Borland
A MICRO FOCUS COMPANY

Silk Performer 16.5

Supplemental Tutorials

Borland Software Corporation
700 King Farm Blvd, Suite 400
Rockville, MD 20850

2015-03-16

Contents

Introduction

About this manual

This book provides a series of tutorials designed to ease you into the process of load testing with Silk Performer, and to get you up and running with load testing as quickly as possible.

What you will learn

This chapter contains the following sections:

Overview

Silk Performer is now easier than ever to use, with a new Workflow bar that guides you through the complete load-testing process. Using these tutorials will help you to take full advantage of this ease of use, and to exploit the leading-edge functionality embodied in the load-testing tool of choice in the world of e-business.

You can go directly to the chapter that best suits your needs. Each chapter contains a stand-alone set of tutorials for the most important types of interface in use in e-business today, including Web, database, and middleware applications.

Workflow Bar

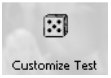
The key to load testing with Silk Performer is the **Workflow** bar at the top of Silk Performer's main window. These icons are your personal guide. They take

you right through the load-testing process from the very beginning of your project to when you view your test results.

Every load test is part of a project. Start by sketching the broad outlines of your project – give it a name, add a short description, and choose the kind of application you want to test.

Every load test uses a set of instructions for what is to happen during the test; these instructions are contained in test scripts. You create a script usually by modeling one on some real-world behavior; you can also use sample scripts or write your own.

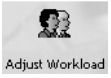
You now do a trial run of your script – not a real load test, just a try-out to make sure that the script works properly and that it is driving your application the way you want.



If you use your present script, you will be running an identical set of instructions many times, and this will not create a realistic test. You need to create lifelike traffic in your load test by customizing your test script.

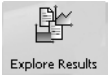
Before you run your full load test, you need to know the ideal, basic performance that your application is capable of. You find this baseline performance by running a minimal load test; it also serves as a try-out of your customized script.

To confirm that you have identified the performance you want from your application, you look at the results from your baseline test. This is the time also prepare for your full load test.



Now you are going to select your workload model and configure your load test in conjunction with your accepted baseline results.

You now run your full load test to assess the performance of your application, using the customized script and the full complement of agent computers and virtual users. There are lots of ways to watch the action as it happens!



To view the results of your load test, use Silk Performer's incredible array of graphic and reporting facilities.

Finally, to reuse a project you can upload it to SilkCentral Test Manager.

Where to Go

Go straight to the tutorials that you need.

- If you are going to use Silk Performer to load test your Web application, proceed to "Load Testing a Web Application".
- To get started with database load testing, either proceed to "Load Testing a Database through ODBC" (if your database server and client communicate via ODBC) or to "Load Testing an Oracle OCI7 Database" (if you are using Oracle software). These tutorials may also be of interest to you if you are first going to run end-to-end load tests of your Web application and then test separate application modules, like database servers.
- If you are going to test your CORBA-based distributed application, proceed to "Load Testing a CORBA Application".
- If the application that you are going to test uses middleware like TUXEDO, proceed to "Load Testing a TUXEDO Application". Afterwards, "Load Testing a Jolt Application" might be of interest to you.

Setup and configuration instructions for all the sample applications used in the tutorials are provided in "Setup and Configuration".

Multi-Protocol Recording

Silk Performer supports multi-protocol recording. Protocol-recording settings are configured using Silk Performer's **Application Profile** dialog box , located in Settings/System.../System Settings - Recorder/Application Profiles/(Add button)/.

Refer to the Silk Performer Online Help for full details regarding multi-protocol recording.

1

Load Testing a Web Application

Introduction

This chapter is available as a separate tutorial book in PDF format. "**Silk Performer Web Load Testing Tutorial**" guides you through all the steps in successfully completing a Web load testing project.

You can find this tutorial in the Doc subfolder of your Silk Performer start menu folder and in the Doc subfolder of your local Silk Performer installation.

If you do not have the installation CD available, contact Customer Care.

The sample application used in this tutorial is called "**ShopIt**" and is contained on the installation CD in the folder "*Extras*".

2

Load Testing a Database through ODBC

Introduction

This chapter contains a series of tutorials that will help you to start load-testing a database through ODBC with Silk Performer.

What you will learn

This chapter contains the following sections:

Overview

Each of the following tutorials provides step-by-step instructions for important Silk Performer tasks. You should perform these tutorials in order, since each tutorial builds on what you have learned in the previous ones.

You will learn how to do the following:

- Execute a predefined load test on your local machine.
- Create a test script by recording traffic using the Recorder.
- Customize the recorded test script.
- Use reporting tools to view multiple and selected throughput and response time information.

Sample: Person1

These tutorials use a sample database application called Person1. This application is included on your Silk Performer CD-ROM (*\Extras\Person1Setup.exe*) and is available for download on http://supportline.microfocus.com/websync/index.asp. After logging in, download the version of the installer under your version of Silk Performer. You should run this setup prior to beginning the tutorials. See "Setting up the Person1 application".

Tutorial 1: Running a predefined test

In this tutorial you will learn to create a load-testing project and execute sample scripts that are provided with Silk Performer. You will run one of the sample scripts with a single virtual user on your local machine.

You will use the test script LoadPers.bdf, which is available at **<Public User Documents>\Silk Performer 16.5\Samples\Database**. This script will create the tables that the Person1 application requires and insert into your database the data that you will need for the remaining tutorials.

Creating a load-testing project

For each server you load test with Silk Performer, you have to create a project file. A project file administers all the settings that are associated with a series of load tests, like the test type and a number of database-specific options.

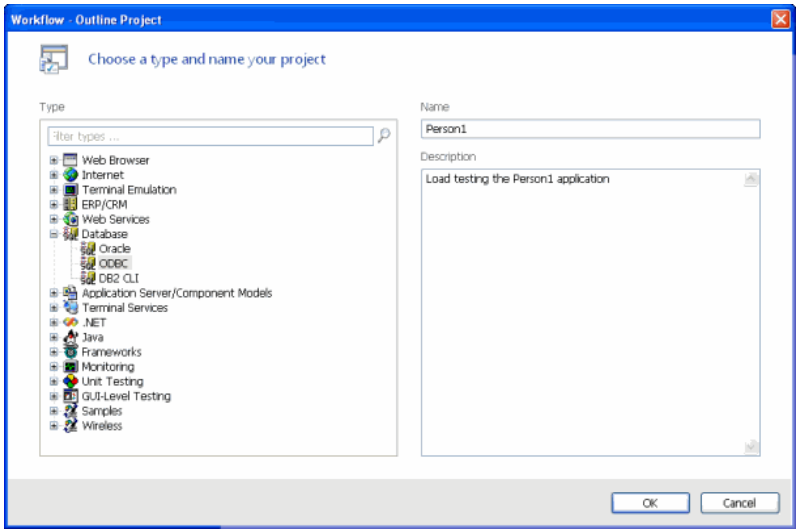
Creating a project

Procedure To create your load-testing project:

1. Open Silk Performer

   If you did not close the project you last worked on, Silk Performer will open with that project.
2. Close any open projects.
3. In the **Workflow** bar, click **Start here**.

The **Workflow – Outline Project** dialog box opens.



4. In the **Name** text box, enter **Person1** as the name for your load-testing project.
5. Enter an optional project description in **Description**.
6. In the **Type** list, select the **ODBC** option.
7. Click **Next**.

   Silk Performer will create a new load-testing project called Person1 with the default settings profile and your local computer as the only agent.

Adding a load-testing script

Each project contains a number of load-testing scripts. A test script defines different groups of virtual users and all the actions that these users have to perform. To be able to execute a load-testing script, you have to add it to your project.

Adding a test script

Procedure To add a predefined script to your project

1. In the menu tree, right-click **Scripts** and select **Add Existing Script**.

   The **Select Script(s)** dialog box opens.
2. Navigate to **<Public User Documents>\Silk Performer 16.5\Samples\Database** and select **LoadPers.bdf**.
3. Click **Open**.

Silk Performer adds the **LoadPers.bdf** test script to the **Scripts** folder of your current project and displays the script in a new editor window.

Configuring load test settings

Each project contains a number of profiles, one of which is activated. A profile is a collection of settings that determine the type of test to be performed, the workload model, and particular options that depend on the server being tested. By default, there is one profile included and activated in your project, the Profile1 profile.

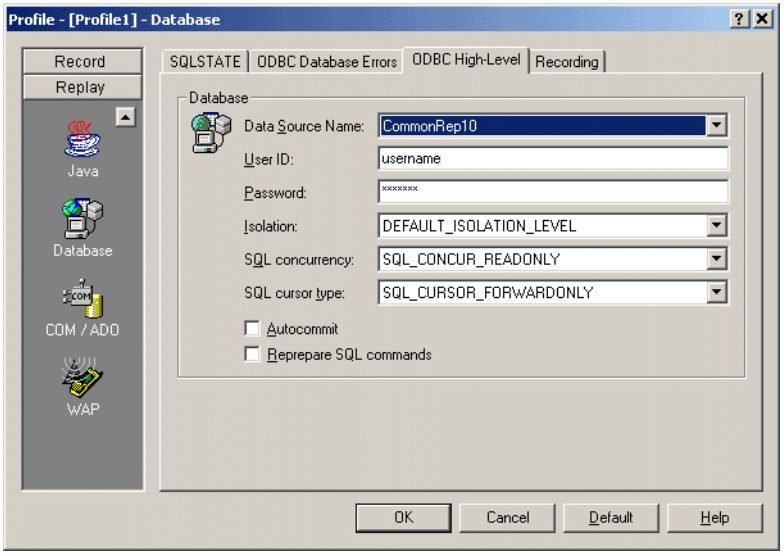
Configure load test settings

Procedure To configure settings for the load test:

1. In the **Project** window, expand the **Profiles** folder.
2. Right-click the **Profile1** profile and select **Edit Profile**.

   The **Profile – [Profile1] – Simulation** dialog box opens.
3. In the shortcut list on the left side, select the **Replay** category and click the **Database** icon.
4. Click the **ODBC High-Level** tab.

   You see the following.



5. From the **Data Source Name** list box, select the ODBC data source of the database that you want to use for the load tests in these tutorials. This should be the local repository in the SQL server database, to avoid any problems.

The test script will create tables and insert data into the database you select here.

6 In the **User ID** text box, enter the user name that you want to use for accessing the database.

7 In the **Password** text box, enter the password for the user name that you have specified.

8 Click **OK**.

Executing the test script

In the LoadPers test script, three different user groups are set up:

- Creator: A virtual user of this group will create all the tables that the Person1 application requires.
- Loader: A virtual user of this group will load sample data into the database so that the Person1 application can be load tested under conditions of real-world use.
- Dropper: A virtual user of this group will remove the sample data and delete the tables that were created earlier for the Person1 application.

To create the tables and insert the sample data for the remaining tutorials, you will first run a virtual user of the Creator group and then a user of the Loader group.

Executing the test script

Procedure To execute the LoadPers test script:

1 In the **Active Script** window, expand the **User Groups** folder.

2 In the **User Groups** folder, right-click **Creator** and select **Run User "Creator"**.

Silk Performer runs a virtual user, performing all the actions defined in the test script, that is, creating the tables in the database that the Person1 application requires. While the virtual user is being run, you can watch progress in the **Monitor** window, for example, the number of transactions executed, the response time of the last transaction, and the average response time.

