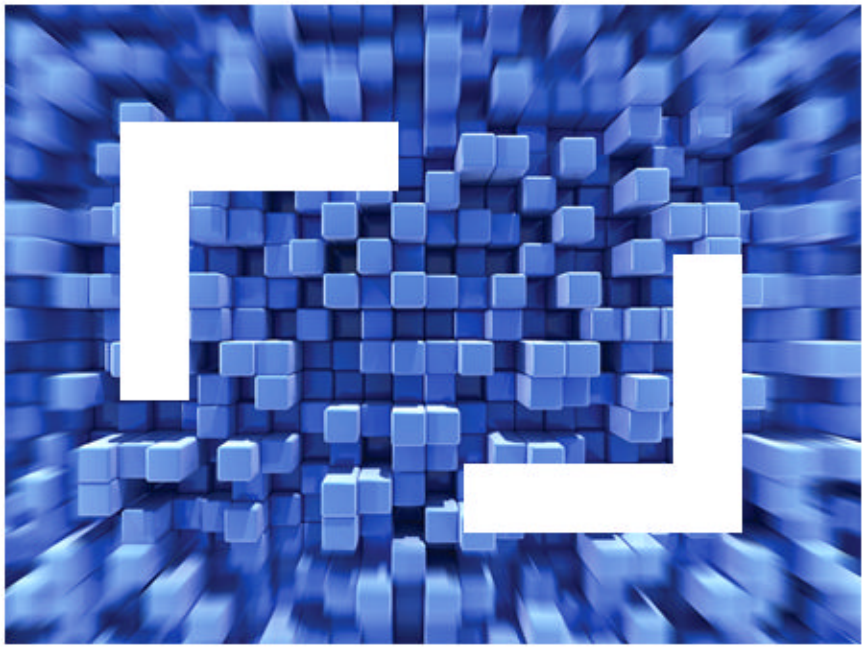


SilkCentral Test Manager 2010 R2

SilkCentral Test Manager / SilkPerformer Integration Tutorial

Borland®
(A MICRO FOCUS COMPANY)

MICRO® FOCUS
Leading the Evolution™

Borland Software Corporation
4 Hutton Centre Dr., Suite 900
Santa Ana, CA 92707

Contents

Introduction

This tutorial guides you through a typical SilkPerformer / SilkCentral Test Manager (Test Manager) integration scenario and provides you with the information you need to make the most of the integration.

Overview

SilkPerformer (version 7.1 or later) is fully integrated with the test-planning and test-execution functionality of Test Manager. SilkPerformer projects can be integrated into Test Manager test plans and executed directly from Test Manager. This allows for powerful test-result analysis and reporting. It also enables unattended SilkPerformer tests, which are tests that are run automatically by Test Manager based on pre-configured schedules.

From Test Manager, SilkPerformer projects can be opened directly in SilkPerformer for the editing of scripts and settings. Edited SilkPerformer projects can subsequently be uploaded back to Test Manager to make them available for future test executions.

Integration with Test Manager also enables access to your test plans directly from SilkPerformer Workbench, allowing you to easily open, edit, and upload your Test Manager test projects from SilkPerformer. Access to Test Manager is completely Internet-enabled, allowing access from a Web browser or SilkPerformer Workbench.

The results of the most recent test runs in SilkPerformer, excluding recent Try Script runs, can also be uploaded to Test Manager and associated with test definitions. To do this, SilkPerformer searches its directories for the most recent directory that includes a baseline report file. The files in the directory are then uploaded to Test Manager.

Load Test Results

SilkPerformer stores all data that are generated during load test runs (including time series data, TrueLog files, result files, and summary data) in the central Test Manager repository, which is easily accessible via the Test Manager Web GUI.

Cross Load-Test Analysis

When used in combination with Test Manager, Silk Performance Explorer offers a cross load-test comparison-report facility. Simply browse your Test Manager test plans with Performance Explorer and select up to four load-test runs for comparison. Heat fields and rankings help you analyze the results of your optimization efforts across test runs.

Source Control Integration

SilkPerformer integrates with the source control profiles of Test Manager. For additional information regarding Test Manager source control profiles, refer to *Test Manager Help*. Source-code control functionality, such as checking files in and out and getting the latest file versions, is available directly from SilkPerformer and is executed automatically when you open test projects from Test Manager or upload edited projects to Test Manager.

Enhanced Test Asset Management

SilkPerformer provides additional directories for custom user data and SilkPerformer include files (BDH) enabling you to better organize your test scripts and test data and to share them across multiple projects.

Dynamic Workload-Assignment to Agent Clusters

SilkPerformer workload delivered by Test Manager can make use of dynamic workload-to-agent assignment. The dynamic workload-assignment functionality in SilkPerformer matches your specific test requirements to the replay capabilities of available agent computers at execution time.

Working with Performance Explorer

Performance Explorer can be used for in-depth analysis of test runs. Performance Explorer results analysis can be launched directly from Test Manager through execution runs on the **Runs** page. Results analysis can also be launched from Performance Explorer. For additional information, refer to *Performance Explorer Help*.

SilkPerformer Integration with Test Manager

Note: This tutorial assumes that you have a working knowledge of both Test Manager and SilkPerformer.

This tutorial walks you through a typical integration scenario between SilkPerformer and Test Manager, demonstrating the following:

- Configuring SilkPerformer for use with Test Manager.
- Setting up Test Manager test definitions and execution definitions.
- Executing SilkPerformer tests from Test Manager.
- Analyzing test results in Performance Explorer.
- Generating cross load-test reports.

Configuring Access to Test Manager

To begin this tutorial, you must configure access to Test Manager from SilkPerformer.

1. Select **Settings** > **System** from the SilkPerformer menu bar.
   The **System Settings - Workbench** dialog box displays.
2. Click the **Test Manager** tab.
3. In the **Hostname** text box, type the host name or IP address of the host where the Test Manager front end server is running.
   Do not include the *http://* protocol in the host name.
4. In the **Port** text box, specify the number of the port on which Test Manager is listening.
   This port is typically **`80`** when using an ISAPI Web server. The port is typically **`19120`** for standalone Web server configurations.
5. Type valid user credentials in the **Username** and **Password** text boxes.
6. Click **OK**.
7. *Optional:* To configure proxy settings for your server environment, click **Internet Options** to open the **Internet Properties** dialog box. Click the **Connections** tab, then click **LAN settings**.

Configuring SilkPerformer Workbench

In addition to Test Manager access, other basic SilkPerformer testing configurations can be set through **Workbench** system settings. If you already have a fully configured SilkPerformer installation, you may not need to configure these settings to complete this tutorial.

Configuring Directory Settings

You can specify the directories where SilkPerformer files are stored. The default directories are set up during installation. You can specify alternative or additional directories on the **Directories** page. You can also define locations for the following:

- Testing scripts
- Include files
- Test result files
- User-data files
- Certificate files
- Random-data files

Note: If you maintain a number of SilkPerformer projects, you should define and use *custom include* and *custom data directories* to store your BDH and data files. Such directories help you to store your custom files outside of the SilkPerformer installation directories.

Configuring Directory Settings

1. From the SilkPerformer menu bar, choose **Settings** > **System**.
   The **System Settings** dialog box displays.
2. Click the **Directories** tab.
3. Use the **File locations** area to specify the directories where various SilkPerformer file types are stored. Default directories are set during installation.

