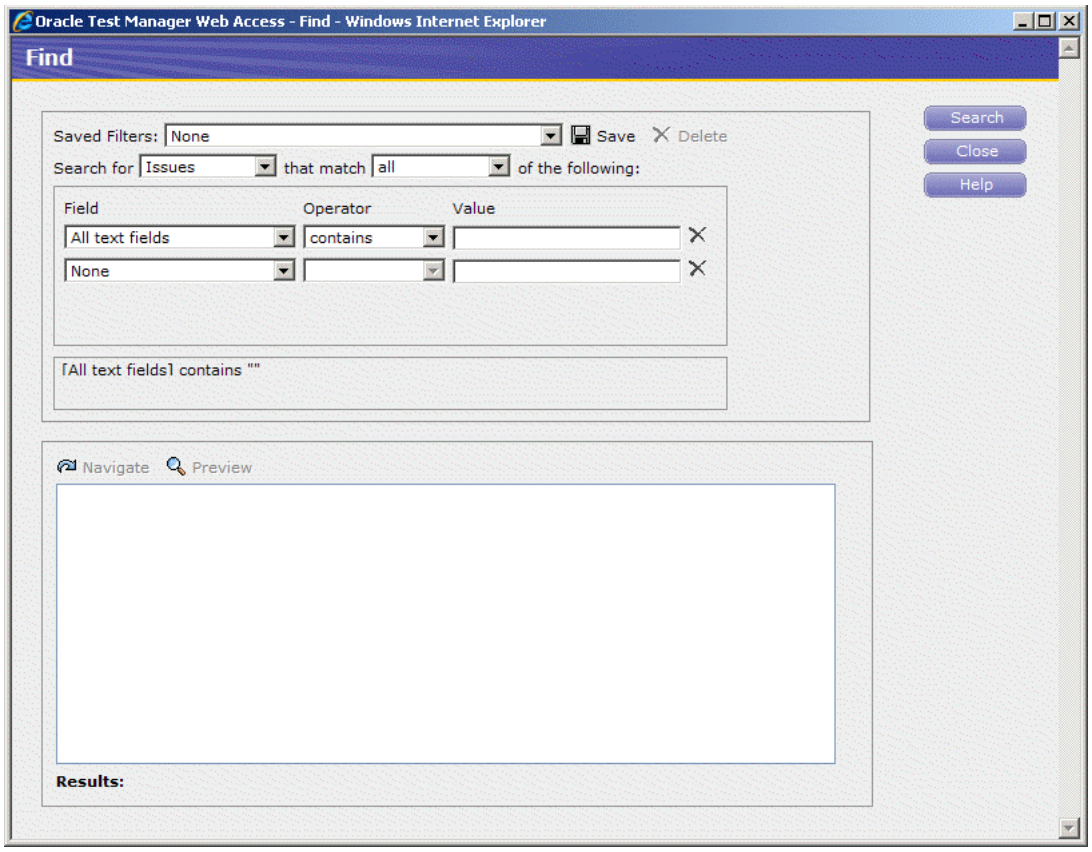
This example explains how to search through issues, add an issue, associate it with a test, and add an attachment with additional information. For the purposes of this example, we will assume that the script run in example 3 failed.

To search through issues to see if an issue already exists for this topic, or to find related issues:

1. Click the **Issues** tab.

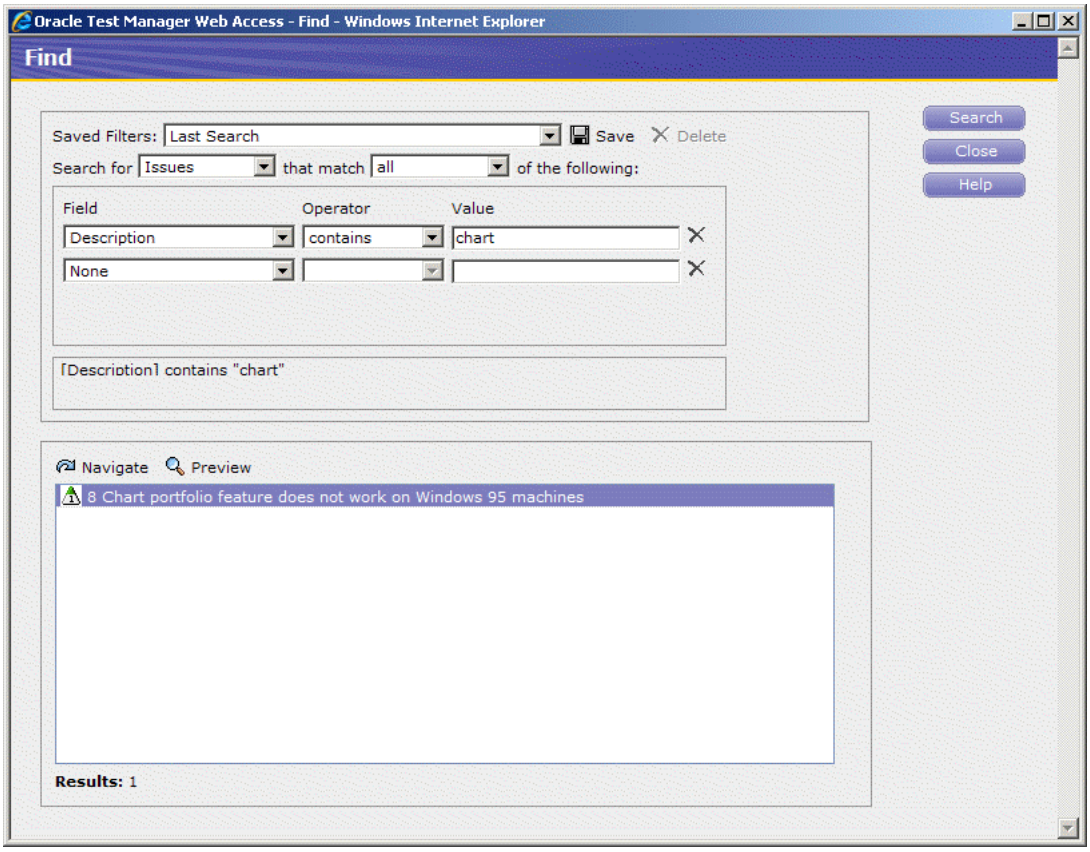
2. Click the **Find** icon (magnifying glass) to display the Find dialog box.

Figure 5–21 Find Window



3. Select **Description** for the field to search.
4. Enter "chart" in the **Value** field.
5. Make sure that **Issues** is selected in the **Search for** field and that **all** is selected in the **that match** field. This will search all of the issue descriptions for the word, "chart."
6. Click **Search**. If there are any matches, they are displayed in the Results portion of the window.