3 Monitor the load test and wait until the virtual user has finished executing its tasks.

a In the top part of the **Monitor** window, select an agent computer or a user group that you want to monitor.

In the bottom part of the **Monitor** window, Silk Performer displays overview information about all the virtual users running on the selected agent computer or belonging to the selected user group.

b In the bottom part of the **Monitor** window, right-click a virtual user for which you want to view detailed information and select **Show Output**.

Silk Performer displays detailed run-time information about the selected user in the **Virtual User** window at the bottom, for example, the transactions and functions the user executes, and the data that the user sends to and receives from the server. The type of information displayed depends on the options that you select at the top of the Monitor window.

Display Errors. Select this button to display an appropriate error message indicating the probable reason whenever a user-related error occurs.

Display Transactions. Select this button to display a message when a user has finished executing a transaction; the message indicates whether the transaction was successful or not.

Display Functions. Select this button to display a message for each function call a given user performs, including the function name and all its parameters.

Display Info. Select this button to display messages containing additional information about the current action a given user performs.

Display Data. Select this button to show data exchanged with the server.

Display all Errors of all Users. Select this button to display error messages for all errors of all users. Each error message will indicate user, agent and probable cause of the error.

4 From the menu bar, select **File/Close** to close the **Monitor** window.

5 In the **User Groups** folder, right-click **Loader** and select **Run User "Loader"**.

Silk Performer runs a virtual user that inserts sample data into the previously created tables.

What you have learned

In Tutorial 1, you learned how to:

- Create a load-testing project

- Add a predefined script to the project
- Execute the script with one virtual user
- View progress information for the load test

Tutorial 2: Creating a script with the Recorder

The Silk Performer Recorder allows you to capture database traffic transferred between a client application and a server. After you record database traffic, you can save it as a test script. You can then easily customize the script and replay the script to simulate a large number of virtual users. Recording, customizing, and replaying a script will be covered separately in Tutorials 2, 3, and 4. In Tutorial 5, you will analyze the results of the load test you ran.

This tutorial includes the following steps:

- Setting up an application profile
- Recording traffic between a client application and a server
- Generating a test script based on the recorded traffic
- Trying out the generated test script
- Validating the generated test script

Setting up an application profile

Whenever you want the Silk Performer Recorder to capture the traffic that is exchanged between a client application and a server, you have to set up a profile for the client application. This profile specifies the type of the client application and what type of traffic to record.

Setting up an application profile

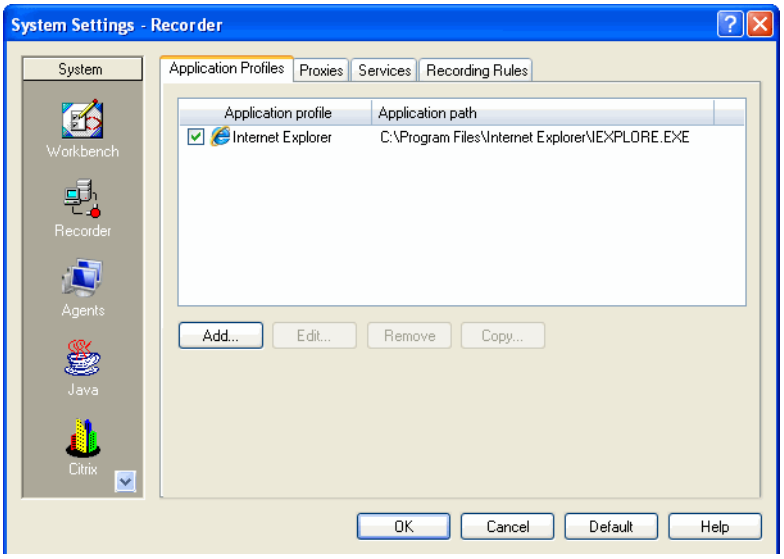
Procedure To set up a profile for the client application:

1. From the menu bar, select **Settings/System**.

   The **System Settings – Workbench** dialog box opens.

2. In the shortcut list on the left side, click the **Recorder** icon.

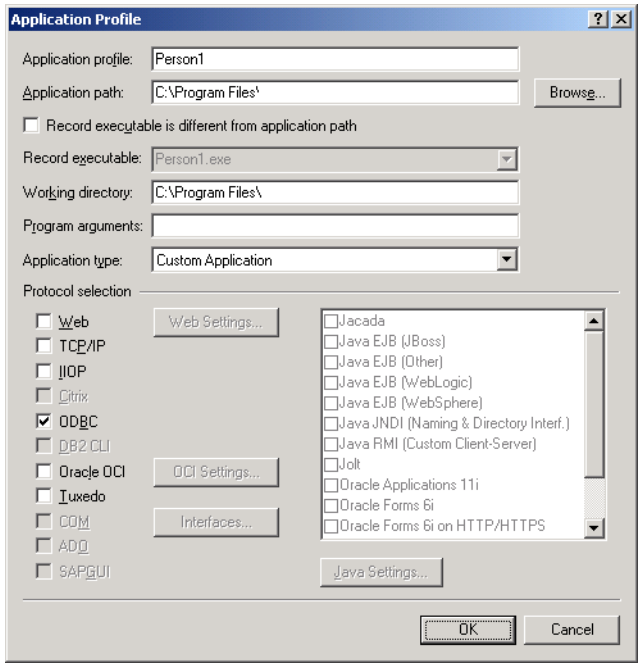
You see the following.



The list contains all the application profiles that are currently set up on your computer.

3 Click **Add**.

The **Application Profile** dialog box opens.



4. In the **Application profile** text nox, enter a name for the application profile, for example, **Person1**.
5. Click the **Browse** button and locate the executable, called Person1.exe. This file is located in the directory you selected during setup (see "Sample: Person1").
6. From the **Application type** list box, select the **Custom Application** option.
7. In the **API selection** area, select the **ODBC** option.
8. Click **OK**.

   Silk Performer adds the new application profile to the profile list in the **System Settings – Recorder** dialog box.
9. Click **OK** to close the dialog box.
10. In the **Project** window, expand **Profiles,** right-click **Profile1** and select **Edit Profile**.

    The **Profile – [Profile1] – Simulation** dialog box opens.
11. Disable **Choose transactions randomly** option. This prevents Logoff from occurring in test script.
12. Click **OK**.

Modeling a test script

To load test a database, the traffic between the client application and a server needs to be recorded and then described in a test script. The script finally enables any number of virtual users to perform actions similar to those that you performed during the recording session.

In order to create as realistic a load test as possible, you will record five separate transactions in a single recording session:

- Logging on to the database
- Searching for customer information
- Updating a customer record
- Inserting a new customer record
- Logging off from the database

When you run the load test, you will create three user groups to which you will assign each of these tasks in a ratio that simulates real-world behavior.

Person1 application notes

Special considerations when working with the person1 application:

- In order to search for a user, one has to type the name into the field and then hit the <tab> key
- In order to insert a new record, one has to type the new details into the fields and then hit the <insert> button (after the information is entered)

Modeling a test script

Procedure To model a load-testing script:

1. In the **Workflow** bar, click **Model Script**.

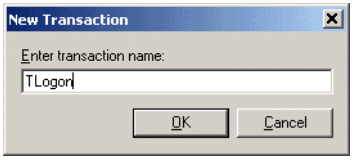
   The **Workflow – Model Script** dialog box opens.

2. From the **Application Profile** list box, select **Person1**.
3. Click **Start recording**.

   Silk Performer opens the Recorder and starts the Person1 application.

4. In the Recorder, click the **New Transaction** button to insert a new transaction into the test script you are generating.

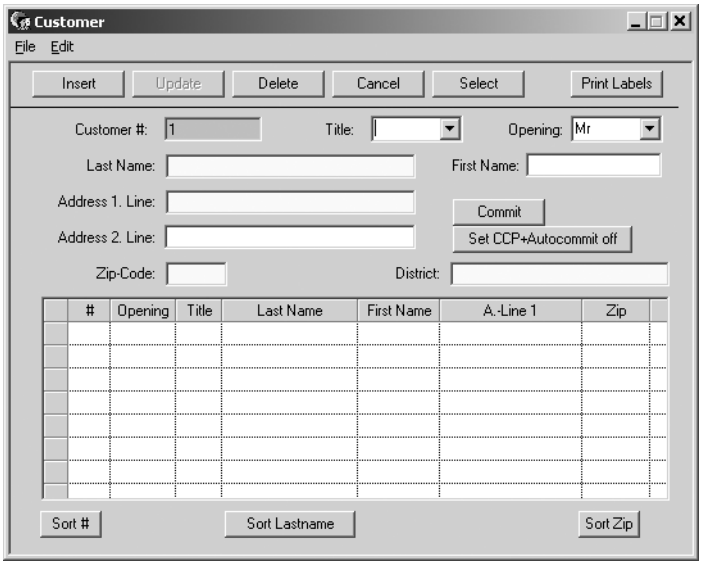
   The following dialog box opens.



5 Create a new transaction called **TLogon**; then click **OK**. A distinct set of time measurements will be made for each transaction you create. When recording traffic, you should create a new transaction for each distinct user session, from connection to shutdown.

6 In the **Login** dialog of the Person1 application, enter the ODBC data source, the user name, and the password for logging on to the database. Then click **OK**.

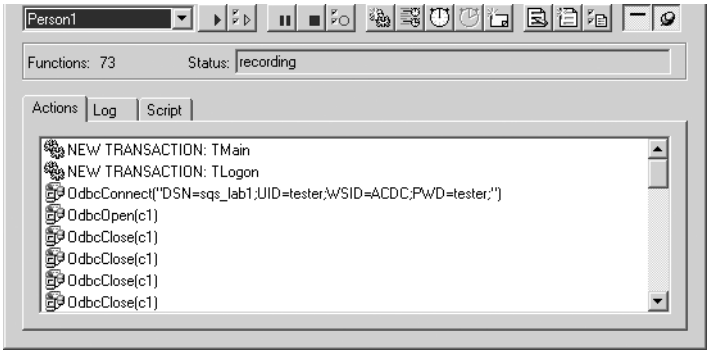
The **Customer** window opens.



As you connect to your database, the Recorder records the database traffic automatically.

7 To view your actions as well as the function calls that the Person1 application performs, click the **Change GUI Size** button.

The **Silk Performer Recorder** window then looks like the following.



8 In the Recorder, create a new transaction called **TSearch**.

9 In the **Customer** dialog box, enter a last name for which you wish to search, for example, **Williams**.

10 In the Person1 application, perform the following additional steps: Before you perform each step, create a new transaction with a name that describes that user session, such as **TUpdate** and **TInsert**.

a Change the address for a customer.

b Insert a new customer record.