   Click **[...]** to the right of each field to browse the directory structure and locate appropriate files and directories.
4. In the **Projects** field, specify the directory where the project files are to be saved.
5. In the **User data files** (`.pem`, `.rnd`, `.csv`, `.txt`, `.idl`) field, specify the directory where the system-defined user data files (`.csv`), certificate files (`.pem`), random data files (`.rnd`), text files (`.txt`), and Interface Definition Language files (`.idl`) are to be stored.

   The compiler first searches for random data files in the directory where the `.bdf` file is located, then in this directory. To use a random data file from a different location, specify a full path for the file in your script file. This location is used for user data files that are delivered with SilkPerformer.
6. In the **Custom user data files** (`.pem`, `.rnd`, `.csv`, `.txt`, `.idl`) field, specify the directory where your self-created user data files (`.csv`), certificate files (`.pem`), random data files (`.rnd`), text files (`.txt`), and Interface Definition Language files (`.idl`) are located.

   In contrast to the specified **User data files** location, this location is used for your own user data files. As with SilkPerformer pre-defined user data files, custom user data files in this directory can be shared across multiple projects.
7. In the **Include files** (`.bdh`) field, specify the directory where the include files (`.bdh`) are located.

   The compiler searches for include files first in the directory where the `.bdf` file is located, and then in this directory. If you want to use an include file from a different location, you can specify a full path in the use statement in your script for your include file. External function prototypes, internal functions, constants, random variables, and external function declarations are reusable in different script files when defined in include files. This location is used for include files that are delivered with SilkPerformer.
8. In the **Custom include files** (`.bdh`) field, specify the directory where your self-created include files (`.bdh`) are located.

   In contrast to the specified **Include files** location, this location is used for your own include files. As with SilkPerformer pre-defined include files, custom include files in this directory can be shared across multiple projects.
9. In the **Working** field, specify the default directory for opening and saving scripts (`.bdf` files).

Source Code Control

If you utilize a source control system to store program sources, you will need to define a source control profile within Test Manager so that Test Manager execution servers can retrieve program sources for test executions. For additional information regarding source control settings, refer to *Test Manager Help*. For additional information regarding SilkPerformer source code control (SCC), refer to *SilkPerformer Help*.

Your project may require additional configurations. For additional information, please refer to *SilkPerformer Help*.

Creating Your SilkPerformer Project

Create a SilkPerformer project of application type **`Web business transaction (HTML/HTTP)`**. Name the project **`SampleWeb`**. Within this project you will record simulated user interactions with ShopIt, the sample Web application that ships with SilkPerformer.

Note: This tutorial assumes that you have working knowledge of SilkPerformer. For additional information regarding the use of SilkPerformer, refer to *SilkPerformer Help*.

Installing ShopIt V 6.0

The SilkPerformer sample Web application is ShopIt V 6.0. ShopIt V 6.0 simulates a simple Web e-commerce site with a catalog of camping merchandise that is available for simulated online purchase. Use this application to experiment with SilkPerformer Web-application capabilities. ShopIt V 6.0 is designed to generate errors, including missing Web links (due to merchandise being out of stock) and session errors.

ShopIt V 6.0 setup is available from the following locations:

- SilkPerformer installation CD `\Extras\ShopItV60.exe`
- Download area: *http://sso.borland.com/download/*
- Web package: Locate `\Extras\ShopItV60.exe` from the location to which you extracted the Web package to.

1. Double-click the `ShopItV60.exe` file in the `\Extras` folder of the installation CD, or from the location where you downloaded it.

   Note: ShopIt V 6.0 requires that IIS (Internet Information Server) be installed on the computer on which ShopIt V 6.0 is installed.

   InstallShield prepares the installation and then the **Welcome** page opens.
2. Click **Next**.
   The **Choose Destination Location** page opens.
3. To change the default installation directory, click **Browse** to open the **Choose Folder** dialog box.

   The default installation destination is displayed in the *Destination Folder* section.

   Specify the folder to which you want to install ShopIt V 6.0, then click **OK** to return to the previous dialog box.
4. Click **Next** to continue the installation process.
   Enter the name of the virtual directory for the Web application in the entry field. This is the name of the directory that will be created on the Web server. Click **Next** to continue.

The **Specify Virtual Directory** dialog box opens.

5. Setup installs the files and configures IIS to run the ShopIt V 6.0 Web application.
   When done, the **Installation Complete** dialog box opens.
6. Click **Finish** on the **Installation Complete** dialog box.

   The ShopIt V 6.0 Web application is now ready for use on the computer where you installed it. You can access ShopIt V 6.0 with a Web browser of your choice by entering the following URL:

   ```
   http://<computer name>/<virtual directory name>/
   ```

   Example:

   If your computer name is `JohnSmith`, and you have not modified the default value `ShopItV60` for the virtual directory, the URL is:

   ```
   http://JohnSmith/ShopItV60/
   ```

   Or, if you access ShopIt V 6.0 from the computer on which you installed it, the following URL also works:

   ```
   http://localhost/ShopItV60/
   ```

Outlining a Project

1. Click **Start here** on the SilkPerformer workflow bar.

   Note: If another project is already open, choose **File** > **New Project** from the menu bar and confirm that you want to close your currently open project.

   The **Workflow - Outline Project** dialog box appears.
2. In the **Project** text box, enter a name for your project.
3. In the **Project description** text box, enter a description for your project.
   This description is for your own project management purposes only.
4. From the **Application** menu tree, choose the type of application that you want to use in your test.

   Note: If you are testing a Web application, choose the **Web business transaction (HTML/HTTP)** option to create simpler scripts while incorporating advanced functionality. Choose the **Web low level (HTTP)** option if you want to put the highest possible load on your application. Web low level scripts are more complex than Web business transaction scripts, which require more effort to customize. It is recommended that you use the **Web business transaction (HTML/HTTP)** instead of the **Web low level (HTTP)** scripts for browser based applications.
5. Click **OK**.

   Note: If you need to add additional resources to the project, right-click the project icon in the **Project** menu tree view. It is particularly important that all the user data files (`.csv`), random data files (`.rnd`), and `.idl` files needed by SilkPerformer are set up for your project.

An icon for the new project appears in the **Project** menu tree.

Recording the Sample Application

Record some interactions with the sample Web application, ShopIt, to flesh out a BDF script for your newly created **`SampleWeb`** project. For additional information on recording scripts, refer to the *SilkPerformer Help*.

1. Click **Model Script** on the workflow bar.

The *Workflow - Model Script* dialog box opens.

2. In the URL text box, type the URL of your sample application.
   For example **`http://localhost/shopItV60/default.asp`**.
3. Click **OK**.
   The SilkPerformer Recorder starts.
4. Interact with the sample application.
5. In the SilkPerformer Recorder, click **Stop** to stop the recorder.
6. Save the newly recorded script as `SampleWeb.bdf`.

Executing a Try Script Run

Evaluate the readiness of the test script that you have recorded by executing a Try Script run. Try Script runs are SilkPerformer trial test runs that determine if recorded scripts run without error. If your script can not be run, use TrueLog Explorer to customize your script. For additional information, refer to *TrueLog Explorer Help*.

Note: ShopIt is designed to generate errors, including missing Web links and session errors. If you experience a session error, proceed as explained in *Customizing Your Test Script in TrueLog Explorer*.

Executing a Try Script Run from SilkPerformer

1. On the SilkPerformer workflow bar, click **Try Script**.
   The **Try Script** dialog box opens.
2. Click **Run**.

Note: To open TrueLog Explorer during the Try Script run, check the **Animated** check box on the **Try Script** dialog box.