Figure 5–22 Find Window with Search Results



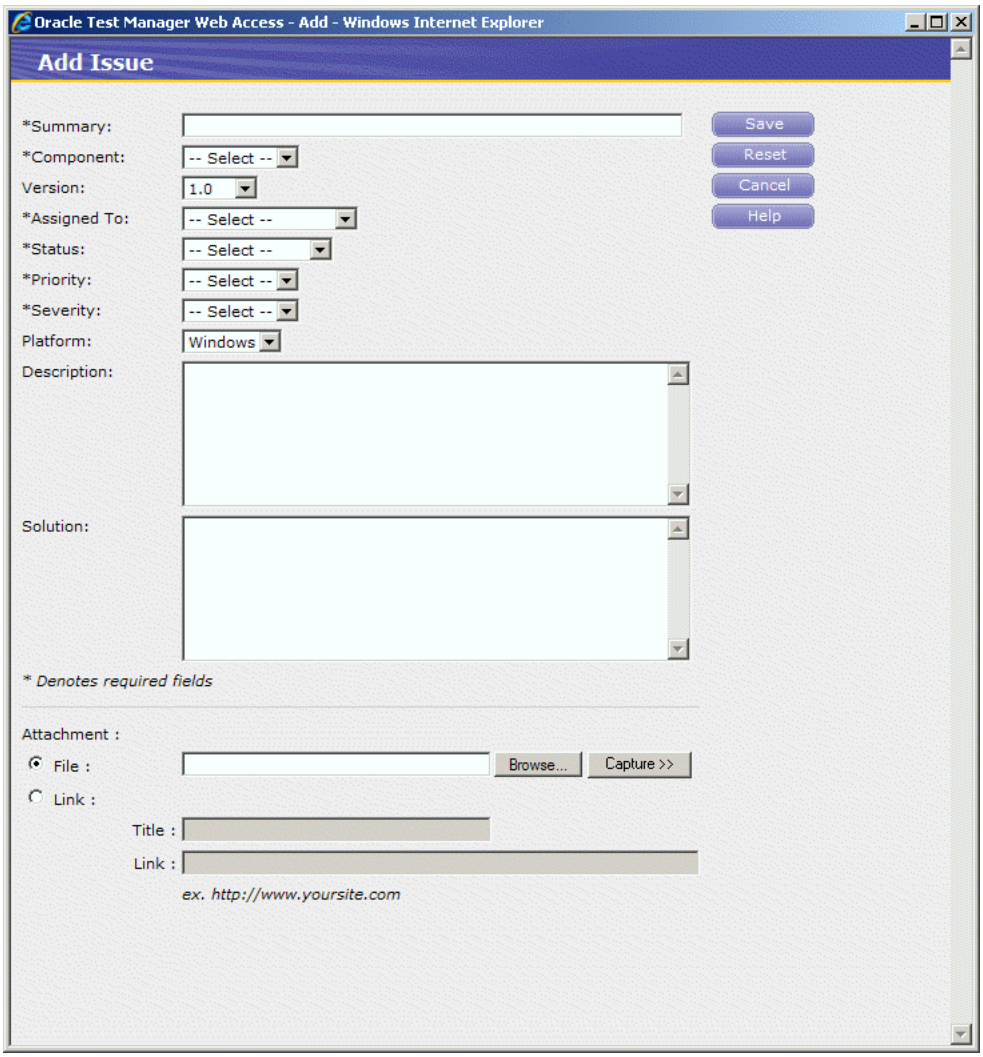
One match was found. You can click **Navigate** in the Results list to display it in Oracle Test Manager or click **Preview** to view the issue in a separate preview window. We will assume that our failure is different enough to warrant creating a new issue.

7. Click **Close** to close the Find dialog box.

To create an issue.

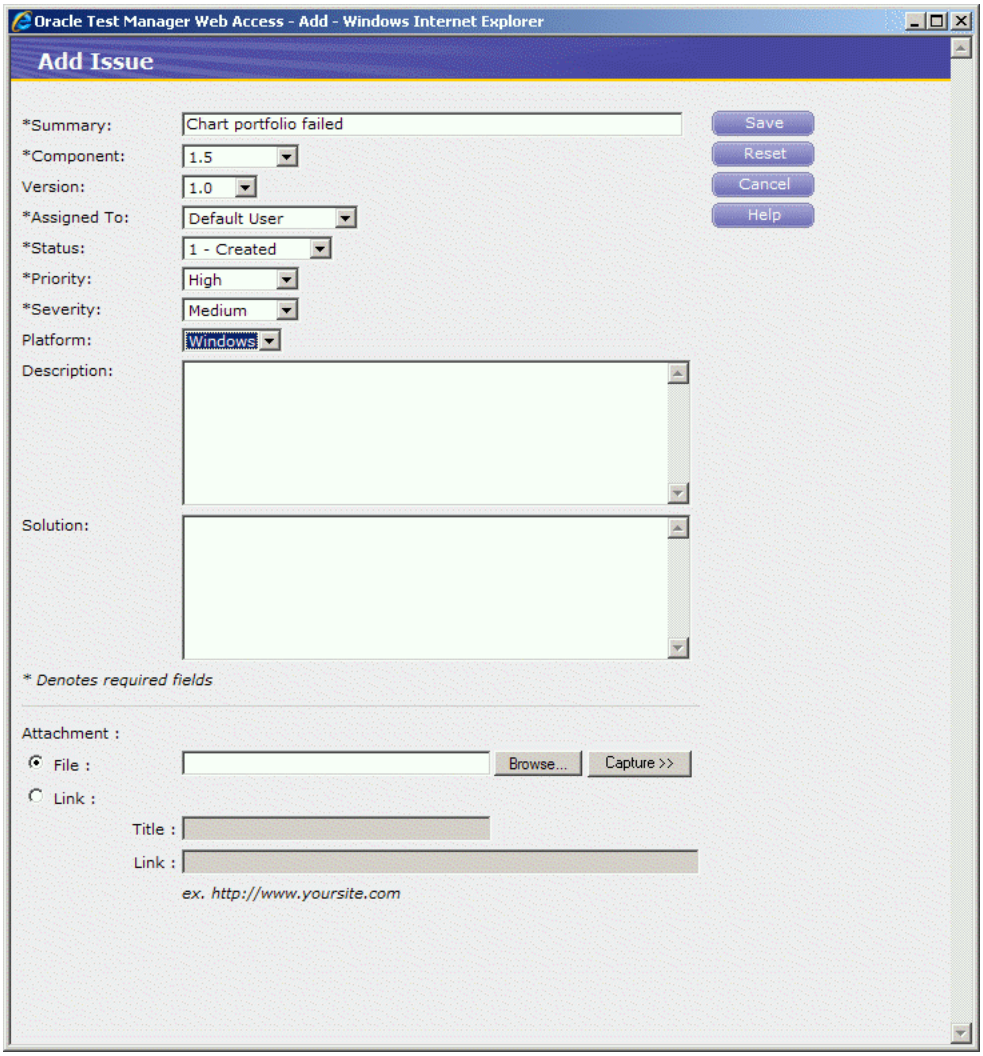
1. Click the **Issues** tab.
2. Click **Add** to display the Add Issue dialog box.