11 In the Recorder, create a new transaction called **TLogoff**.

12 In the **Customer** window, select **File/Exit** from the menu bar.

13 In the Recorder, click the **Stop Recording** button.

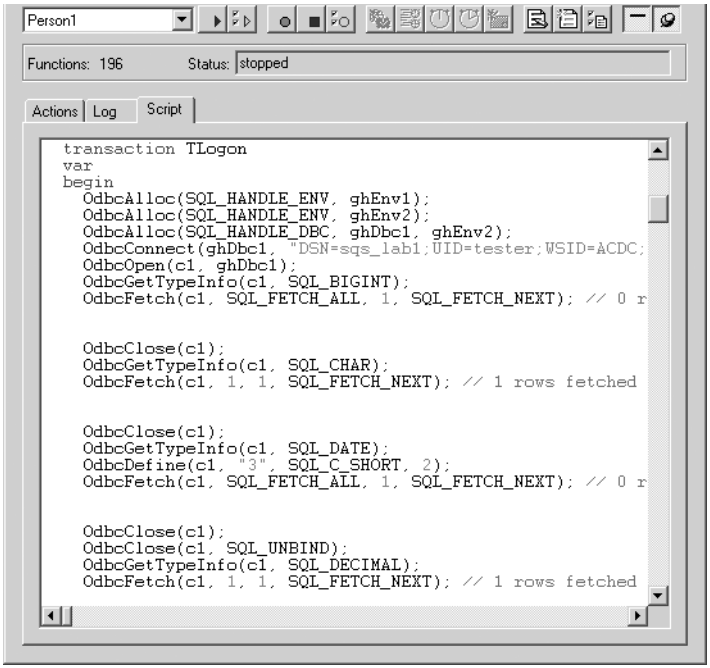
The **Save As** dialog box opens.

14 In the **File name** edit field, enter **Person1**.

15 Click **Save**.

16 You can now decide whether to close the Recorder.

If you keep it open, you can examine the script that has been generated.



17 Close the Recorder.

Silk Performer automatically adds the script to your current load-testing project.

Trying out the generated test script

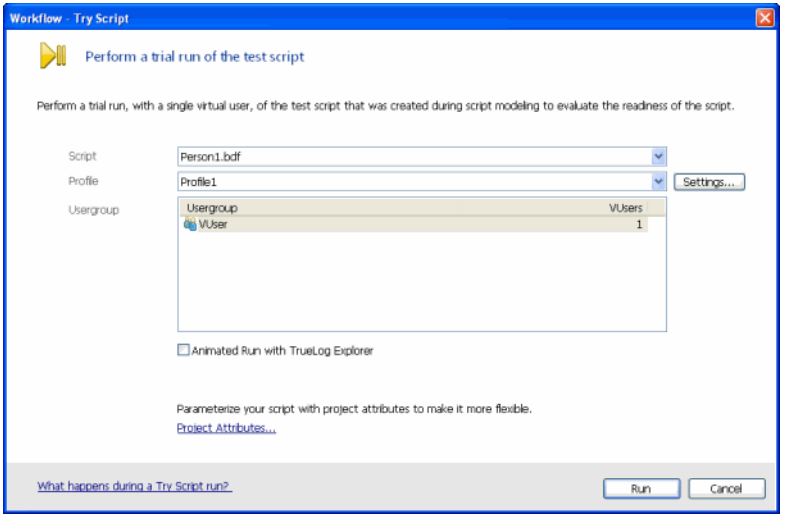
Next you should do a trial run of the test script that was created. The object of the trial run is to ensure that the script is free from error, and that it will reproduce accurately the interaction between the client application and the database server.

Trying out the script

Procedure To try out the generated test script:

1 In the **Workflow** bar, click the **Try Script** button.

The **Workflow – Try Script** dialog box opens.



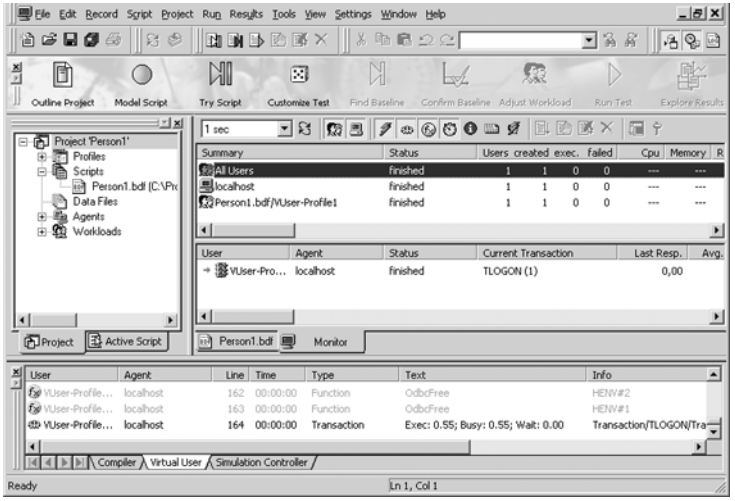
2 From the **Script** list box, select the **Person1.bdf** script.

3 Make sure the **VUser** group is selected in the **Usergroup** list.

4 Click **Run**.

Silk Performer runs a single virtual user, called VUser, which performs all the actions that you have performed during the previous recording session.

While the single-user test is running, you can watch progress in the **Monitor** window.



What you have learned

In Tutorial 2, you learned how to:

- Set up an application profile
- Record traffic between a client application and a database
- Generate a test script based on the recorded traffic
- Try out the generated test script

Tutorial 3: Customizing the generated test script

Customizing the test script you have generated allows you to take the actions performed by a single user and create from them a realistic simulation of multiple users. Two key tasks can help you create a real-world simulation:

- Replacing constant values with random variables
- Specifying which tasks each virtual user will perform

Note You must use Benchmark Description Language (BDL), Silk Performer's high-level scripting language, to customize generated test scripts. Although knowledge of BDL is not required for this tutorial, it might be helpful to browse the introductory BDL Reference topics in the Online Help for more information about BDL before you begin.

Replacing constant values with random variables

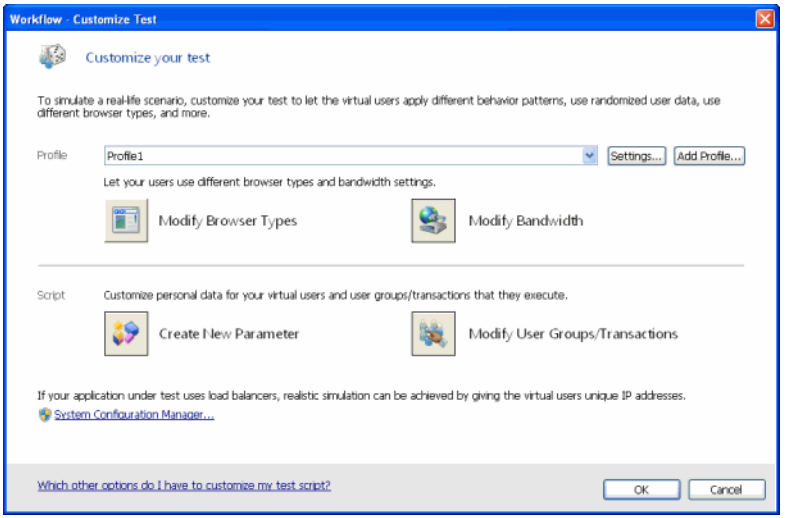
Where randomized data is necessary, random variables are employed to provide realistic user data. In this way, Silk Performer furnishes the virtual users with varied personal data – like name, address, phone number, and so on.

Replacing constant values

Procedure To replace constant values with random variables:

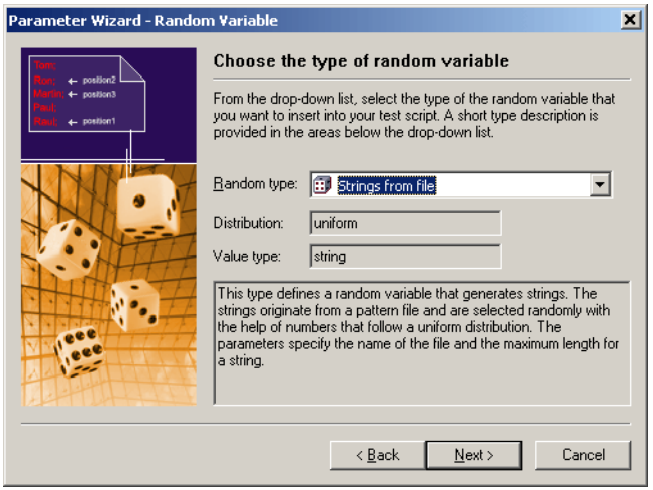
1 In the **Workflow** bar, click **Customize Test**.

The **Workflow – Customize Test** dialog box opens.



2 Click the **Create New Parameter** icon.
The **Parameter Wizard - Create New Parameter** dialog box opens.

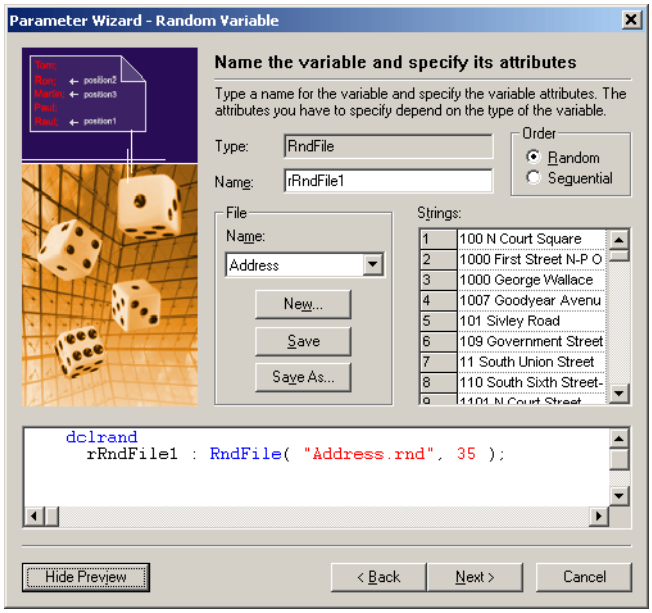
3 Select **Parameter from Random Variable** and click **Next**.
The **Parameter Wizard - Random Variable** dialog box opens.



4 From the **Random type** list box, select the **String from file** type.

5 Click **Next**.

The following dialog box opens.



6 In the **File** area, select **Address** from the **Name** list box.

This random variable will draw from a Silk Performer file containing random addresses and select an address up to 35 characters in length.

7 Click **Finish**.

Silk Performer inserts the following lines into your test script.

```
dclrand
  rRndFile1 : RndFile("Address.rnd", 35);
```

8 Locate the TUpdate transaction.

You can do this by expanding the **Transactions** folder in the **Active Script** window and double-clicking **TUpdate**.

9 Replace the existing address with the **rRndFile1** variable.

This section of the TUpdate transaction should now look like the following. The code shown in bold type is what you have changed.

```
OdbcPrepare(c2, TUpdate_SQL005);
OdbcBind(c2, ":1", SQL_C_CHAR, 10);
OdbcBind(c2, ":2", SQL_C_CHAR, 10);
OdbcBind(c2, ":3", SQL_C_CHAR, 60);
OdbcBind(c2, ":4", SQL_C_CHAR, 40);
OdbcBind(c2, ":5", SQL_C_CHAR, 80);
OdbcBind(c2, ":6", SQL_C_CHAR, 80);
OdbcBind(c2, ":7", SQL_C_CHAR, 10);
OdbcBind(c2, ":8", SQL_C_CHAR, 80);
OdbcBind(c2, ":9", SQL_C_DOUBLE, 0, SQL_INTEGER);
OdbcSetString(c2, ":1", "Mag. ");
OdbcSetString(c2, ":2", "Company ");
```

```
OdbcSetString(c2, ":3", "Williams                                        ");
OdbcSetString(c2, ":4", "John                                            ");
OdbcSetString(c2, ":5", rRndFile1);
OdbcSetString(c2, ":6", "Line2");
OdbcSetString(c2, ":7", "3584 BW   ");
OdbcSetString(c2, ":8", "Mississippi                                     ");
OdbcSetFloat(c2, ":9", 1813.00);
OdbcExecute(c2);
OdbcCommit(ghDbc1);
```

10 Locate the TInsert transaction.

You can do this by expanding the **Transactions** folder in the **Active Script** window and double-clicking **TInsert**.