Customizing Your Test Script in TrueLog Explorer

The ShopIt sample application is designed to generate errors, including missing Web links due to out-of-stock merchandise and session errors. Session errors result when session information embedded in Javascript (for example a session ID) is not detected by SilkPerformer context management during script replay.

Your `SampleWeb.bdh` script must be customized before it can run correctly because it contains a static session ID. To customize the script, you need to utilize TrueLog Explorer, the SilkPerformer script-customization tool.

Note: For additional information on TrueLog Explorer and the customization of test scripts, refer to TrueLog Explorer Help.

Customizing Session Handling

You can create a new recording rule in TrueLog Explorer using values that you have specified in a parsing function. For example, creating a recording rule from a session-customization parsing function allows you to record an application while avoiding session customization issues entirely.

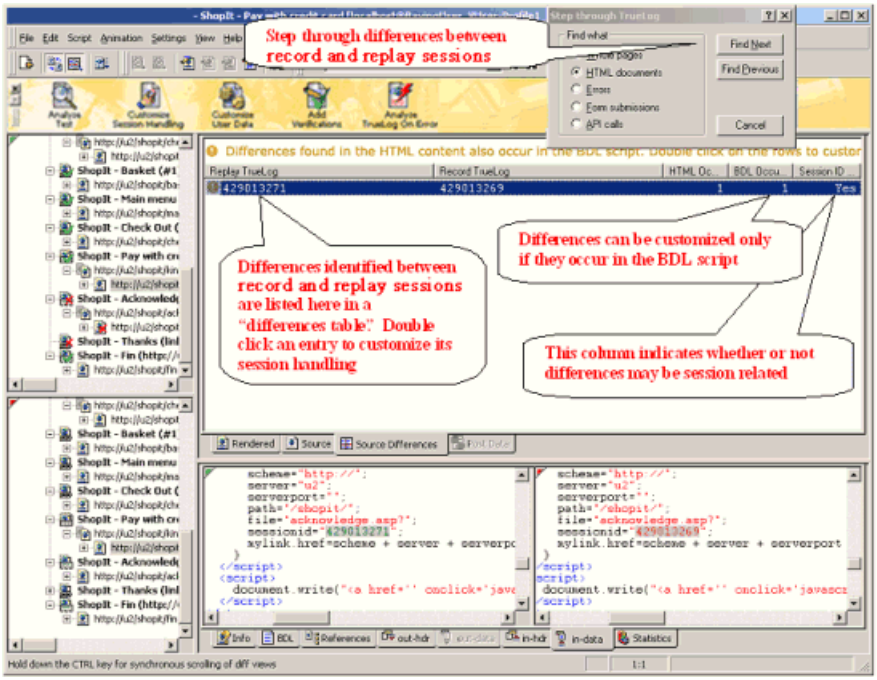
1. Select a TrueLog in the menu tree.
2. Click **Customize Session Handling** on the workflow bar.
   The **Workflow - Customize Session Handling** dialog box opens.
3. Click the **Find differences** link to view a differences table on the **Source Differences** page.

   This step reveals instances in which the server responded with different (or dynamic) information during the replay session compared to of the record session. Such static information may need to be replaced with a variable.

   Note: When the corresponding record TrueLog is not already open, a dialog box opens asking if you want to have the corresponding record TrueLog opened.

   a) Click **OK**.
4. Step through the HTML server responses using the **Step Through TrueLog** dialog box.

   Recorded responses are displayed alongside the corresponding replayed responses. Only those differences that indicate that static information is included in the test script and being sent back to the server need to be parsed. For example, a difference between replay and record sessions might be due to an e-commerce site running out of stocked merchandise. Such a difference would not be appropriate for script customization because it is not session related.



   Tip: The column headers on the **Source Differences** page offer helpful mouse-over tooltips that describe the elements contained in each column.
5. Double-click one of the errors listed on **Source Differences** page.
   The **Insert parsing function** dialog box opens with the boundary values already inserted. There is no need to locate and enter these values manually.
6. *Optional:* Click **Create Recording Rule** to create a recording rule based on the values of the parsing function. This enables you to record the application in the future without having to care about session customization again.
7. Back on the **Insert parsing function** dialog box, click **OK**.
8. Click **Customize Session Handling** on the workflow bar once the script has been successfully modified.
9. Click **Try Script Run** to see if the script runs correctly now that session handling has been modified.

A new Try Script run is initiated.

10. Analyze the results of the subsequent test run to determine whether or not session handling customization was successful.

Uploading Your Project to Test Manager

After you have created a SilkPerformer project, recorded a script based on the sample Web application, customized the session data in your recorded script using TrueLog Explorer, and confirmed that your test script can be run via a Try Script run, you need to create a Test Manager test definition to which you can upload your SilkPerformer project. Name the new test definition *ShopItV_Increasing*.

Note: For the purposes of this tutorial it is assumed that you are familiar with the functionality of Test Manager. For additional information regarding the use of Test Manager, refer to *Test Manager Help*.

Setting Up a Test Manager Project and Test Container

Before you can upload your SilkPerformer project to Test Manager, you must configure a Test Manager project with a test plan structure to which you can upload your SilkPerformer project.

1. From Test Manager, create a new Test Manager project called *Sample Web Test*.
2. Within the new project, create a new test container called *SilkPerformer Test Definitions*.

Uploading Projects to SilkCentral Test Manager

Before uploading a SilkPerformer project to Test Manager, ensure that you have properly configured the project's workload through SilkPerformer. Workload settings are subsequently specified along with uploaded projects in the SilkPerformer **Test Properties** portion of the **Properties** page in Test Manager's **Test Plan** unit.

1. In SilkPerformer, open the project that you want to upload and choose **File** > **Upload Project to Test Manager** .

   If the project has previously been uploaded to Test Manager, the association with a Test Manager test definition is already known and preselected. Click **Next** and go to step 3.
2. From the **Projects** list, select the SilkCentral Test Manager project to which you want to upload the SilkPerformer project and click **Next**.

   You can select an existing test definition to replace or you can create a new test definition by right-clicking the appropriate folder or container choosing **New Child Test Definition**.

   You can also create a new folder within an existing test container by right-clicking a container and choosing **New Child Test Folder**.

   Note: Newly created test definitions and folders are displayed with bold text to indicate that they have not yet been written to the Test Manager database. If you click **Cancel**, the items you have created are not saved.
3. Check the **Enable Results Upload** check box to select specific test run results to upload with the project and then click **Next**.

   If you do not want to upload test run results at this time, do not check the **Enable Results Upload** check box. Instead, click **Finish**.

4. On the **Select results** page, check the appropriate check boxes in the **Results** list to select the results that you want to upload along with the project and then click **Next**.
5. Specify version and build numbers for the assigned product to which the uploaded results belong.
6. Click **Finish** to upload the project to Test Manager.
   The newly uploaded items appear in Test Manager's test plan and **Project** menu tree.

Creating an Execution Definition

Once you have defined your test definition (*ShopItV_Increasing*) you need to create an execution definition for your SilkPerformer test. Execution definitions specify when and where tests are to be executed.