Figure 5–23 Add Issue Window



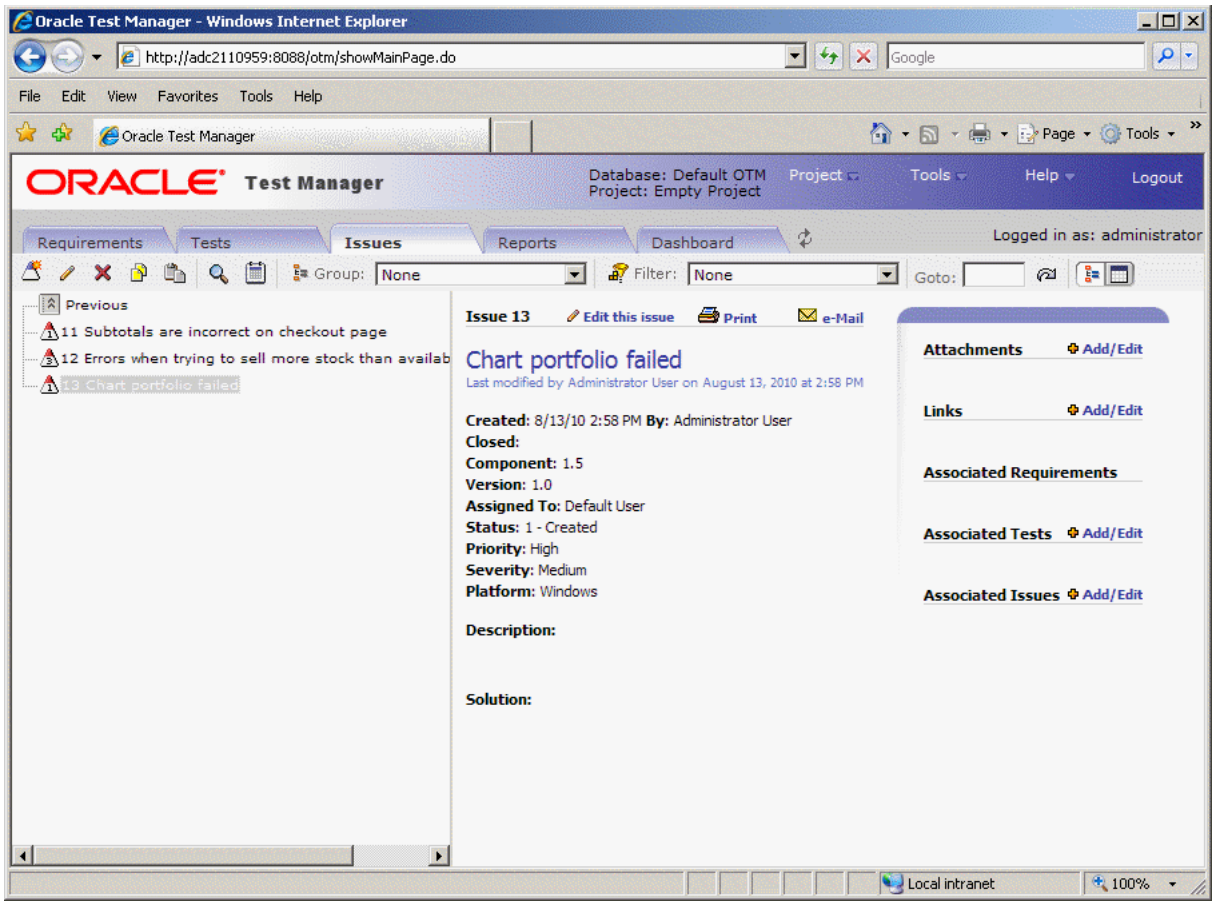
3. Enter "Chart portfolio failed" in the **Summary** field.
4. Select the default component.
5. Select 1.0 for the version.
6. Assign the issue to Default User.
7. Set the status to Created.
8. Change the Priority to High.
9. Change the Severity to Medium.
10. Select the Windows platform.

Figure 5–24 Add Issue Window with Sample Data



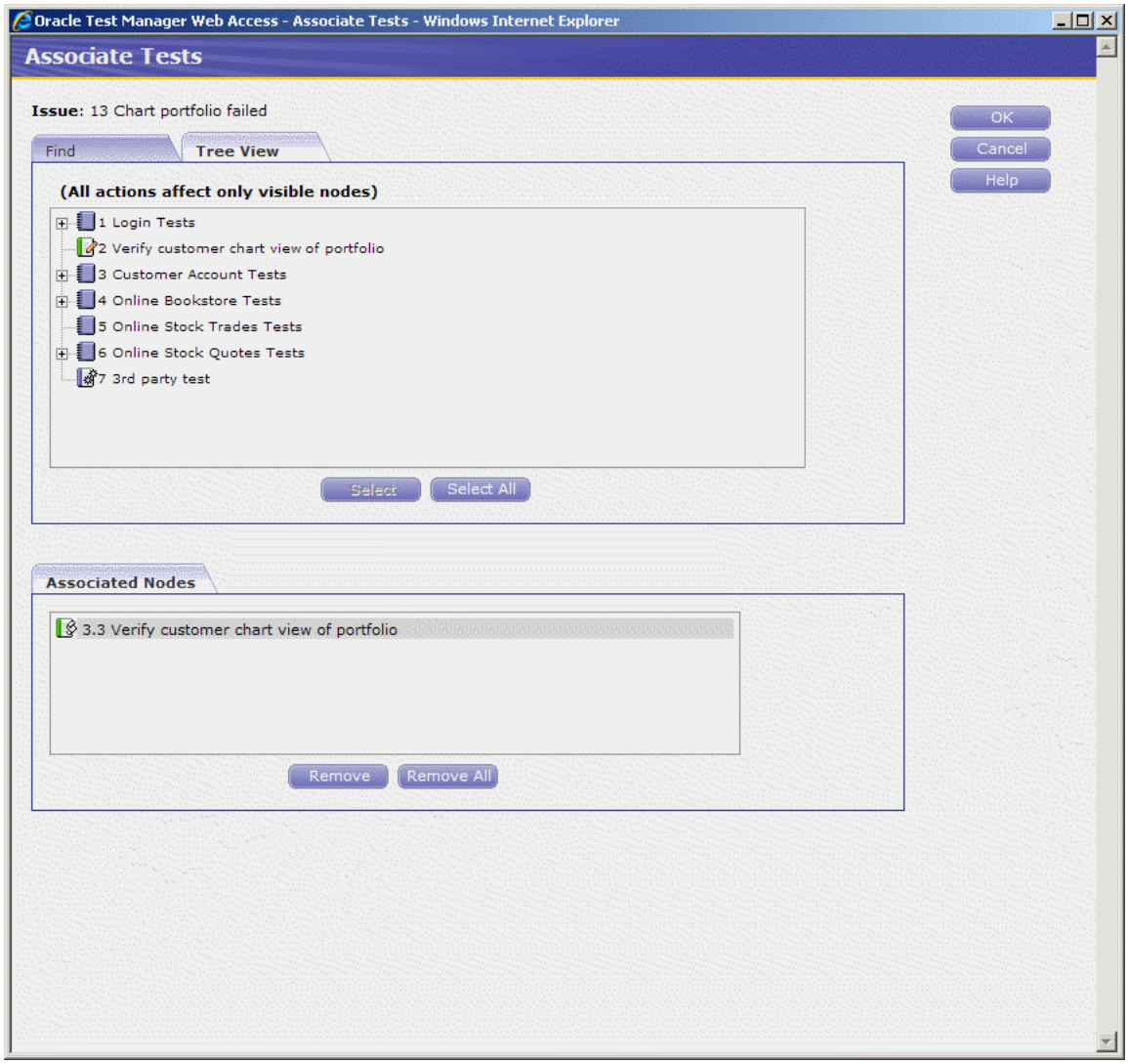
11. Click **Save**. The issue is assigned the next available number and added to the bottom of the list.