11 In this transaction, declare a local floating-point variable called fCustomerId.

The code section should then look like the following. The code shown in bold type is what you have inserted.

```
transaction TInsert
  var
    fCustomierId: float;
  begin
    OdbcClose(c1, SQL_UNBIND);
```

12 Within this transaction, locate the section in which the customer ID is queried the second time.

The customer ID is queried twice: once before it is increased (before the Update statement) and once after it has been increased (after the UPDATE statement).

Immediately after the OdbcFetch function calls, insert the following code.

```
fCustomerId := float(OdbcGetString(c2, "1"));
```

This section of the TInsert transaction should now look like the following.

```
OdbcPrepare(c2, TLogon_SQL001);
OdbcClose(c2);
OdbcExecute(c2);
OdbcDefine(c2, "1", SQL_C_CHAR, 12);
OdbcFetch(c2, SQL_FETCH_ALL, 1, SQL_FETCH_NEXT); // 1 rows fetched
fCustomerId := float(OdbcGetString(c2, "1"));
```

Locate the section in which the new customer is inserted.

13 In the first OdbcSetFloat function call, replace the last parameter with the variable fCustomerId.

This will ensure that each time a new customer is inserted, a unique customer ID number is used. This section of the TInsert transaction should now look like the following.

```
OdbcPrepare(c2, TInsert_SQL008);
OdbcBind(c2, ":1", SQL_C_DOUBLE, 0, SQL_INTEGER);
OdbcBind(c2, ":2", SQL_C_CHAR, 10);
OdbcBind(c2, ":3", SQL_C_CHAR, 10);
```

```
OdbcBind(c2, ":4", SQL_C_CHAR, 40);
OdbcBind(c2, ":5", SQL_C_CHAR, 60);
OdbcBind(c2, ":6", SQL_C_CHAR, 80);
OdbcBind(c2, ":7", SQL_C_CHAR, 80);
OdbcBind(c2, ":8", SQL_C_CHAR, 10);
OdbcBind(c2, ":9", SQL_C_CHAR, 80);
OdbcSetFloat(c2, ":1", fCustomerId);
OdbcSetString(c2, ":2", "Company ");
OdbcSetString(c2, ":3", "Mag. ");
OdbcSetString(c2, ":4", "Richards");
OdbcSetString(c2, ":5", "Mike");
OdbcSetString(c2, ":6", "First Avenue");
OdbcSetString(c2, ":7", "");
OdbcSetString(c2, ":8", "10010");
OdbcSetString(c2, ":9", "Manhattan");
OdbcExecute(c2);
OdbcCommit(ghDbc1);
```

14 When you are done making changes to the script file, select **File/Save** from the menu bar to save your changes.

Specifying which tasks each user will perform

Usually a load test simulates a number of different types of users. In the test script, you have to specify which action each type of virtual user will perform.

Assigning tasks to users

Procedure To specify which tasks each user will perform:

1 Navigate to the **dcluser** section of the Person1.bdf script file. You can do this by selecting **User Groups/VUser** in the **Active Script** window.

You see the following.

```
dcluser
  user
    VUser
  transactions
    TInit           : begin;
    TLogon          : 1;
    TSearch         : 1;
    TUpdate         : 1;
    TInsert         : 1;
    TLogoff         : 1;
```

In this section, a single user group is defined, called VUser. By default, this user group will perform the transactions you created, TLogon, TSearch, TUpdate, TInsert, and TLogoff, once each.

2 Edit the **dcluser** section of the script so it looks like the following.

```
dcluser
  user
    Searcher
  transactions
    TLogon          : begin;
    TSearch         : 5;
    TLogoff         : end;


  user
    Updater
```

```
transactions
  TLogon          : begin;
  TSearch         : 3;
  TUpdate         : 2;
  TLogoff         : end;

user
  Inserter
transactions
  TLogon          : begin;
  TSearch         : 1;
  TInsert         : 3;
  TLogoff         : end;
```

In the edited version of the script, the user group Searcher performs the TLogon transaction at the beginning, the TSearch transaction five times, and the TLogoff transaction at the end. The user group Updater performs the TLogon transaction at the beginning, the TSearch transaction three times, the TUpdate transaction twice, and the TLogoff transaction at the end. The user group Inserter performs the TLogon transaction at the beginning, the TSearch transaction once, the TInsert transaction three times, and the TLogoff transaction at the end.

3 Select **File/Save** from the menu bar to save your changes.

4 In the **Workflow** toolbar, click **Customize Test**.

The **Workflow – Customize Test** dialog box opens.

5 Click **OK** to confirm that you are done customizing the test script.

What you have learned

In Tutorial 3, you learned how to:

- Replace constant values with random variables
- Specify which tasks each virtual user will perform

Tutorial 4: Replaying the customized script

After you have created a test script by recording traffic and then customizing the script, you can load test your server by replaying the script. To run a load test, you must perform the following steps:

- Find the test baseline
- Confirm the test baseline
- Set up server monitoring for your load test
- Execute the load test

Finding the test baseline

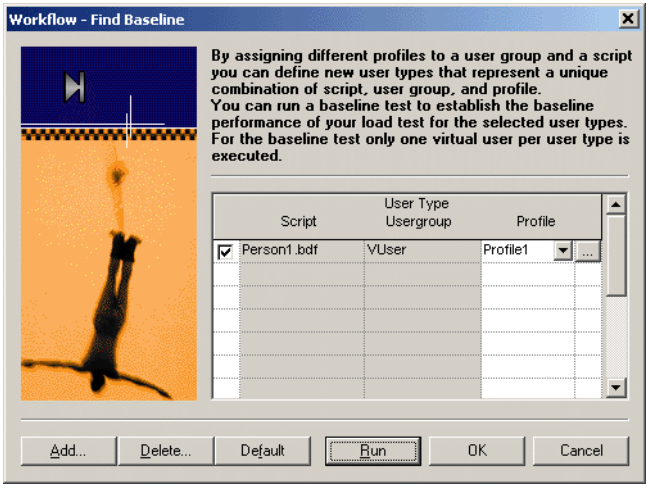
After you have customized the load-testing script, you need to determine baseline performance. This is the ideal performance of your application when it is not under real pressure. To obtain this test baseline, the fully customized script is run with just one user per user group. This will provide you with a solid basis for comparison with the situation when the application is fully taxed by a load test.

Finding the baseline

Procedure To find the load test baseline:

1. In the **Workflow** toolbar, click **Find Baseline**.

   The **Workflow – Find Baseline** dialog box opens.



   This dialog lists all the user groups you have set up in your test script.

2. Click **Run**.

   Silk Performer runs one virtual user from each user group that you have declared in the test script. While the test is running, you can watch progress in the **Monitor** window.

Confirming the test baseline

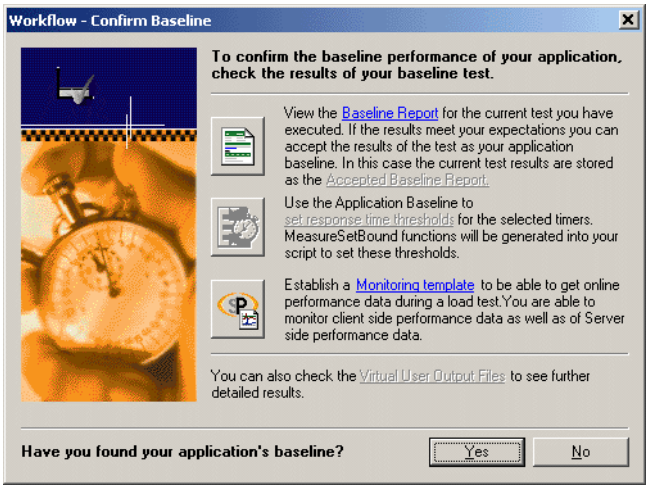
After running the baseline test, you have to confirm the test result reflects the desired performance of the tested server. For this you inspect the results of the test. If they are satisfactory, the baseline is established and will form the basis for comparison with results from full load tests later.

Confirming the baseline

Procedure To confirm the load test baseline:

1 In the **Workflow** toolbar, click **Confirm Baseline**.

The **Workflow – Confirm Baseline** dialog box opens.



2 Click the **Baseline Report** button.

The **Baseline Report** opens.

3 Check the report carefully. In particular, make sure that no errors have occurred during your baseline test.

4 Click **Accept Baseline** in the **Baseline Report**.

5 Click Yes and OK by the Message boxes that comes up.

6 If your baseline report indicates no errors, click **Yes** in the **Workflow – Confirm Baseline** dialog box to confirm that you have found the baseline.

Adjusting the workload

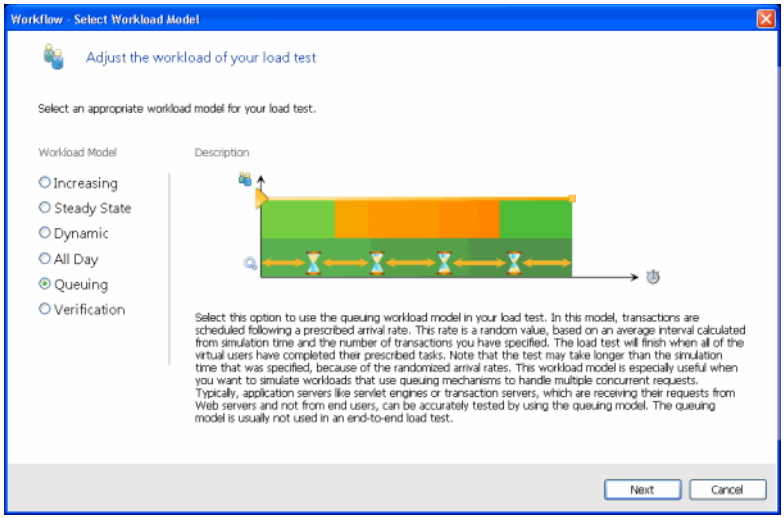
After you have confirmed your baseline, you now select your workload model and prepare the workload to run a full load test.

Adjusting the workload

Procedure To adjust the workload:

1 In the **Workflow** toolbar, click **Adjust Workload**.

The **Workflow - Select Workload Model** dialog box opens.



This dialog box shows all possible workload models, which can be selected.

2 Select **Queuing** workload model.

3 Click **Next**.

4 On the **Workflow Assign Agents** dialog box, select an option and click **OK**.

Silk Performer now opens the Workload Configuration dialog box.