Creating an Execution Definition

1. In Test Manager, navigate to the **Execution** unit.
2. Click **Execution View**.
3. Select the *Sample Web Test* project in the **Execution** tree.
4. Click **New Execution Definition** on the toolbar.
   The **New Execution Definition** dialog box displays.
5. Type **`ShopItV_Increasing_execution`** in the **Name** text box.
6. *Optional:* Type a description for the execution definition in the **Description** text box.
7. Accept the test container that is selected by default in the **Test Container** list box.
   The only test container that exists in this project is *SilkPerformer Test Definitions*.
8. Accept the product version and build that are selected by default in the **Version** and **Build** list boxes.
   The version and build that are associated with the test container are automatically populated.
9. Accept the default priority for the execution definition, which is `low`.
10. Click **OK**.
    The **Execution** tree is updated with your newly created execution definition.

Assigning a Test Definition to an Execution Definition

Assign your *ShopItV_Increasing* test definition to your *ShopItV_Increasing_execution* execution definition.

1. Navigate to **Test Manager** > **Execution**.
2. Click the **Execution View** icon.
3. In the **Execution** tree, select the *ShopItV_Increasing_execution* execution definition.
4. Click the **Assigned Test Definitions** tab.
5. Click **Manual assignment**.
   All test definitions of the *SilkPerformer Test Definitions* test container are displayed in the right-hand **Test Plan** tree.
6. Click the assign arrow of the *ShopItV_Increasing* test definition.

Scheduling Your Execution Definition

After you have assigned a test definition to your execution definition, define the schedule upon which the execution definition is to be executed.

1. Navigate to **Test Manager** > **Execution**.
2. Click the **Execution View** icon.
3. In the **Execution** tree, select the *ShopItV_Increasing_execution* execution definition.
4. Click the **Schedule** tab.
5. Click the **Custom** option button.
   The calendar tool displays.
6. Click the **Edit** icon next to **From** to specify the date and time when the execution schedule is to begin.
   For the purpose of this tutorial, accept the default settings.
7. From the Interval list boxes, select the interval at which the test is to be executed.
   To execute the test daily, select **1** for day, and **0** for hour and minute.
8. In the **Run** section, specify the duration of the schedule.
   For the purpose of this tutorial, click the **Forever** option button.

Note: The calendar tool also enables you to define *Exclusions* and *Definite Runs*. Exclusions are regularly occuring time periods during which executions are not executed, for example weekly-planned system downtime or weekends. Definite Runs are executions that are executed independently of the configured schedule.

Assigning an Execution Server

Assign an execution server to execute your SilkPerformer execution definition. The **Deployment** page in the **Execution** unit of Test Manager displays all of the execution servers that are configured for the selected project and enables you to specify the execution servers on which the selected execution definition is to be executed.

Note: New execution servers are set up at **Administration** > **Locations**. For additional information, refer to *SilkCentral Administration Module Help*.

1. Navigate to **Test Manager** > **Execution**.
2. Click the **Execution View** icon.
3. In the **Execution** tree, select the *ShopItV_Increasing_execution* execution definition.
4. Click the **Deployment** tab.
   The **Deployment** page displays.
5. In the **Execution environment** section of the page, click **Edit**.
   The **Assign Keywords** dialog box displays.
6. Select a keyword from or type a new keyword into the **Select or enter keywords** list box.
   The keyword must match the keywords of one or more execution servers.
7. Click **OK**.
   The keywords are assigned to the execution definition.

Assigning Workload to Agent Clusters

Note: To use this optional part of the tutorial, you must have a test environment with multiple test agents that offer varying software capabilities.

You can distribute the workload that is delivered from Test Manager to SilkPerformer with dynamic workload-to-agent assignment. The dynamic workload-assignment functionality of SilkPerformer matches your specific test requirements to the replay capabilities of the available agent computers at execution time.

Assigning workload to clusters of agents is often preferable to assigning workload to individual agents. *Agent clusters* are groups of agents that can share the same capabilities, such as a Java Development Kit (JDK) installation, an ODBC client, or a Citrix client, or they may consist of agents that have entirely different capabilities, such as the ability to divide the machines in a lab into separate agent groups that can support different teams. With this approach, you select a cluster of agents that is capable of executing the required workload when you configure the workload of your test.

Within SilkPerformer you specify the name of an agent cluster that should deliver the workload for your test. Test Manager provides the list of agent computers that are assigned to the specified cluster. The workload is assigned to specific agents at the moment of execution, based on the capabilities of the individual agents.

When a SilkPerformer execution definition is executed, a Test Manager test agent-clusters XML file is checked out from Test Manager and used for dynamic workload-assignment.

Defining the Capabilities of Your Agent Computers

To facilitate dynamic workload-assignment you must tag each of your agent computers with one or more replay capabilities. For each agent capability, you also need to specify the maximum number of virtual users that the agent is capable of driving.

Replay Capability	Description and Prerequisites
General	Any replay feature that does not require an installation, for example Web, IIOP, COM, or ADO.
Java	A Java run-time environment is available for Java Framework, Oracle Forms, Oracle Applications, Jacada, and Jolt.
ODBC	An ODBC client is installed.
Oracle OCI	An Oracle client is installed.
SAPGUI	A SAPGUI client is installed.
Citrix	A Citrix client is installed.
Tuxedo	A Tuxedo client is installed.
.Net	The .Net Framework is installed.
GuiL Test	Terminal Services is installed and running on this agent. SilkTest and the system under test must be installed.

Defining the Capabilities of an Agent Computer

1. Navigate to **Settings** > **System**.
2. Click **Agents**.
3. Click the **Agent Pool** tab.
4. Select the agent for which you want to define capabilities.
5. Click **Properties**.
6. Click the **Capabilities** tab.

   The **Capabilities** page is pre-populated with suggested replay-capability values that are based on the specifications of the installed hardware.
7. Adjust the **Max. User** values for each capability type, depending on the agent.

   Values from **0** to **9999** are valid.
8. Click **OK** to save your settings.

Preparing an Agent Clusters File for Test Manager

This section describes how to prepare an agent clusters file for Test Manager.

Exporting Agent Pool

Before you can create a Test Manager test-agent cluster file, you need to output the connection properties, capabilities, and system information of the available agents in your agent pool.

1. Navigate to **System** > **Settings** > **Agents** > **Agent Pool**.
2. Click the **Export Agent Pool** button.
3. Specify an appropriate path and filename to save the contents of your agent pool as an XML file.