Figure 5–25 Issues Tab with New Issue Added



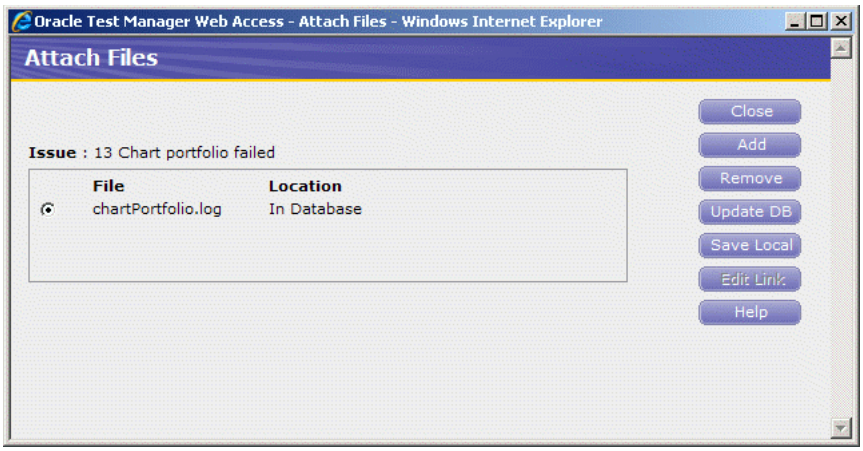
12. Click **Add/Edit** next to Associated Tests in the right pane to display the Associate Test dialog box.
13. Click the **Tree View** tab.
14. Expand item 3 and select item 3.3.
15. Click **Select**.

Figure 5–26 Associate Test Window with Issue Selected



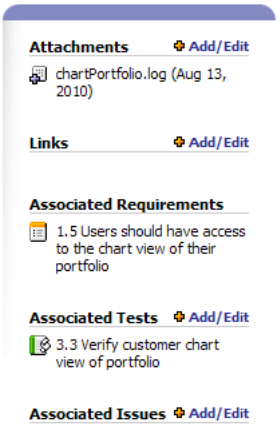
16. Click **OK**. The test is listed in the Associated Tests list.
17. Select **Add/Edit** in the Attachments section to display the Attach Files dialog box.
18. Click **Add**.
19. Click **Browse**.
20. Select a file to attach and click **Open**.
21. Click **Upload**.

Figure 5–27 Attach Files Window with File Selected



22. Click **Close**. The attachment is listed in the Attachments list in the right pane. Click on the attachment to open it in the appropriate application.

Figure 5–28 Issue with File Attachments

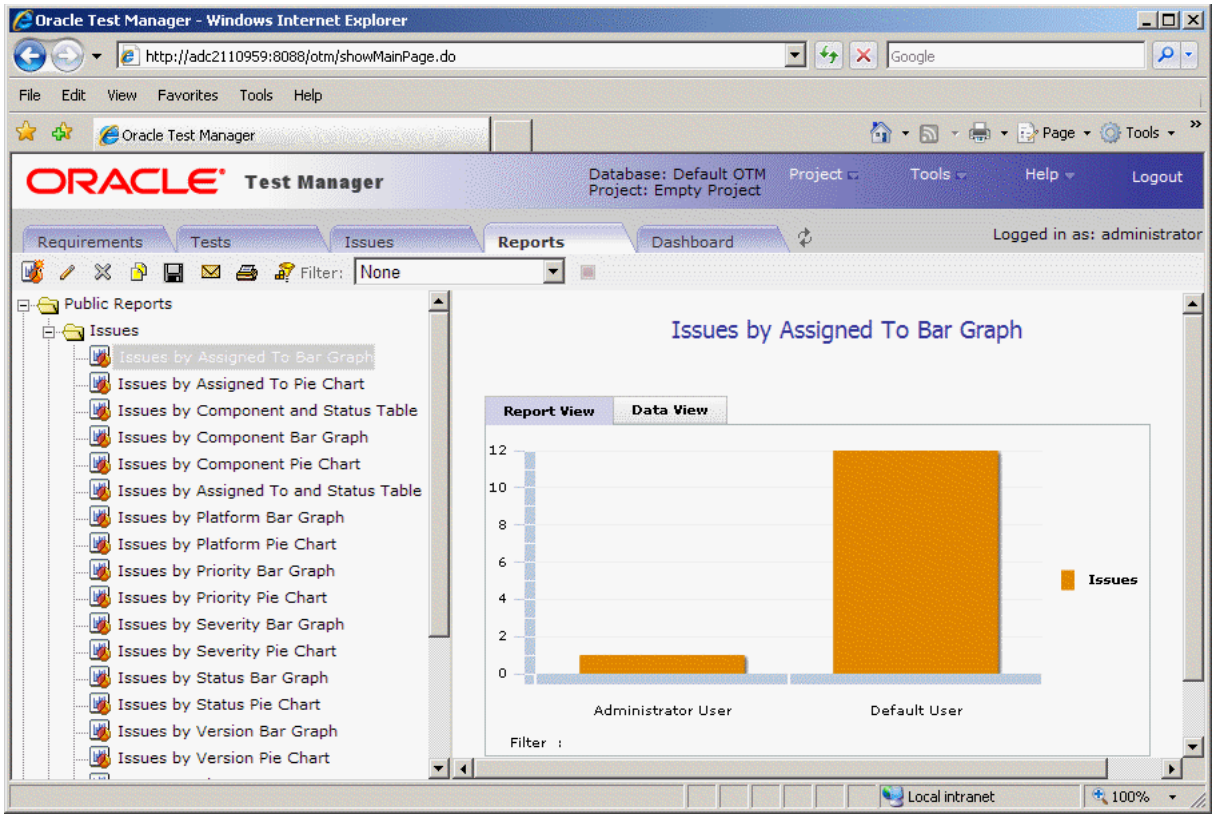


5.7 Example 5: Creating Reports

Oracle Test Manager comes with a standard set of reports. In addition, you can create and optionally save custom reports. Reports can be either public, that is, available to all users, or private, available only to you. This example explains how to view standard reports and how to create a custom report and save it.