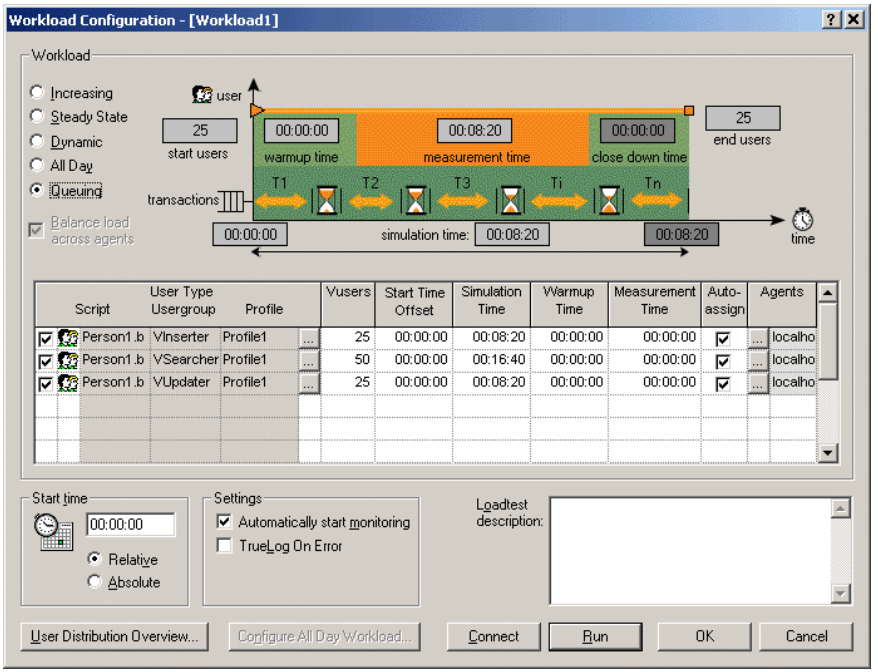
Running the load test

After you have adjust the workload, you run the full load test. The test script is run with multiple virtual users in order to test the database server. A large load test would need the appropriate testing environment to be set up in the local area network, including a full complement of agent computers to host the virtual users. For this tutorial, however, you use only your local computer.

Running the test

Procedure To run the load test:

1 The **Workload Configuration** dialog box is open.



2 In the **VUsers** column, change the number of Inserter users to **25**, the number of Searcher users to **50**, and the number of Updater users to **25**.

3 For each of these user groups, change the simulation time to **300** seconds (5 minutes).

4 Disable all user groups that are declared in the LoadPers test script.

You are now ready to run the load test. When you run a test Silk Performer's sophisticated monitoring tool **Performance Explorer** will automatically generate a live graphical display of a default set of test data for the server you are testing. You can disable this default option by deselecting the setting **Automatically start monitoring**. You can also change the default monitoring settings by pressing the **Active profile** button (See Customizing server monitoring). Once monitoring is set up the way you want it, you can continue and run the test.

5 Click **Run** to start the test now, or Click **OK** to save the settings and start the test later.

When you start the test, Silk Performer runs a load test for the database server with 100 virtual users.

If the (default) option **Automatically start monitoring** is enabled, **Performance Explorer** will open and generate a live graphical display of the specified set of test data. If you wish you can adapt the views by

adding or deleting specific measurements. For this you can expand the tree menu on the left and drag and drop selected measurements into existing or new view graphs.

Customizing server monitoring for your load test

An important feature Silk Performer provides, is server monitoring during an actual load test. Server monitoring will help you locate and analyze bottlenecks on the server. You can separately examine the performance of the operating system, the network and the server application, which in this case is the Web server.

Server monitoring is performed with Performance Explorer, a powerful tool Silk Performer incorporates. When you run a test, the default procedure is that Performance Explorer automatically opens a set of views containing a live graphical display of a selection of test data that is particularly significant for the server you are testing. You can disable this default option by deselecting the setting **Automatically start monitoring** in the **Workload Configuration - [.....]** dialog. You can also customize the default monitoring settings and select a particular set of data sources.

Setting up server monitoring

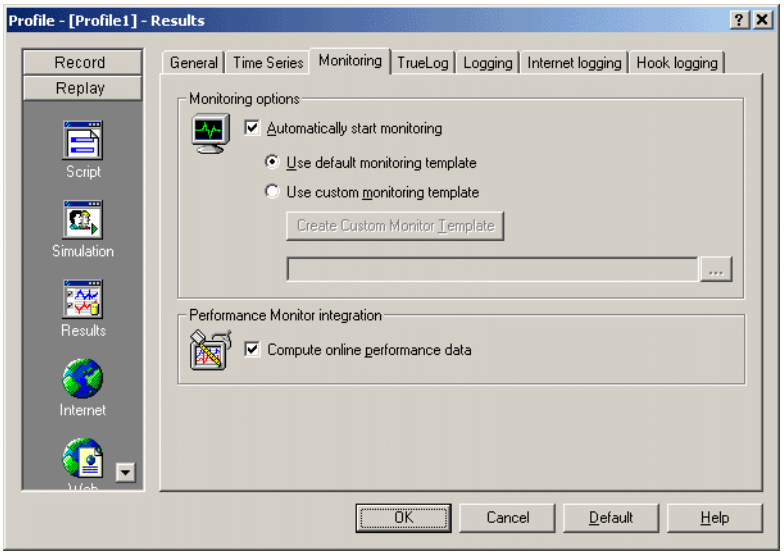
Procedure To customize server monitoring for your load test:

1. In the **Workflow** toolbar, click the **Run Test** button.

   The **Workload Configuration** dialog opens.

2. In the **Settings** section, click the **Active Profile** button.

The **Profile – Results** dialog opens.



3 In the shortcut list on the left side, select the **Replay** category and click the **Results** icon.

4 Select the **Monitoring** tab.

The option **Automatically start monitoring** is activated by default. It means, that when you run a load test, the Performance Explorer will open automatically using a default template that invokes display of relevant client information.

5 If you want to customize monitoring, enable the **Use custom monitoring template**, and click the **Create Customer Monitor Template** button.

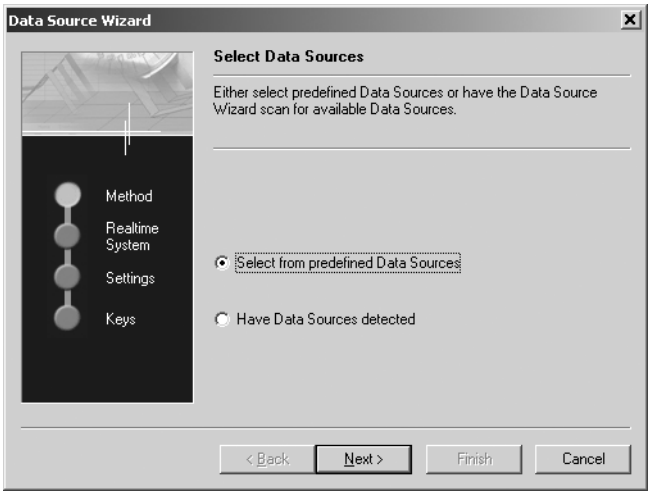
Performance Explorer will create and open a template with the name of your project, which is equal to the default template.

6 Click **Edit Customer Monitor Template**, to customize the new template and change the display of information. You can close or maintain the default monitoring windows and you can add one or more windows to provide additional information.

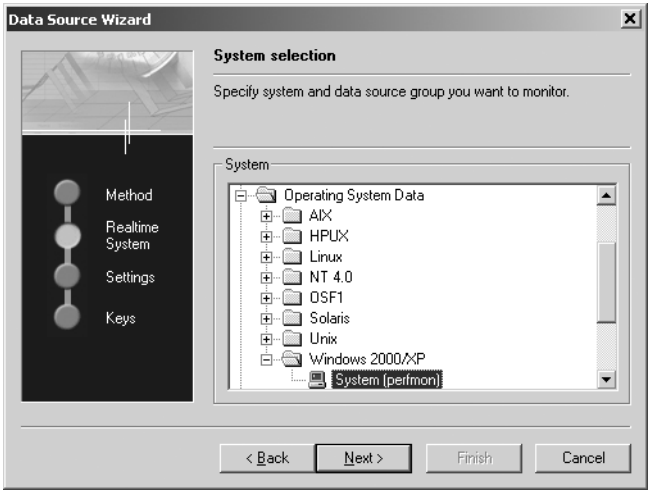
To set up Performance Explorer to display additional information, proceed as follows:

a From the menu bar, select **Monitor/Add Data Source**.

The **Data Source Wizard** opens.



b Click **Next**.
The **Data Source Wizard - System selection** dialog opens.



c In the tree view, expand the folder that corresponds to the operating system on which the database server is running. Within that folder, select **System**.

d Click **Next**.

e In the **Connection Parameters** dialog, enter connection parameters, like the host name or IP address of the database server, the connection port, the user name, and the password.

f The data you have to enter here depends on the operating system that is running on the computer you are monitoring.

g Click **Next** when you have entered the connection parameters.

h In the next dialog, select the performance counters you want to monitor.

i Of particular interest are the processor and memory utilization of the server being tested.

j Click **Finish**.

k Performance Explorer will open a new window, displaying real-time data for the performance counters you have selected.

l In the **Monitor** tree view, right-click the name of the computer you are monitoring and select **Add Data Source**.

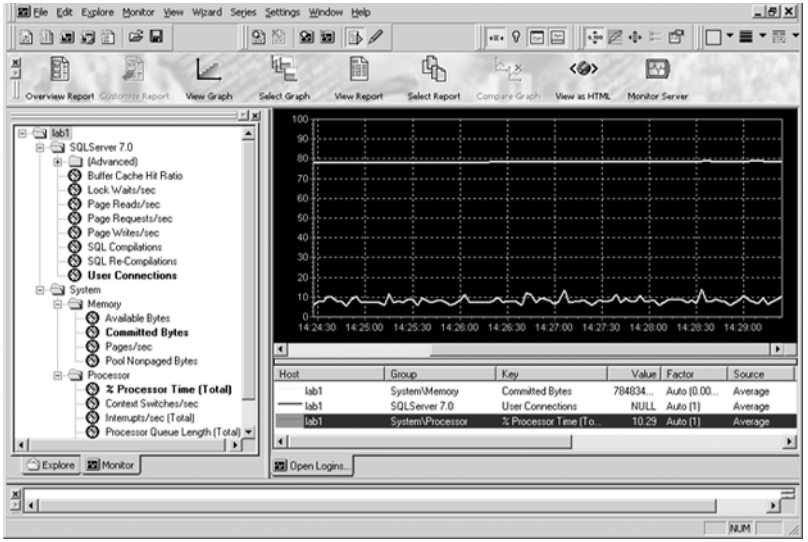
m The Data Source Wizard opens again.

n Repeat step a - i, but this time:

I In the first dialog, select the name of the database server software instead of **System**.

II In the third dialog, select which performance counters you want to monitor.

Later, when you are running the load test, the monitor view will look similar to the following.



What you have learned In Tutorial 4, you learned how to:

- Find the test baseline
- Confirm the test baseline
- Set up server monitoring for your load test
- Adjusting the workload
- Execute the load test

Tutorial 5: Viewing the results of your load test

Silk Performer incorporates **Performance Explorer**, a powerful graphing and analysis tool to help you work with your results information.

Performance Explorer provides a comprehensive array of results information, which can be displayed using advanced features for creating statistical reports and displaying performance results in real time generated graphics. Results information may vary, depending on the type of application and/or server being tested. In general it falls into the following categories:

- Response time information: This is the total time to process a given timer; it provides application performance information from the point of view of the client computer.
- Throughput information: This is the rate at which a given timer is processed on average by the database server; it allows you to analyze performance from the point of view of the server.