Example agent pool file

This XML file includes an **AgentPool** root element. Within the root element are **Agent** elements, one for each agent in the agent pool. Within each **Agent** element are elements that convey the connection properties, capabilities, and system information of that agent.

```
<?xml version='1.0' encoding='UTF-8'?>
<AgentPool name="LocalPool">
  <Agent id="LAB108">
    <ConnProperty name="ConnectPort">19200</ConnProperty>
    <ConnProperty name="ConnectSecurePort">19201</ConnProperty>
    <ConnProperty name="IpAddress">192.168.1.108</ConnProperty>
    <ConnProperty name="LastConnectStatus">5</ConnProperty>
    <ConnProperty name="UseAuthentication">false</ConnProperty>
    <Capability maxVU="390" name=".Net"></Capability>
                <Capability maxVU="100" name="Browser-driven"></
Capability>
    <Capability maxVU="39" name="Citrix"></Capability>
    <Capability maxVU="3900" name="General"></Capability>
    <Capability maxVU="39" name="GuiLTest"></Capability>
    <Capability maxVU="390" name="Java"></Capability>
    <Capability maxVU="390" name="ODBC"></Capability>
```

```
    <Capability maxVU="390" name="Oracle OCI"></Capability>
    <Capability maxVU="39" name="SAPGUI"></Capability>
    <Capability maxVU="390" name="Tuxedo"></Capability>
    <SysInfo name="AgentRAC">7803300</SysInfo>
    <SysInfo name="AgentVersion">7.8.0.3332</SysInfo>
    <SysInfo name="Memory">2039 MB</SysInfo>
    <SysInfo name="ProcType"></SysInfo>
    <SysInfo name="ProcessorCount">1</SysInfo>
    <SysInfo name="ProcessorSpeed">3200 MHz</SysInfo>
    <SysInfo name="ServicePack"></SysInfo>
    <SysInfo name="SysVersion">5.0</SysInfo>
    <SysInfo name="System">WinNT</SysInfo>
  </Agent>
  <Agent id="lab116">
    <ConnProperty name="AuthPassword"></ConnProperty>
    <ConnProperty name="ConnectPort">19200</ConnProperty>
    <ConnProperty name="ConnectSecurePort">19201</ConnProperty>
    <ConnProperty name="EncryptionSSL">false</ConnProperty>
    <ConnProperty name="HTTPTunnel">:8080</ConnProperty>
    <ConnProperty name="IpAddress">192.168.1.116</ConnProperty>
    <ConnProperty name="LastConnectStatus">5</ConnProperty>
    <ConnProperty name="SOCKSTunnel">:1080</ConnProperty>
    <ConnProperty name="UseAuthentication">false</ConnProperty>
    <ConnProperty name="UseHttpTunnel">false</ConnProperty>
    <ConnProperty name="UseSocksTunnel">false</ConnProperty>
    <Capability maxVU="380" name=".Net"></Capability>
                <Capability maxVU="100" name="Browser-driven"></
Capability>
    <Capability maxVU="38" name="Citrix"></Capability>
    <Capability maxVU="3800" name="General"></Capability>
    <Capability maxVU="38" name="GuiLTest"></Capability>
    <Capability maxVU="380" name="Java"></Capability>
    <Capability maxVU="380" name="ODBC"></Capability>
    <Capability maxVU="380" name="Oracle OCI"></Capability>
    <Capability maxVU="38" name="SAPGUI"></Capability>
    <Capability maxVU="380" name="Tuxedo"></Capability>
    <SysInfo name="AgentRAC">7803300</SysInfo>
    <SysInfo name="AgentVersion">7.8.0.3343</SysInfo>
    <SysInfo name="Memory">1983 MB</SysInfo>
    <SysInfo name="ProcType">Intel Pentium IV</SysInfo>
    <SysInfo name="ProcessorCount">2</SysInfo>
    <SysInfo name="ProcessorSpeed">3200 MHz</SysInfo>
    <SysInfo name="ServicePack">Service Pack 2</SysInfo>
    <SysInfo name="SysVersion">5.2</SysInfo>
    <SysInfo name="System">WinNT</SysInfo>
  </Agent>
  <Agent id="lab125">
    <ConnProperty name="ConnectPort">19200</ConnProperty>
    <ConnProperty name="ConnectSecurePort">19201</ConnProperty>
    <ConnProperty name="IpAddress">192.168.1.125</ConnProperty>
    <ConnProperty name="LastConnectStatus">5</ConnProperty>
    <ConnProperty name="UseAuthentication">false</ConnProperty>
    <Capability maxVU="650" name=".Net"></Capability>
                <Capability maxVU="100" name="Browser-driven"></
Capability>
    <Capability maxVU="65" name="Citrix"></Capability>
    <Capability maxVU="6500" name="General"></Capability>
    <Capability maxVU="65" name="GuiLTest"></Capability>
    <Capability maxVU="650" name="Java"></Capability>
    <Capability maxVU="650" name="ODBC"></Capability>
    <Capability maxVU="650" name="Oracle OCI"></Capability>
```

```
    <Capability maxVU="65" name="SAPGUI"></Capability>
    <Capability maxVU="650" name="Tuxedo"></Capability>
    <SysInfo name="AgentRAC">7803300</SysInfo>
    <SysInfo name="AgentVersion">7.8.0.3371</SysInfo>
    <SysInfo name="Memory">3318 MB</SysInfo>
    <SysInfo name="ProcType"></SysInfo>
    <SysInfo name="ProcessorCount">2</SysInfo>
    <SysInfo name="ProcessorSpeed">3000 MHz</SysInfo>
    <SysInfo name="ServicePack"></SysInfo>
    <SysInfo name="SysVersion">5.2</SysInfo>
    <SysInfo name="System">WinNT</SysInfo>
  </Agent>
  <Agent id="lab47">
    <ConnProperty name="ConnectPort">19200</ConnProperty>
    <ConnProperty name="ConnectSecurePort">19201</ConnProperty>
    <ConnProperty name="IpAddress">192.168.1.47</ConnProperty>
    <ConnProperty name="LastConnectStatus">5</ConnProperty>
    <ConnProperty name="UseAuthentication">false</ConnProperty>
    <Capability maxVU="180" name=".Net"></Capability>
                <Capability maxVU="100" name="Browser-driven"></
Capability>
    <Capability maxVU="18" name="Citrix"></Capability>
    <Capability maxVU="1800" name="General"></Capability>
    <Capability maxVU="18" name="GuiLTest"></Capability>
    <Capability maxVU="180" name="Java"></Capability>
    <Capability maxVU="180" name="ODBC"></Capability>
    <Capability maxVU="180" name="Oracle OCI"></Capability>
    <Capability maxVU="18" name="SAPGUI"></Capability>
    <Capability maxVU="180" name="Tuxedo"></Capability>
    <SysInfo name="AgentRAC">7803300</SysInfo>
    <SysInfo name="AgentVersion">7.8.0.3414</SysInfo>
    <SysInfo name="Memory">1007 MB</SysInfo>
    <SysInfo name="ProcType"></SysInfo>
    <SysInfo name="ProcessorCount">2</SysInfo>
    <SysInfo name="ProcessorSpeed">2593 MHz</SysInfo>
    <SysInfo name="ServicePack"></SysInfo>
    <SysInfo name="SysVersion">5.1</SysInfo>
    <SysInfo name="System">WinNT</SysInfo>
  </Agent>
</AgentPool>
```

You can now use the XML data in the exported agent-pool file to create a Test Manager test-agent clusters file.

Creating a Test Manager Agent-Clusters File

Before you can complete this task you must export the SilkPerformer agent pool as an XML file (which includes the connection properties, capabilities, and system information of an agent) for each agent in your agent pool.

You only need to create one Test Manager agent-clusters file. The file may contain one or more agent clusters, each of which specifies its associated agents including their capabilities, connection properties and system information. The contents of the file you create will be available to all SilkPerformer users on the **Workload Configuration** dialog when they select the **Dynamic assignment to Test Manager agent cluster** option.

You must create a Test Manager agent-clusters file if you intend to run your test against a Test Manager agent cluster (this is configured by clicking the **Dynamic assignment to Test Manager agent**

cluster button on the **Workload Configuration** > **Agent Assignment** tab). Workloads that use a Test Manager agent cluster can be executed from within the SilkPerformer Workbench and they can be scheduled as Test Manager test definitions.

1. Create an empty XML file on your local system.

   This file must be accessible by Test Manager. It can be placed under source control within your Test Manager directory structure.
2. Use the contents of an exported agent pool file to build the agent-clusters file as structured in the example below.