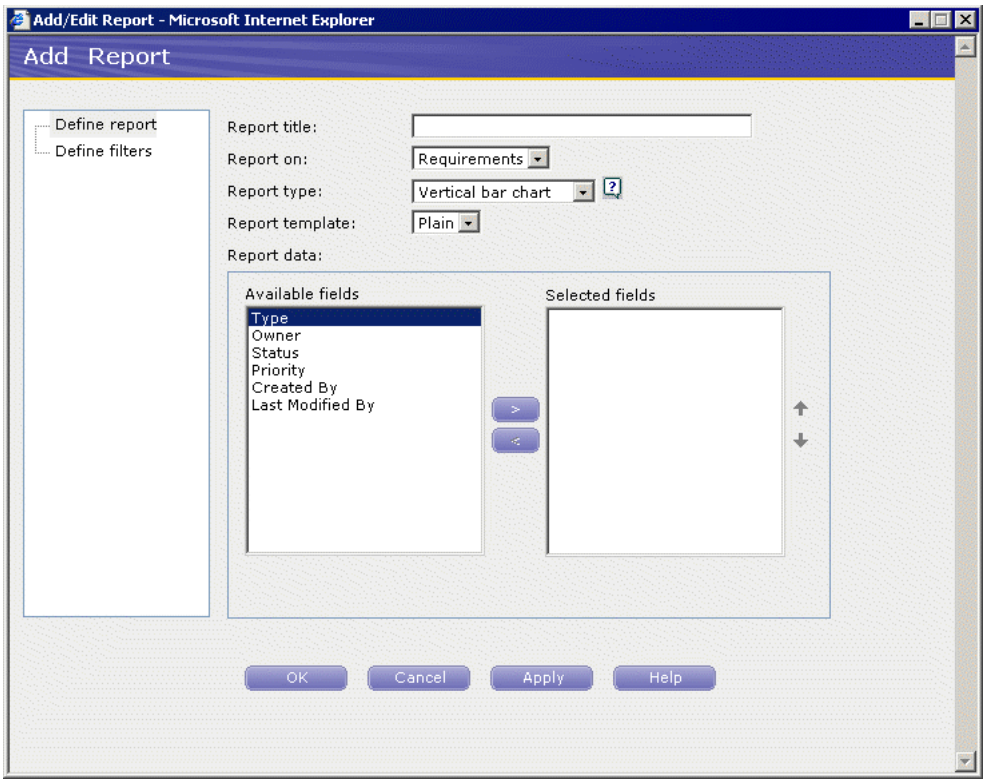
1. Click the **Reports** tab.
2. Expand the Public Reports node, then the Issues node.
3. Select **Issues by Assigned to Bar Graph** to display the graph in the right pane.

Figure 5–29 Reports Tab with Bar Graph Report



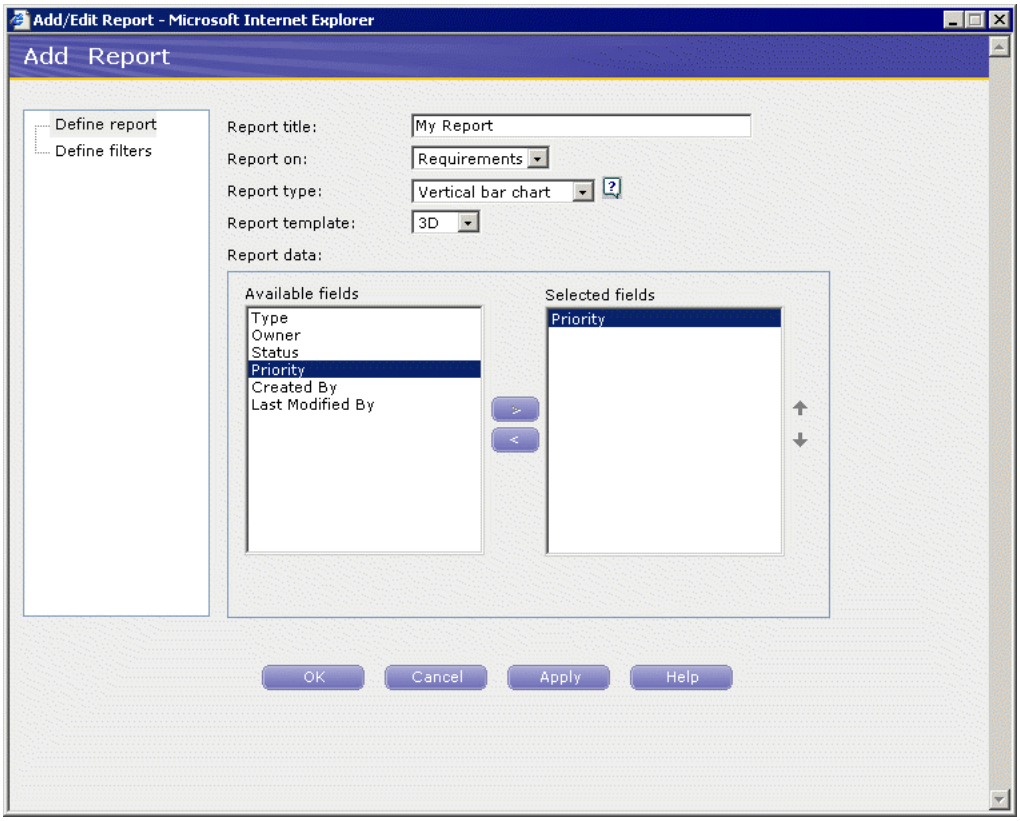
4. Click once on the graph area then mouse over the bars to view the actual values. Click **Data View** to view just the data.
5. To create a custom report click **Add**.