Overview report of performed measurements

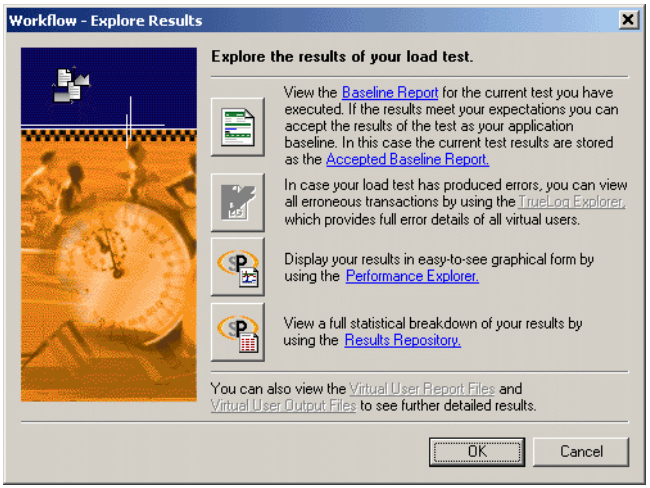
You can consult an HTML overview report that contains results of all performed measurements.

Exploring overview report

Procedure to explore overview report:

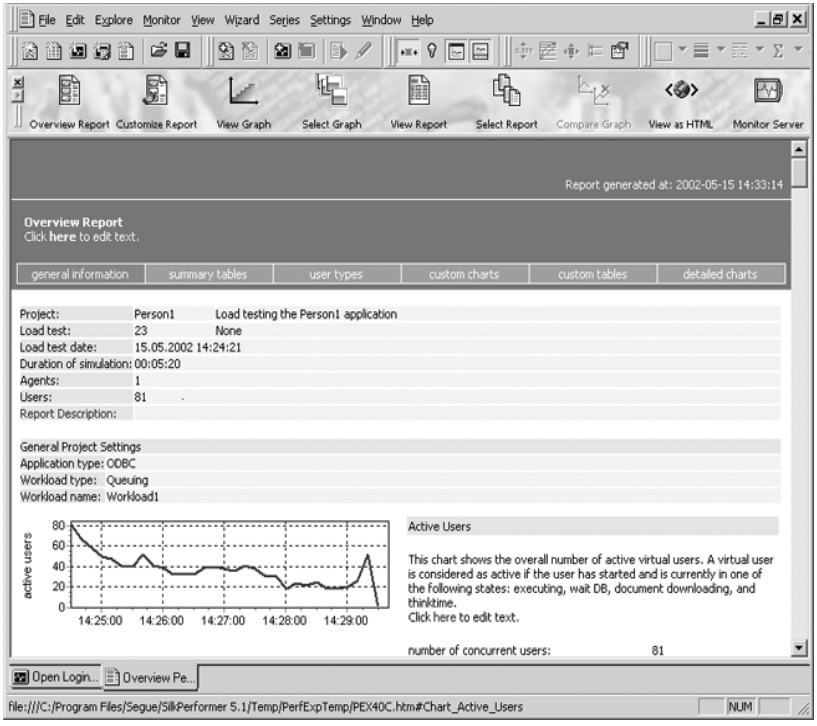
1 In the **Workflow** toolbar, click the **Explore Results** button.

The **Workflow - Explore Results** dialog opens.



2 Click the **Performance Explorer** button.

Performance Explorer opens, and the HTML overview report is generated.



You may consult this report which includes short explanations of the displayed graphs. If you want you can view more detailed information.

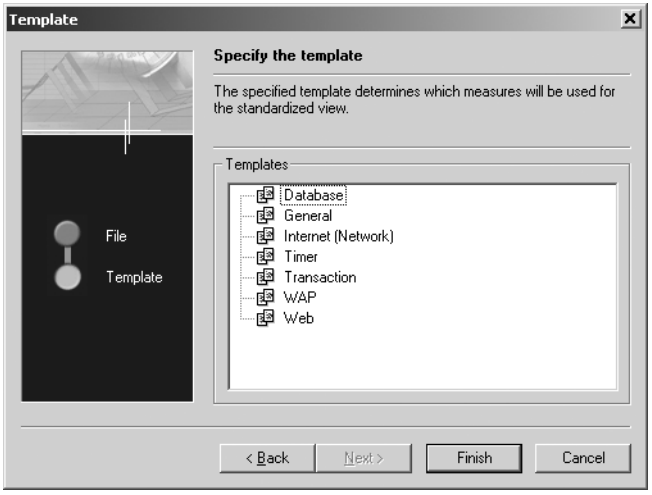
Detailed response time measurements

When you analyze the performance of a database server from the point of view of the client, you examine the response times for all the transactions, SQL statements, and individual timers. This enables you to find out how users will experience their interaction with the database server. A common goal in load testing is to ensure that the response times for all the database operations remain below a specified, critical limit.

Analyzing response times

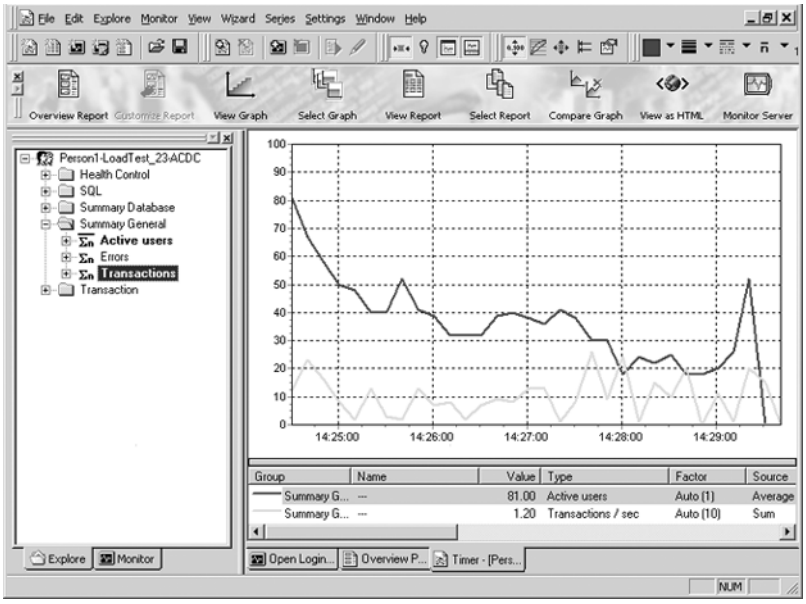
Procedure To explore response time measurements:

1. In the Performance Explorer **Workflow** toolbar, click the **Select Graph** button.
2. The **Template** dialog opens.



3. In the **Templates** area, select the **Timer** option.
4. Click **Finish**.

The Performance Explorer opens a new chart that contains a graph of the active users.



You see the default graph with the average number of active users, and one or more default measurements (if any).

5 Expand the **Explore** tree view.

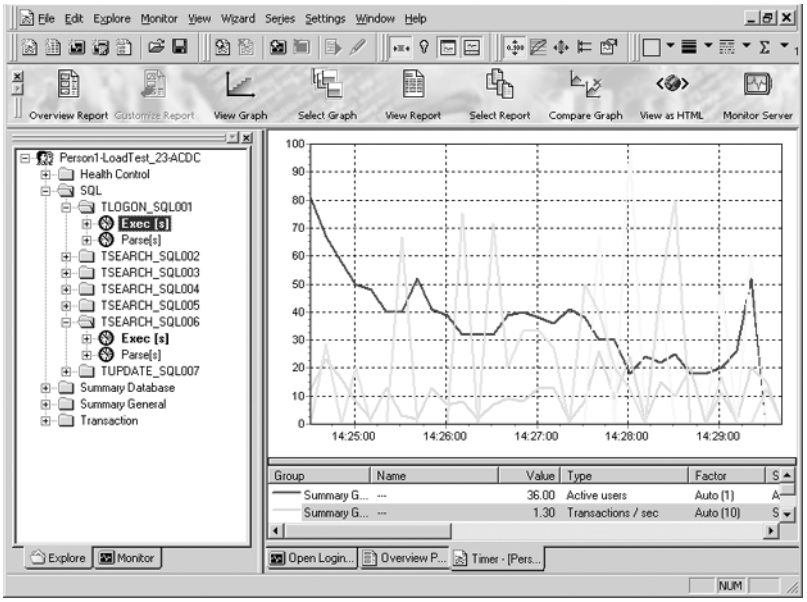
6 In the **Explore** tree view, expand **SQL Commands/TLOGON_ SQL001**.

7 Drag the **Exec** measurement and drop it on the new view.

8 In the **Explore** tree view, expand **SQL Commands/TINSERT_ SQL006**.

9 Again, drag the **Exec** measurement and drop it on the new view.

You see a chart similar to the following.



The load test that you ran in the previous tutorial lasted only a few minutes, so there are only a few measurements, which makes it difficult to draw any conclusions.

In real-life you will run load tests that last several hours, a few days, or even a week. These tests will provide results information that will allow you to analyze the performance of your server accurately.

Exploring throughput measurements

When you analyze the performance of a database server from the point of view of the server, you examine the throughput rate. Throughput is the work done by the server within a specific time interval, for example, the number of requests that the server processes within a second, a minute, or an hour. A common goal in load testing is to ensure that the throughput rate is above a specified, critical limit.

Throughput rate details

Procedure To explore throughput measurements:

1. In the Performance Explorer **Workflow** bar, click the **Select Graph** button.

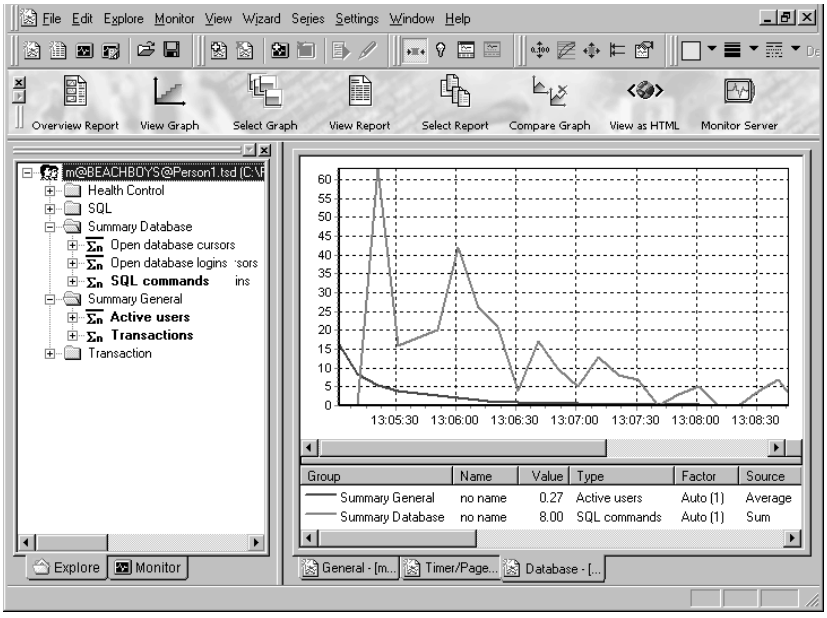
   The **Template** dialog opens.

2. In the **Templates** area, expand the **Throughput** folder.

3 In the **Throughput** folder, select **Database**.

4 Click **Finish**.

The Performance Explorer opens a new chart that looks like the following.



Again, since the load test that you ran lasted only five minutes, it is difficult to derive conclusions from the results.

What you have learned

In Tutorial 5, you learned how to:

- Explore response time measurements
- Explore throughput measurements

3

Load Testing an Oracle OCI7 Database

Introduction

This chapter contains a series of tutorials that will get you started using Silk Performer to load test Oracle OCI7 databases.