   The contents of an exported agent-pool file and the agent-clusters file are nearly identical, so this typically only involves enclosing the **<AgentPool/>** elements of the exported agent-pool file within a **<SctmAgentClusters/>** element.

Example of a manually created Test Manager agent-clusters file

This manually created agent-clusters file includes a **SctmAgentClusters** root element. Within the root element are **AgentPool** elements, one for each agent pool in the cluster. Within each **AgentPool** element are **Agent** elements that convey the connection properties, capabilities, and system information of the individual agents.

```
<?xml version='1.0' encoding='UTF-8'?>

<SctmAgentClusters>
 <AgentPool name="cluster_1">
  <Agent id="LAB100">
    <ConnProperty name="ConnectPort">19200</ConnProperty>
    <ConnProperty name="ConnectSecurePort">19201</ConnProperty>
    <ConnProperty name="IpAddress">192.168.1.100</ConnProperty>
    <ConnProperty name="LastConnectStatus">5</ConnProperty>
    <ConnProperty name="UseAuthentication">false</ConnProperty>
    <Capability maxVU="390" name=".Net"></Capability>
                <Capability maxVU="100" name="Browser-driven"></
Capability>
    <Capability maxVU="39" name="Citrix"></Capability>
    <Capability maxVU="3900" name="General"></Capability>
    <Capability maxVU="39" name="GuiLTest"></Capability>
    <Capability maxVU="390" name="Java"></Capability>
    <Capability maxVU="390" name="ODBC"></Capability>
    <Capability maxVU="390" name="Oracle OCI"></Capability>
    <Capability maxVU="39" name="SAPGUI"></Capability>
    <Capability maxVU="390" name="Tuxedo"></Capability>
    <SysInfo name="AgentRAC">7803300</SysInfo>
    <SysInfo name="AgentVersion">7.8.0.3332</SysInfo>
    <SysInfo name="Memory">2039 MB</SysInfo>
    <SysInfo name="ProcType"></SysInfo>
    <SysInfo name="ProcessorCount">1</SysInfo>
    <SysInfo name="ProcessorSpeed">3200 MHz</SysInfo>
    <SysInfo name="ServicePack"></SysInfo>
    <SysInfo name="SysVersion">5.0</SysInfo>
    <SysInfo name="System">WinNT</SysInfo>
  </Agent>
  <Agent id="lab101">
    <ConnProperty name="AuthPassword"></ConnProperty>
    <ConnProperty name="ConnectPort">19200</ConnProperty>
    <ConnProperty name="ConnectSecurePort">19201</ConnProperty>
    <ConnProperty name="EncryptionSSL">false</ConnProperty>
    <ConnProperty name="HTTPTunnel">:8080</ConnProperty>
    <ConnProperty name="IpAddress">192.168.1.101</ConnProperty>
    <ConnProperty name="LastConnectStatus">5</ConnProperty>
```

```
    <ConnProperty name="SOCKSTunnel">:1080</ConnProperty>
    <ConnProperty name="UseAuthentication">false</ConnProperty>
    <ConnProperty name="UseHttpTunnel">false</ConnProperty>
    <ConnProperty name="UseSocksTunnel">false</ConnProperty>
    <Capability maxVU="380" name=".Net"></Capability>
                <Capability maxVU="100" name="Browser-driven"></
Capability>
    <Capability maxVU="38" name="Citrix"></Capability>
    <Capability maxVU="3800" name="General"></Capability>
    <Capability maxVU="38" name="GuiLTest"></Capability>
    <Capability maxVU="380" name="Java"></Capability>
    <Capability maxVU="380" name="ODBC"></Capability>
    <Capability maxVU="380" name="Oracle OCI"></Capability>
    <Capability maxVU="38" name="SAPGUI"></Capability>
    <Capability maxVU="380" name="Tuxedo"></Capability>
    <SysInfo name="AgentRAC">7803300</SysInfo>
    <SysInfo name="AgentVersion">7.8.0.3343</SysInfo>
    <SysInfo name="Memory">1983 MB</SysInfo>
    <SysInfo name="ProcType">Intel Pentium IV</SysInfo>
    <SysInfo name="ProcessorCount">2</SysInfo>
    <SysInfo name="ProcessorSpeed">3200 MHz</SysInfo>
    <SysInfo name="ServicePack">Service Pack 2</SysInfo>
    <SysInfo name="SysVersion">5.2</SysInfo>
    <SysInfo name="System">WinNT</SysInfo>
  </Agent>
 </AgentPool>
 <AgentPool name="cluster_2">
  <Agent id="LAB200">
    <ConnProperty name="ConnectPort">19200</ConnProperty>
    <ConnProperty name="ConnectSecurePort">19201</ConnProperty>
    <ConnProperty name="IpAddress">192.168.2.200</ConnProperty>
    <ConnProperty name="LastConnectStatus">5</ConnProperty>
    <ConnProperty name="UseAuthentication">false</ConnProperty>
    <Capability maxVU="390" name=".Net"></Capability>
                <Capability maxVU="100" name="Browser-driven"></
Capability>
    <Capability maxVU="39" name="Citrix"></Capability>
    <Capability maxVU="3900" name="General"></Capability>
    <Capability maxVU="39" name="GuiLTest"></Capability>
    <Capability maxVU="390" name="Java"></Capability>
    <Capability maxVU="390" name="ODBC"></Capability>
    <Capability maxVU="390" name="Oracle OCI"></Capability>
    <Capability maxVU="39" name="SAPGUI"></Capability>
    <Capability maxVU="390" name="Tuxedo"></Capability>
    <SysInfo name="AgentRAC">7803300</SysInfo>
    <SysInfo name="AgentVersion">7.8.0.3332</SysInfo>
    <SysInfo name="Memory">2039 MB</SysInfo>
    <SysInfo name="ProcType"></SysInfo>
    <SysInfo name="ProcessorCount">1</SysInfo>
    <SysInfo name="ProcessorSpeed">3200 MHz</SysInfo>
    <SysInfo name="ServicePack"></SysInfo>
    <SysInfo name="SysVersion">5.0</SysInfo>
    <SysInfo name="System">WinNT</SysInfo>
  </Agent>
  <Agent id="lab201">
    <ConnProperty name="AuthPassword"></ConnProperty>
    <ConnProperty name="ConnectPort">19200</ConnProperty>
    <ConnProperty name="ConnectSecurePort">19201</ConnProperty>
    <ConnProperty name="EncryptionSSL">false</ConnProperty>
    <ConnProperty name="HTTPTunnel">:8080</ConnProperty>
    <ConnProperty name="IpAddress">192.168.2.201</ConnProperty>
```

```
    <ConnProperty name="LastConnectStatus">5</ConnProperty>
    <ConnProperty name="SOCKSTunnel">:1080</ConnProperty>
    <ConnProperty name="UseAuthentication">false</ConnProperty>
    <ConnProperty name="UseHttpTunnel">false</ConnProperty>
    <ConnProperty name="UseSocksTunnel">false</ConnProperty>
    <Capability maxVU="380" name=".Net"></Capability>
                <Capability maxVU="100" name="Browser-driven"></
Capability>
    <Capability maxVU="38" name="Citrix"></Capability>
    <Capability maxVU="3800" name="General"></Capability>
    <Capability maxVU="38" name="GuiLTest"></Capability>
    <Capability maxVU="380" name="Java"></Capability>
    <Capability maxVU="380" name="ODBC"></Capability>
    <Capability maxVU="380" name="Oracle OCI"></Capability>
    <Capability maxVU="38" name="SAPGUI"></Capability>
    <Capability maxVU="380" name="Tuxedo"></Capability>
    <SysInfo name="AgentRAC">7803300</SysInfo>
    <SysInfo name="AgentVersion">7.8.0.3343</SysInfo>
    <SysInfo name="Memory">1983 MB</SysInfo>
    <SysInfo name="ProcType">Intel Pentium IV</SysInfo>
    <SysInfo name="ProcessorCount">2</SysInfo>
    <SysInfo name="ProcessorSpeed">3200 MHz</SysInfo>
    <SysInfo name="ServicePack">Service Pack 2</SysInfo>
    <SysInfo name="SysVersion">5.2</SysInfo>
    <SysInfo name="System">WinNT</SysInfo>
  </Agent>
  <Agent id="lab203">
    <ConnProperty name="ConnectPort">19200</ConnProperty>
    <ConnProperty name="ConnectSecurePort">19201</ConnProperty>
    <ConnProperty name="IpAddress">192.168.2.203</ConnProperty>
    <ConnProperty name="LastConnectStatus">5</ConnProperty>
    <ConnProperty name="UseAuthentication">false</ConnProperty>
    <Capability maxVU="650" name=".Net"></Capability>
                <Capability maxVU="100" name="Browser-driven"></
Capability>
    <Capability maxVU="65" name="Citrix"></Capability>
    <Capability maxVU="6500" name="General"></Capability>
    <Capability maxVU="65" name="GuiLTest"></Capability>
    <Capability maxVU="650" name="Java"></Capability>
    <Capability maxVU="650" name="ODBC"></Capability>
    <Capability maxVU="650" name="Oracle OCI"></Capability>
    <Capability maxVU="65" name="SAPGUI"></Capability>
    <Capability maxVU="650" name="Tuxedo"></Capability>
    <SysInfo name="AgentRAC">7803300</SysInfo>
    <SysInfo name="AgentVersion">7.8.0.3371</SysInfo>
    <SysInfo name="Memory">3318 MB</SysInfo>
    <SysInfo name="ProcType"></SysInfo>
    <SysInfo name="ProcessorCount">2</SysInfo>
    <SysInfo name="ProcessorSpeed">3000 MHz</SysInfo>
    <SysInfo name="ServicePack"></SysInfo>
    <SysInfo name="SysVersion">5.2</SysInfo>
    <SysInfo name="System">WinNT</SysInfo>
  </Agent>
 </AgentPool>
</SctmAgentClusters>
```