Figure 5–30 Add Report Window



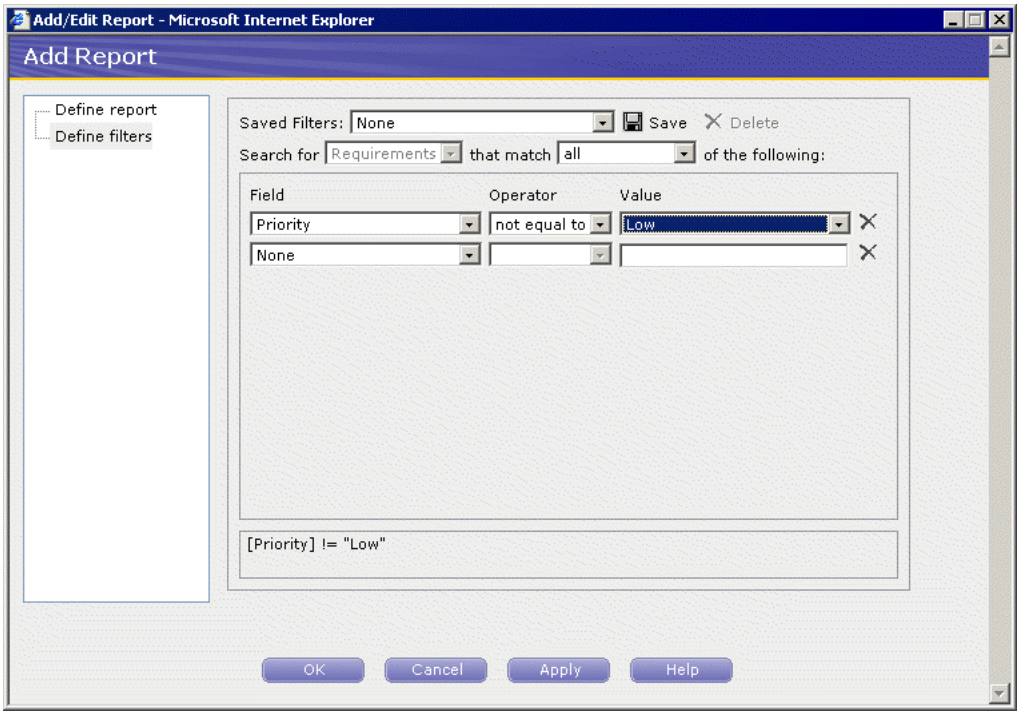
6. Enter a name for the report.
7. Select **Vertical bar chart** for the report type and **3D** for the report template.
8. Select **Priority** from the **Available fields** and click the right arrow to add it to the **Selected fields** list.

Figure 5–31 Add Report Window with Selected Fields



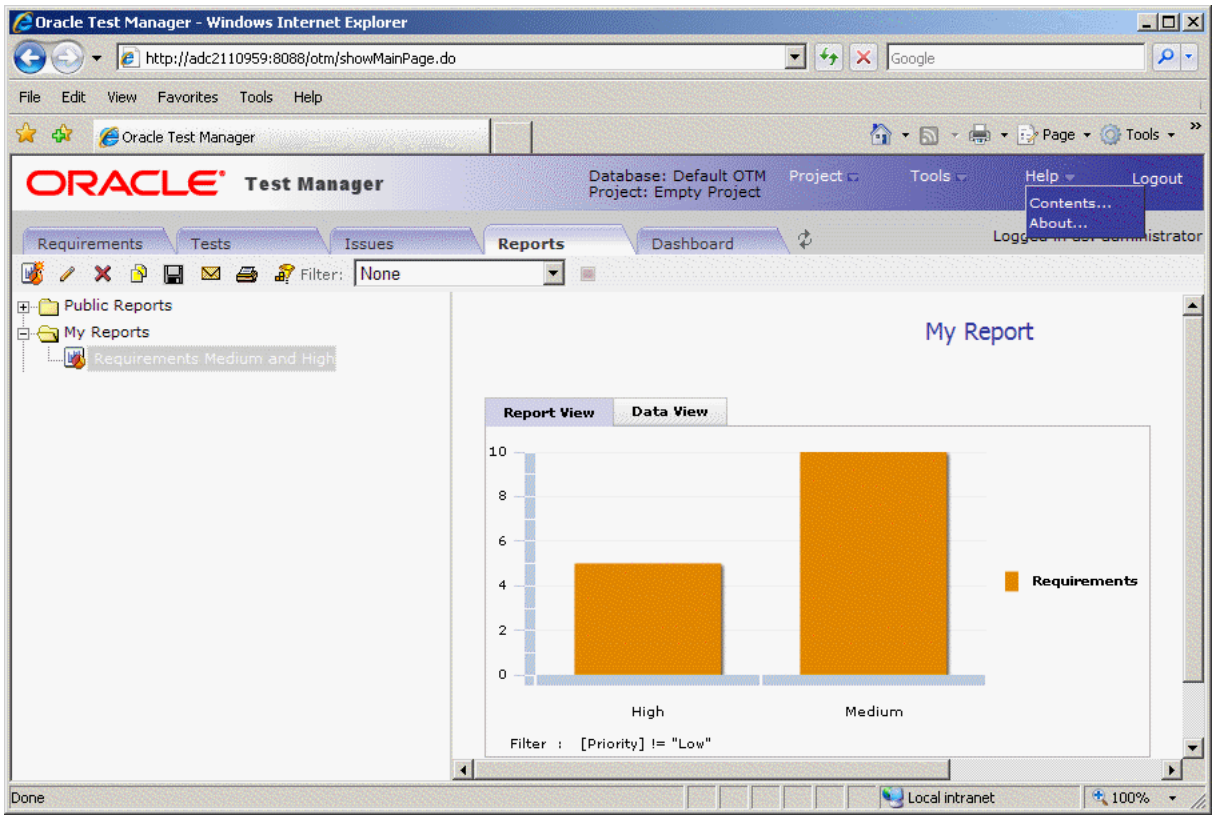
9. Click **Define filters**.

Figure 5–32 Add Report Filters Window



10. Select **Priority** as the field, **not equal to** in the **Operator** field, and **Low** for the Value. This means that the report will only display information for Medium and High priority requirements.
11. Click **OK** to display the report.
12. Click **Save** to save this report.
13. Enter **Requirements Medium and High** for the report name. The name will be displayed in the report tree.
14. Select **My Reports** as the Report Category.
15. Click **OK**. The new report is added under the My Reports folder.