What you will learn

This chapter contains the following sections:

Overview

Each of the following tutorials provides step-by-step instruction for important Silk Performer tasks. These tutorials should be followed in order, as each tutorial builds on what is covered in the preceding tutorials.

You will learn how to:

- Execute a predefined load test on your local machine.
- Create a test script by recording database traffic using the Recorder.
- Customize a recorded test script.
- Run a customized script, using multiple virtual users.

- Use reporting tools to view select throughput and response time information.

Sample: PersonPB

These tutorials use a sample database application called **PersonPB**. This application is included on the Silk Performer CD-ROM and is installed in the SampleApps\Oracle subfolder of the Silk Performer home directory.

Oracle client

To perform this tutorial **Oracle Client** software must be installed on your computer.

Tutorial 1: Running a predefined test

In this tutorial you will learn how to create a load-testing project and execute sample scripts provided with Silk Performer. You will run a sample script with a single virtual user on your local machine.

You will use the **OraLoadPers.bdf** test script, which is available at **<Public User Documents>\Silk Performer 16.5\Samples\Database\Oracle**. This script creates the tables that the PersonPB application requires and inserts data into your database that is required for the remaining tutorials.

Creating a load-testing project

For each server you load test with Silk Performer, you must create a project file. A project file administers all the settings that are associated with a series of load tests, including test type, workload model, and options that depend on the server type under test. In addition, project files enable you to easily switch between different load-testing projects without having to define new settings each time.

Creating a project

Procedure To create your load-testing project:

1. Open Silk Performer.

   Close any open projects (Silk Performer opens with the last project you worked on).
2. On the **Workflow** bar, click **Start here**.

   The **Workflow – Outline Project** dialog box opens.
3. In the **Name** text box, enter **PersonPB** as the name of your load-testing project.
4. Enter an optional project description in **Description**.
5. In the **Type** list, select the **Oracle** option.
6. Click **Next**.

Silk Performer creates a new load-testing project called **PersonPB** with the default settings profile and your local computer as the only agent.

Adding a load-testing script

To run a load test, you must have a test script. Each project contains one or more load-testing scripts. Such scripts define one or a group of virtual users and the actions they perform. Usually, such scripts are modeled on real-world behavior. In this tutorial you will use a **Sample script** provided by Silk Performer.

Before you can run the load-testing script, you must add it to your project.

Adding a test script

Procedure To add a predefined script to your project:

1. In the menu tree, right-click **Scripts** and select **Add Existing Script**.

   The **Select Script(s)** dialog box opens.

2. Navigate to **<Public User Documents>\Silk Performer 16.5\Samples\Database\Oracle** and select **OraLoadPers.bdf**.

3. Click **Open**.

   Silk Performer adds the **OraLoadPers.bdf** test script to the **Scripts** folder of your current project and displays the script in a new editor window.

Personalizing the sample script

The **OraLoadPers** script contains three constant declarations that specify the database to which the virtual users connect. Before you can execute the script, you must change these constant values so that the users connect to your database.

Personalizing the test

Procedure To personalize the sample load-testing script:

1. In the **Active Script** window, select **Symbols/Constants/USERNAME**.

   You see the following code:

```
const
  USERNAME       := "user";     // specify YOUR user name here
  PASSWORD       := "password"; // specify YOUR password here
  CONNECTSTRING  := "orclnet2"; // specify the Oracle connect string
```

2. Replace the “user” string with the user name you wish to use to connect to the database.

3. Replace the “password” string with the password you specified for the user.

4 Replace the “orclnet2” string with the connection string that refers to your database.

5 Select **File/Save** from the menu bar to save your changes.

Executing the test script

Three user groups are set up in the OraLoadPers test script:

- **Creator**: A virtual user of this group creates the tables that the PersonPB application requires.
- **Loader**: A virtual user of this group loads sample data into the database so that the PersonPB application can be load tested under real-world conditions.
- **Dropper**: A virtual user of this group removes the sample data and deletes the tables that were created earlier for PersonPB.

To create the tables and insert the sample data for the remaining tutorials, you must first run a virtual user of the Creator group and then a user of the Loader group.

Executing the test script

Procedure To execute the OraLoadPers test script:

1 In the **Active Script** window, expand the **User Groups** folder.

2 In the **User Groups** folder, right-click **Creator** and select **Run User “Creator.”**

Silk Performer runs a virtual user, performing all the actions defined in the test script (i.e., creating the tables in the database that the PersonPB application requires). While the virtual user runs, you can watch its progress in the **Monitor** window (e.g., number of transactions executed, response time of the last transaction, and average response time).

3 Monitor the load test and wait until the virtual user has finished executing its tasks.

a In the top part of the **Monitor** window, select an agent computer or a user group that you wish to monitor.

In the bottom part of the **Monitor** window, Silk Performer displays overview information about the virtual users running on the selected agent computer, or belonging to the selected user group.

b In the bottom part of the **Monitor** window, you can right-click a virtual user to view detailed information and select **Show Output**.

Silk Performer then displays detailed run-time information about the selected user in the **Virtual User** window at the bottom (e.g., transactions and functions the user executes, and the data the user sends to and receives from the server). The type of information displayed depends on the options selected at the top of the Monitor window.

Display Errors. Select this button to display appropriate error messages indicating the probable causes of user-related errors.

Display Transactions. Select this button to display a message when a user has finished executing a transaction; messages indicate whether or not transactions were successful.

Display Functions. Select this button to display a message for each function call a given user performs, including function name and its parameters.

Display Info. Select this button to display messages containing additional information about the current action a given user performs.

Display Data. Select this button to show data exchanged with the server.

Display all Errors of all Users. Select this button to display error messages for all errors of all users. Each error message indicates user, agent and probable cause.

4 From the menu bar, select **File/Close** to close the **Monitor** window.

5 In the **User Groups** folder, right-click **Loader** and select **Run User "Loader."**

Silk Performer runs a virtual user that inserts sample data into the previously created tables.

What you have learned

In Tutorial 1, you learned how to:

- Create a load-testing project
- Add a predefined script to a project
- Execute a script with one virtual user
- View progress information for a load test

Tutorial 2: Creating a script with the Recorder

The Silk Performer Recorder allows you to capture database traffic transferred between a client application and a server. After you record database traffic, you can save it as a test script. You can then easily customize the script and replay it to simulate a large number of virtual users. Recording, customizing, and replaying scripts is covered in depth in Tutorials 2, 3, and 4. In Tutorial 5, you will analyze the results of a load test.

This tutorial includes the following steps:

- Setting up an application profile
- Recording traffic between a client application and a server
- Generating a test script based on recorded traffic
- Trying out a generated test script
- Validating a generated test script

Setting up an application profile

When you want the Silk Performer Recorder to capture traffic exchanged between a client application and a server, you must set up a profile for the client application. Profiles specify client application type and what traffic is to be recorded.

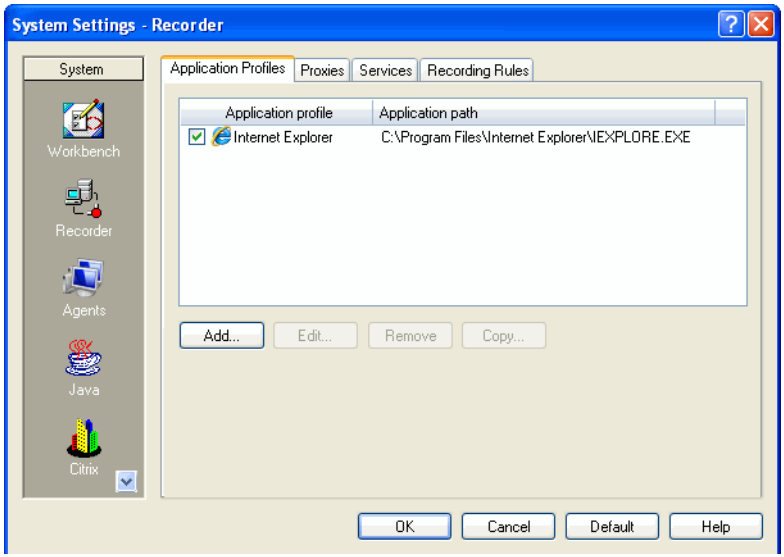
Setting up an application profile

Procedure To set up a profile for the client application:

1 From the menu bar, select **Settings/System**.

The **System Settings – Workbench** dialog box opens.

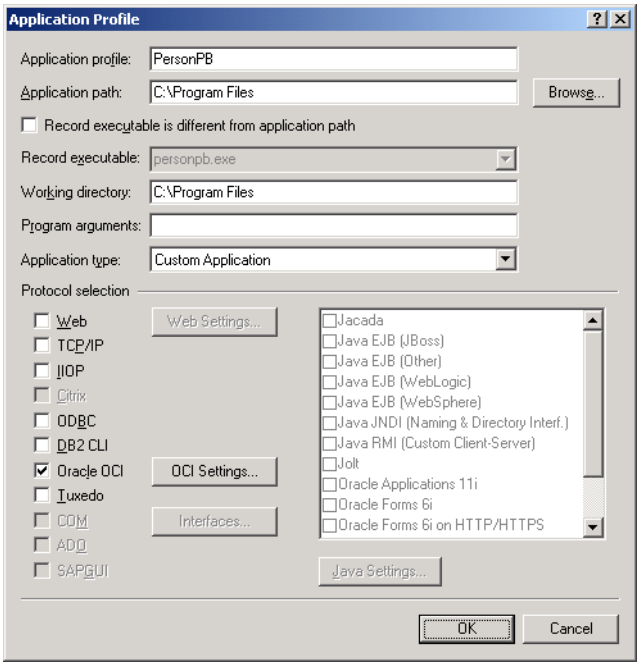
2 In the shortcut list on the left, click the **Recorder** icon.



The list contains all the application profiles that are currently set up on your computer.

3 Click **Add**.

The **Application Profile** dialog box opens.



4. In the **Application profile** text box, enter a name for the application profile, for example **PersonPB**.
5. Click **Browse** and locate the executable called **PersonPB.exe**. This file is located in the SampleApps\Oracle subfolder of the Silk Performer home directory.
6. From the **Application type** list box, select the **Custom Application** option.
7. In the **API selection** area, select the **Oracle** option.
8. Click **OCI Settings...**. The **Oracle OCI Settings** dialog box opens.
9. From the list box select **Ociw32.dll**. Click **OK**.
10. Click **OK**.

    Silk Performer adds the new application profile to the profile list in the **System Settings – Recorder** dialog box.
11. Click **OK** to close the dialog box.

Modeling a test script

To load test a database, traffic between the client application and a server must be recorded and described in a test script. The script enables any number of virtual users to perform actions similar to those that were performed during the recording session.

To create as realistic a load test as possible, you will record five separate transactions in a single recording session:

- Logging on to the database
- Searching for customer information
- Updating a customer record
- Inserting a new customer record
- Logging off from the database

When you run the load test, you will create three user groups to which you will assign each of the above tasks in a ratio that simulates real-world behavior.

Modeling a test script

Procedure To model a load-testing script:

1. On the **Workflow** bar, click **Model Script**.

   The **Workflow – Model Script** dialog box opens.