Once you have created an agent clusters file, you must configure Test Manager to reference the file. Test Manager will copy the file to the execution servers so that whenever a SilkPerformer project with dynamic

workload-assignment is executed, the project will read the file to determine how workload should be allocated to the agents within the cluster.

Uploading Your Agent-Cluster File to Test Manager

After you have created the agent-cluster file for your project, you need to upload the file from your source control system to Test Manager. When your SilkPerformer project with dynamic workload-assignment is executed, the project will read the file to determine how workload should be allocated to the agents, which are the execution servers, within your cluster.

1. Navigate to **Administration** > **System**.
2. Click the **Load Test Agent Clusters** tab.
3. Click **Upload**.
   The **Upload Agent Clusters File** dialog box opens.
4. Click **Browse** to browse to and select the agent-clusters file.
5. Click **OK** to confirm your selection.

You have now completed configuration of dynamic workload-assignment for your project.

Assigning Agents to Workload

1. Open the **Workload Configuration** dialog box.

   Note: The **Workload Configuration** dialog box opens automatically when you set up a new workload. Alternatively, you can edit an existing workload profile by right-clicking the workload profile in the menu tree and choosing **Edit Workload** from the context menu.
2. Click the **Agent Assignment** tab.
3. Select the **Assignment type**:

   - **Static assignment to project agents** : Use this method to statically assign specific agent computers (rather than clusters of agents) to your project. Agent locking is disabled with this method. Select this method if you want to use Agents deployed in the cloud.
   - **Dynamic assignment to project agents** : With this method, workload is delivered using dynamic agent-assignment at execution time against the project's agents. Workload delivery is enhanced with agent-capability specifications to create optimized workload-to-agent assignments based on the capabilities of each agent. Agent locking at execution time is enabled with this method.
   - **Dynamic assignment to Test Manager agent cluster** : SilkPerformer workload delivered by way of Test Manager can also use dynamic workload-to-agent assignment. Within SilkPerformer you choose the name of the agent cluster (from the drop list) that should deliver your test's workload. Test Manager then provides the list of agent computers that are assigned to the cluster. Workload is then assigned to specific agents at the moment of execution based on the capabilities of the individual agents. After you connect to Test Manager, you are presented with the list of available agent clusters. In the right-most window, you can view the agents that are currently associated with the selected agent cluster.
4. Define the agents that are to deliver the workload for your test.

   - If you selected **Static assignment to project agents** , you can check the **Even user distribution** check box to distribute all existing user types evenly across all agents, depending on each agent's

general replay capabilities. To use agents that run as virtual machines in the cloud, check the **Use cloud agents** check box. Click **Cloud Agent Manager** to manage your agents in the cloud.

- If you selected **Dynamic assignment to project agents** , workload is delivered automatically using dynamic agent-assignment at execution time against the project's agents.
- If you selected **Dynamic assignment to Test Manager agent cluster** , you are asked to log in to Test Manager. When you are logged in, you can select the available agent cluster.

The **Agents** list box displays the available agents.

5. Check the **Agent resource utilization** check box to assign a maximum percentage of total virtual users that each agent can run based on the agent's replay capabilities.
6. Check the **Balance load across agents** check box to apportion workload across agents.
7. If you selected **Static assignment to project agents** , use the lower window of the **Agent Assignment** page to define workload assignments for user groups.

   Note: Available options vary depending on the selected workload model.
8. Click **User Distribution Overview** to view the assignment of virtual users to the agent computers that are currently available and then click **Close**.
9. Click **OK** to save your settings.

Workload will be assigned to agents based on the agent-assignment settings you have configured. If there are not enough agents with the required capabilities to deliver the required workload, you will be presented with an error message and details regarding the user types that did not receive an agent assignment.

Running Your Test

If you have successfully completed all the preceding procedures, you can now run your SilkPerformer test.

1. Navigate to **Test Manager** > **Execution**.
2. Click the **Execution View** icon.
3. In the **Execution** tree, select the *ShopItV_Increasing_execution* execution definition.
4. Click **Run** on the toolbar.
   Your execution definition is queued on the execution server that you have specified and the **Run** dialog box opens.
5. Check the **Go to Activities page** check box to open the **Activities** page.

Viewing Test Run Details

After you execute a test in Test Manager, the test-activity statistics are available on the **Activities** page. The page includes details about the recently run execution definitions, the currently running execution definitions, and the execution definitions that are scheduled for execution.

1. Navigate to **Test Manager** > **Test Plan**.
2. In the **Test Plan** tree, select the *ShopItV_Increasing* test definition.
3. Click the **Runs** tab.
4. Click the **Run ID** of the last run to display the **Test Definition Run Results** dialog box.

The **Details** page of the dialog box includes the status of the run, SilkPerformer specific details, error messages, and other details of the run.