Figure 5–33 Reports Tab with Custom Reports

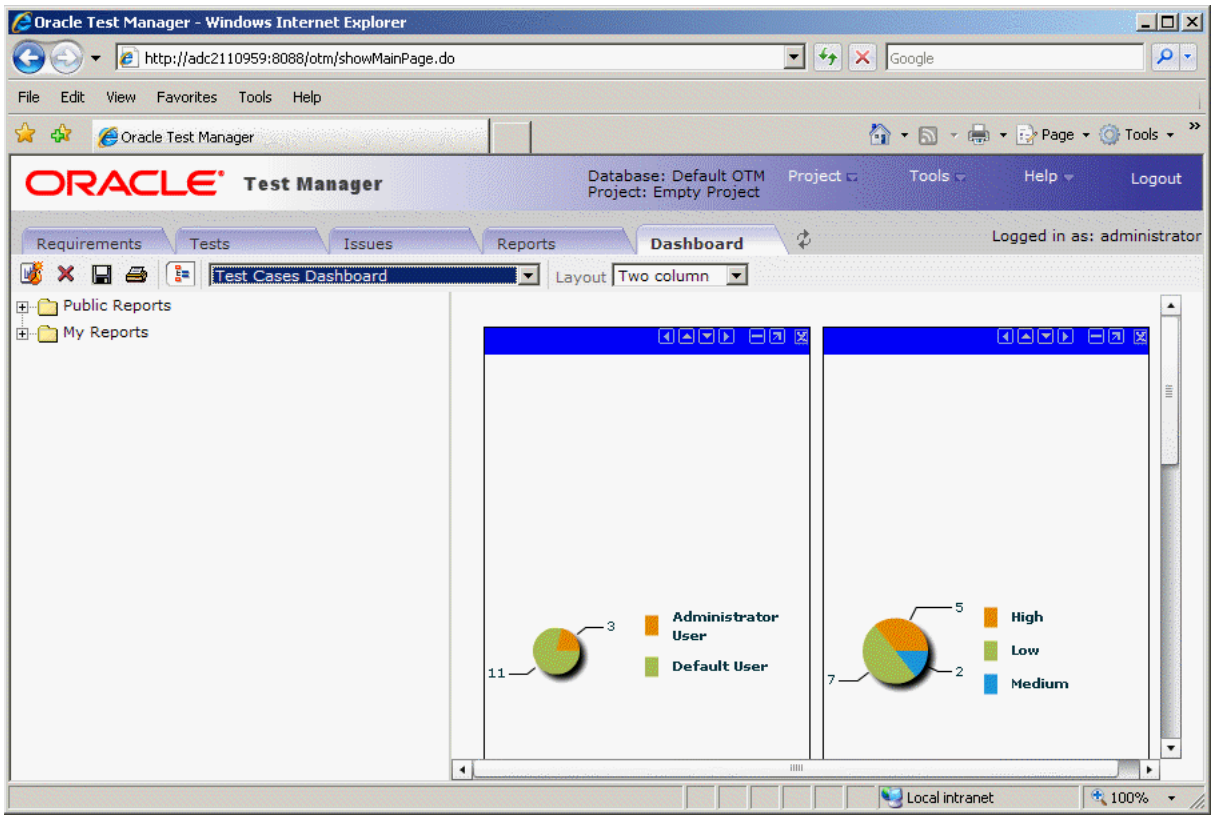


5.8 Using Dashboards

The Oracle Test Manager Web Dashboard tab is where you can create and view multi-report dashboards. These dashboards allow you to select the key reports of interest and save them in a single integrated view. This allows you to create report dashboards that are tailored to the needs of specific users. Each new dashboard can also be saved as either a public dashboard which will be viewable by all users who log into this project or a private dashboard that is only viewable by your account.

1. Click the **Dashboard** tab.
2. Select the Test Cases Dashboard from the select list.
3. Select Two Column from the Layout list.

Figure 5–34 Dashboard Tab with Test Cases Dashboard

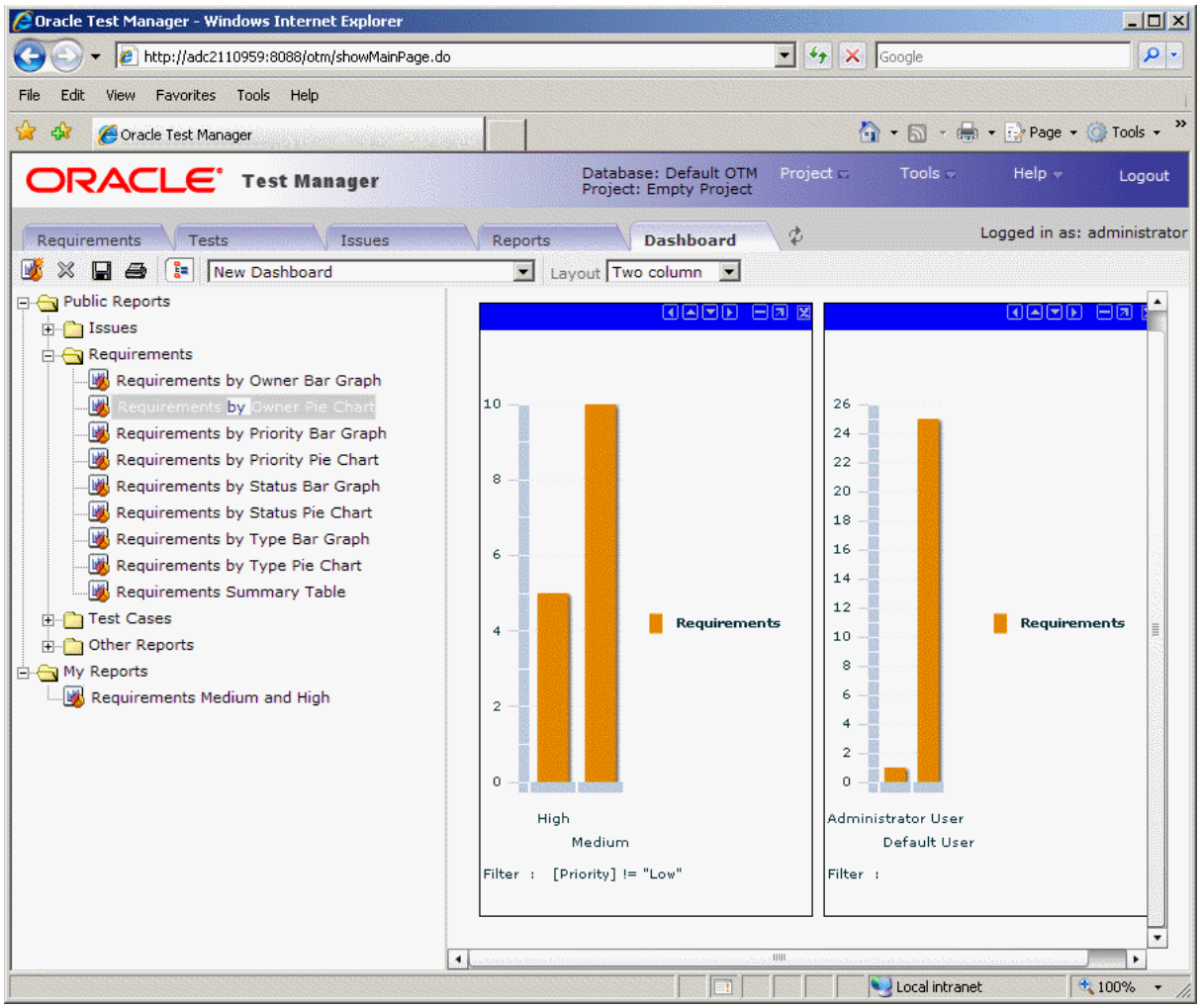


Each graph in the Dashboard has a toolbar for moving, sizing, and closing individual graphs in the Dashboard.

Figure 5–35 Dashboard Graphs Toolbar

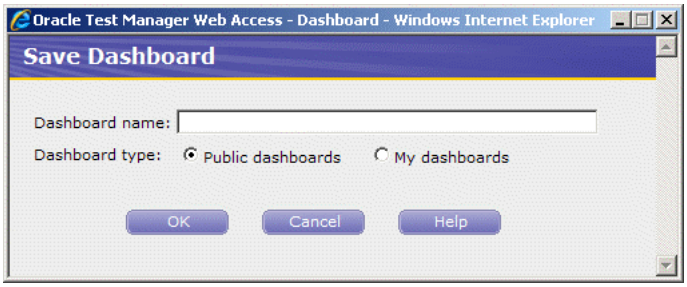
4. Click **New Dashboard** on the main toolbar.
5. Expand the My Reports node.
6. Double-click the **Requirements Medium and High** report created previously in this example.
7. Expand the Public Reports node then the Requirements node.
8. Double-click the **Requirements by Owner Bar Graph report**. The second graph is added to the Dashboard.

Figure 5–36 New Dashboard with Selected Graphs Added



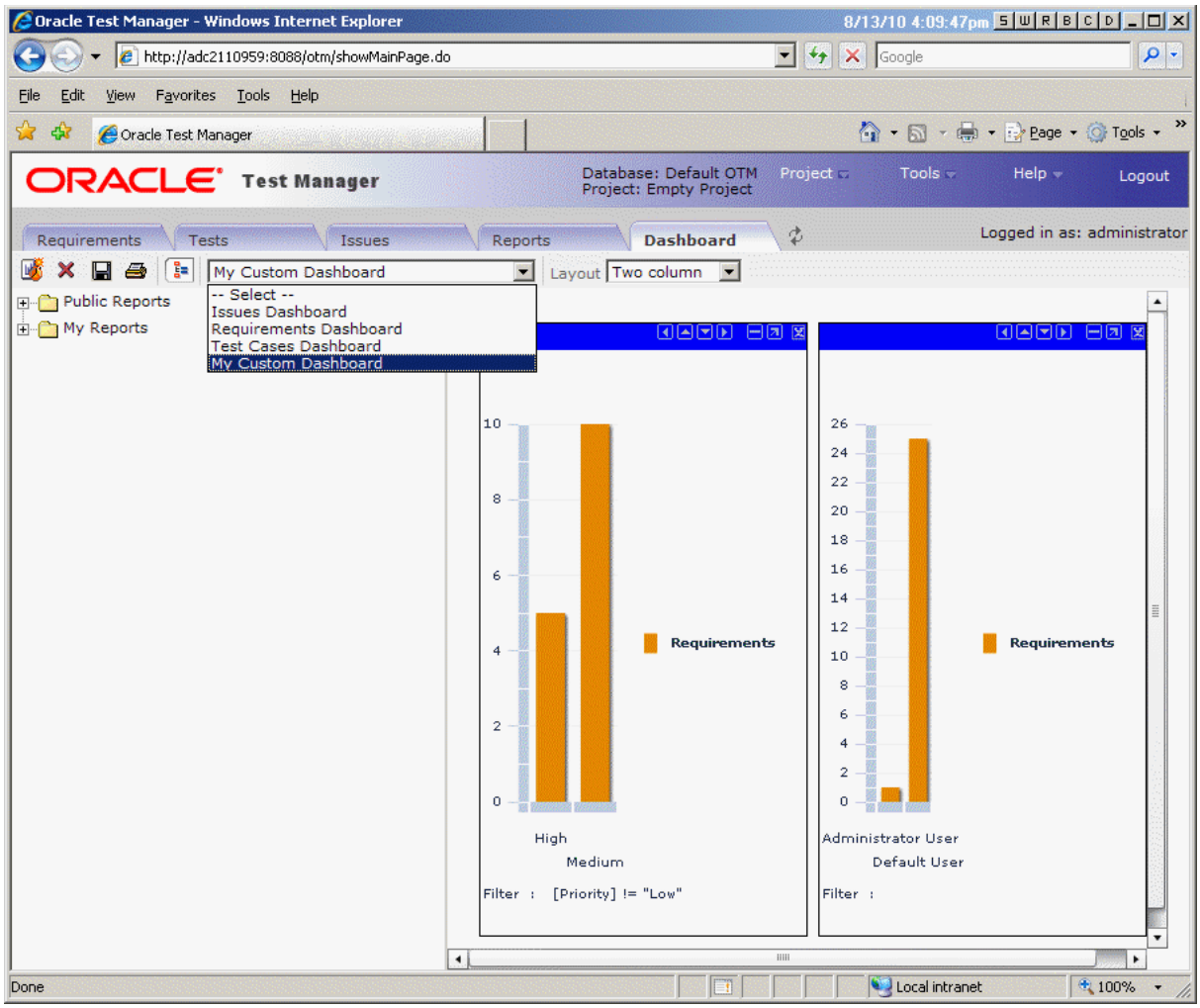
9. Click the left arrow [<] in the second graph's toolbar to move it to the left.
10. Click the minimize [-] button in a graph's toolbar to minimize the graph. Click the restore [+] button in a graph's toolbar to restore the graph.
11. Click **Save**.

Figure 5–37 Save Dashboard Dialog Box



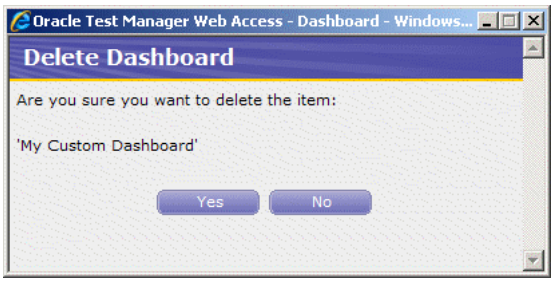
12. Enter **My Custom Dashboard** as the name for the custom Dashboard.
13. Select **My Dashboards** and click **OK**. The new dashboard is added to the Dashboard sector list.

Figure 5–38 Dashboard Tab with a Custom Dashboard Added



14. Make sure My Custom Dashboard is selected and click **Delete**.

Figure 5–39 Delete Dashboard Dialog Box



15. Click **Yes** to delete the custom Dashboard.

This completes the Oracle Test Manager tutorial. Click Logout to exit the application.

Index

S

T

U

V

W