Analyzing Results in Performance Explorer

Performance Explorer enables in-depth analysis of SilkPerformer test results. The **Analyze Results** option in Test Manager downloads the selected results to Performance Explorer where you can evaluate the results of your optimization efforts. Performance Explorer also enables you to compare statistics from multiple test runs side-by-side in cross load-test reports. For additional information, see *Creating a Cross Load-Test Report*.

Note: For additional information regarding the use of Performance Explorer and the integration of Performance Explorer with Test Manager, refer to *Performance Explorer Help*.

1. Navigate to **Test Manager** > **Test Plan**.
2. In the **Test Plan** tree, select the *ShopItV_Increasing* test definition.
3. Click the **Runs** tab.
4. In the **Actions** column, click the **Analyze Results...** icon.
   A download dialog box opens, showing you the name of the Performance Explorer command file (`.sppecmd`) that you are about to download.
5. Click **Open** to open the results in Performance Explorer.

Performance Explorer opens, connected directly to your Test Manager installation, and fetches the results of the selected execution run. The results are displayed in an overview report. For additional information regarding overview reports, refer to the *Performance Explorer Help*.

Running an Attended Test

You may need to run SilkPerformer tests that are not tied to Test Manager schedules. For this reason the integration of SilkPerformer with Test Manager supports *attended tests*. Attended tests are SilkPerformer tests that are executed manually in SilkPerformer and are not executed automatically based on a predefined schedule in Test Manager.

When you **open** a Test Manager project, you need to check out the project from a source control profile, edit the project, and check the project back in to Test Manager. When you import a project, you only need to download a copy of the project and work with the project independently of Test Manager. Any changes you make to an imported project have no effect on Test Manager. To import a Test Manager project, you only need a Web connection.

Once you have executed an attended SilkPerformer test, upload the test results to Test Manager and associate the results with a SilkPerformer test definition. Finally, analyze the results of the test in Performance Explorer, as described in *Analyzing Results in Performance Explorer*.

Adding Another Test Definition

To continue with this tutorial you need to add another SilkPerformer test definition to Test Manager, as described in *Uploading Projects to Test Manager*. In the *SilkPerformer Test Definitions* container that you created earlier, create a SilkPerformer test definition called *ShopIt_Modem*. Upload your *SampleWeb* project to this new test definition.

Executing an Attended Test Run

1. Navigate to **Test Manager** > **Test Plan**.
2. In the **Test Plan** tree, select your new **`ShopIt_Modem`** test definition.
3. Click the **Properties** tab.
4. In the **SilkPerformer Test Properties** section, click the **Run Attended Test** icon.
   A file download dialog box opens, asking you to confirm that you wish to run the specified SilkPerformer command file, `.spwbcmd`.
5. Click **OK** to open the file in SilkPerformer.
   SilkPerformer is invoked. The **Run Attended** dialog box opens.
6. Select the target directory for your SilkPerformer project and click **OK**.
   The **Workload Configuration** dialog box opens with the worload settings that are associated with the **`SampleWeb`** project..
7. Click **Run** to execute the SilkPerformer test in exactly the same way as if it were executed from Test Manager as an unattended test.

Note: For the purposes of this tutorial, vary your workload settings. For example, you might select a dynamic workload type and a user profile that simulates 56k modem users. After editing the workload settings, click **Run** to begin the test and monitor the test through SilkPerformer.

Uploading Test Results to SilkCentral Test Manager

1. Choose **Results** > **Upload Results to Test Manager** .

   Alternatively, you can right-click a results node and choose **Upload Results to Test Manager** .

   If the project has previously been uploaded to Test Manager, the association with a Test Manager test definition is already known and preselected. Click **Next** and go to step 4.
2. From the **Projects** list, select the Test Manager project to which you want to upload the SilkPerformer test results and click **Next**.
3. From the menu tree, select the test definition to which you want to upload the results and click **Next**.

   Alternatively you can right-click in the tree and choose various menu items to create a new test definition, child test definition, test folder, or child test folder to which the results are saved.
4. On the **Select results** page, check the appropriate check boxes in the **Results** list to select the results to upload along with the project.
5. Click **Next**.
6. On the **Specify product information and result status** page, specify the version and build numbers for the assigned product to which the uploaded results belong.

Note: You can click the **Upload path** link to start Test Manager and go directly to the selected test definition.

7. Click **Finish** to upload the results.
   A confirmation dialog box asks if you want to delete the local copies of the uploaded results.
8. Click **Yes**.

Creating a Cross Load-Test Report

When used in combination with Test Manager, Performance Explorer offers a cross load-test comparison report facility. You can select up to four load-test runs for comparison. Heat fields and rankings help you analyze the results of your optimization efforts across runs.

Cross load-test reports are structured like standard baseline reports. They offer summary reports and statistics regarding transactions, page timers, Web forms, rankings, and errors. For additional information regarding cross load-test reports, refer to *Performance Explorer Help*.

To compare the results of the *ShopItV_Increasing* and *ShopIt_Modem* test executions, follow the steps described in *Generating a Cross Load-Test Report*.

Note: If you have followed the steps in this tutorial, the execution results for the test definitions will be loaded into the **Test Manager** tab in Performance Explorer. If they are not, download the results from the **Run** tab in Test Manager, as described in *Analyzing Results in Performance Explorer*. Alternatively, you can download the results from Performance Explorer by right-clicking within the project tree and selecting **Add Test Manager Results**.

Generating a Cross Load-Test Report

Generate a cross load-test report to compare the results of up to four test executions side-by-side.

For rankings, the baseline test is always listed as the first test.

1. Click the **New Cross Loadtest Report** icon on the toolbar.
   The **Cross Loadtest Report** window opens.
2. Open a maximum of four tests.

   Each new report is assigned a title with a unique number from one to four.

   Each new load-test execution you open appears on the **Test Manager** page alongside the other executions you have imported.
3. Drag the executions that you want to include in the report into the **Cross Loadtest Report** window.

   Alternative: Right-click a test definition in the file menu tree on the Performance Explorer **Test Manager** tab and choose **Cross Loadtest Report**. All executions of the selected test definition will then be automatically added to a cross load-test report.

A **b** icon or **baseline** tag identifies the baseline execution in each report. All measures that are evaluated against the baseline are identified with heat fields.

Note: Pass your cursor over any heat field to view the details of the heat-field setting.

Note: To define a different execution run as the baseline to which the other tests should be compared, right-click the execution and choose **Set as Baseline**.

Analyzing Cross Load-Test Reports

In cross load-test reports, the results of multiple test runs are compared side-by-side. To facilitate comparison, one run is defined as the *baseline* to which the other runs are compared. Any test run can be

set as the baseline. A **b** icon or baseline tag identifies the baseline execution in each report. All measures that are evaluated against the baseline are identified with heat fields.

Example Analysis of a Cross Load-Test Report

In this example, *Loadtest 1* is a load test with a workload of 56k modem users and a dynamic workload model. The test is derived from the *ShopIt_Modem* test definition. *Loadtest 1* is tagged as the baseline execution. *Loadtest 2* is a load test with a workload of high-speed connection users and an increasing workload model. If the heat field next to *Loadtest 2* in the cross load-test report is green, then *Loadtest 2* has performed better than *Loadtest 1*.

Move the cursor over any heat field in the cross load-test report to see the details of the heat field setting.

If *Loadtest 2* were defined as the baseline, then the heat field for *Loadtest 1* would be red, indicating that *Loadtest 1* performed significantly worse than *Loadtest 2*.

Index

T

U

V

W