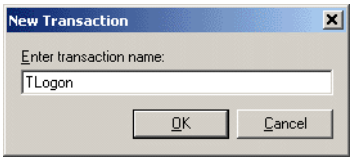
2. From the **Application Profile** list box, select **PersonPB**.

3. Click **Start recording**.

   Silk Performer opens the Recorder and starts the PersonPB application.

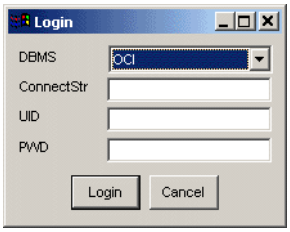
4. In the Recorder, click the **New Transaction** button to insert a new transaction into the test script you are generating.
   Confirm the next Silk Performer Recorder message by clicking **Yes.**

   The following dialog box opens.



5. Create a new transaction called **TLogon**; then click **OK**. A distinct set of time measurements will be made for each transaction you create. When recording traffic, you should create a new transaction for each distinct user session, from connection to shutdown.

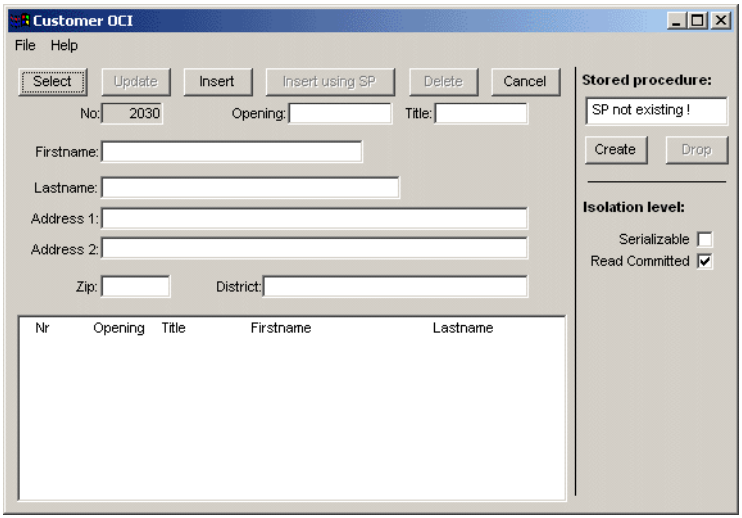
6 In the **Login** dialog box of the PersonPB application, select **OCI** from the **DBMS** list box.



7 Enter the database connection string, user name, and password for logging on to the database.

8 Click **Login**.

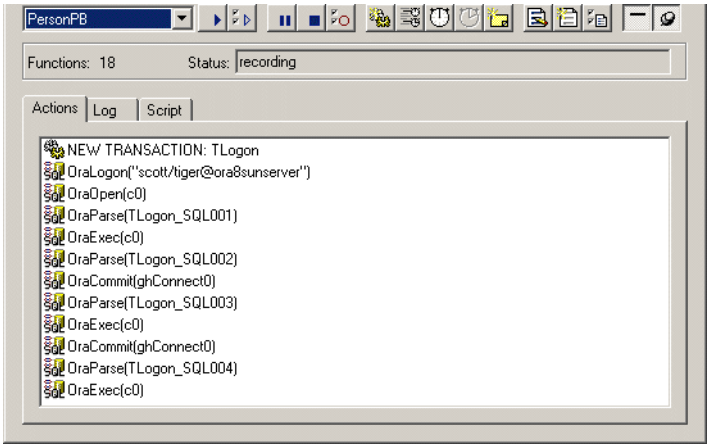
The **Customer** window opens.



The Recorder records the database traffic automatically as you connect to the database.

9 To view your actions and the function calls that the PersonPB application performs, click the **Change GUI Size** button.

The **Silk Performer Recorder** window displays as follows.



10 In the Recorder, create a new transaction called **TSearch**.

11 In the **Customer OCI** dialog box, enter a last name for which you wish to search, for example **Williams**, and press **Select**.

12 In the PersonPB application, perform the following additional steps. Before you perform each step, create a new transaction with a name that describes the user session, for example **TUpdate** or **TInsert**.

a Change the customer address

b Insert a new customer record

Note For detailed information on using the PersonPB sample application, refer to the *Silk Performer PersonPB Reference.*

13 On the Recorder, create a new transaction called **TLogoff**.

14 In the **Customer OCI** window, select **File/Exit** from the menu bar.

15 On the Recorder, click the **Stop Recording** button.

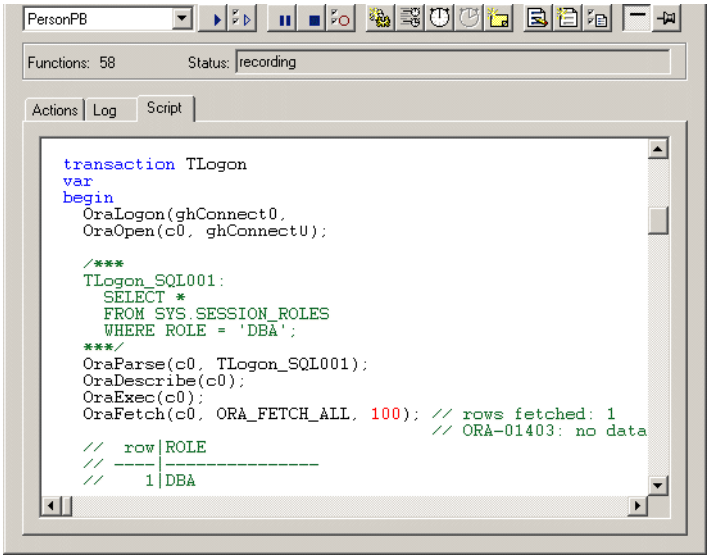
The **Save As** dialog box opens.

16 In the **File name** text box, enter **PersonPB**.

17 Click **Save**.

18 You can now decide whether or not to close the Recorder.

If you keep the Recorder open, you can examine the script that was generated.



19 Close the Recorder.

Silk Performer then automatically adds the script to your current load-testing project.

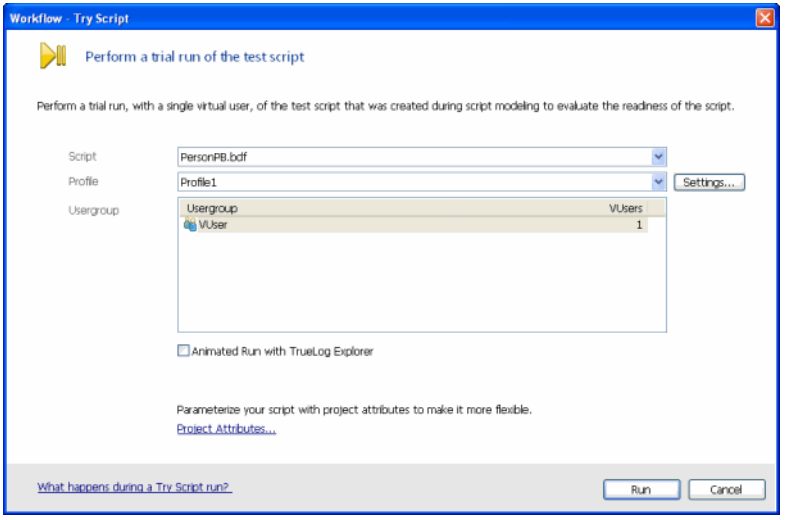
Trying out the generated test script

Next you should do a trial run of the test script that was created. The object of the trial run is to ensure that the script is error free, and that it accurately reproduces the interaction between the client application and the database server.

Trying out the script

Procedure To try out the generated test script:

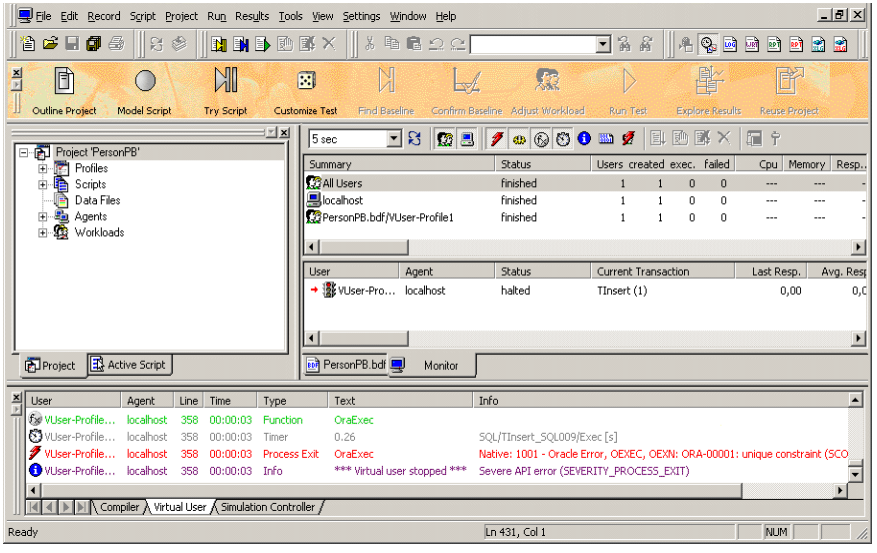
1 In the **Workflow** bar, click the **Try Script** button.



2 From the **Script** list box, select the **PersonPB.bdf** script.

3 Click **Run**.

Silk Performer runs a single virtual user, called VUser, which performs all the actions that were performed during the recording session. While the single-user test runs, you can watch its progress.



The **Try Script** run failed because of an error. The "… **unique constraint** … **violation** …" error message in the **Virtual User** output tab indicates that the script attempted to insert a record with a primary key that was not unique within the data table.

Your script attempted to use the same value that was used during recording. This is a typical problem when replaying scripts that include session dependent, dynamic information. Tutorial 3 shows you how to customize your script to handle such session dependent, dynamic information.

Specifically the script attempted to add a new person record that didn't have a unique person ID. Person ID's are numbers from 1 to 9999. The highest used person ID value is stored by the PersonPB sample application in a separate data table. Before adding a new record, PersonPB retrieves the value of the highest used person ID, increments the number up by one, and updates the data table. Then it retrieves the new value and uses it as an input parameter for the insert statement.

What you have learned

In Tutorial 2, you learned how to:

- Set up an application profile
- Record traffic between a client application and a database
- Generate a test script based on recorded traffic
- Try out a generated test script

Tutorial 3: Customizing a generated test script

Customizing a generated test script allows you to use the actions performed by a single user to realistically simulate the actions of multiple users.

Three key tasks assist in creating real-world simulations:

- Handling session dependent, dynamic information
- Replacing constant values with random variables
- Specifying the tasks each user is to perform

Note Customizing a test script can be done through either Silk Performer Workbench or Silk TrueLog Explorer (TrueLog Explorer). When using Silk Performer Workbench to customize generated test scripts you must use Benchmark Description Language (BDL), Silk Performer's high-level scripting language. Although knowledge of BDL is not required for this tutorial, it is recommended that you read the BDL introductory topics in Silk Performer's online help system.

No knowledge of BDL is required to use TrueLog Explorer. TrueLog Explorer performs all script manipulation for you. In this tutorial TrueLog Explorer is used to handle session relevant information. To learn more about the functionality and use of TrueLog Explorer, please refer to the *TrueLog Explorer Help*.

Handling session dependent, dynamic information

Script customization must be performed whenever session dependent, dynamic information is used by a database application. In most cases customization involves parsing the result value of a database operation into a variable and substituting the input parameter value of another database operation with the parsed value.

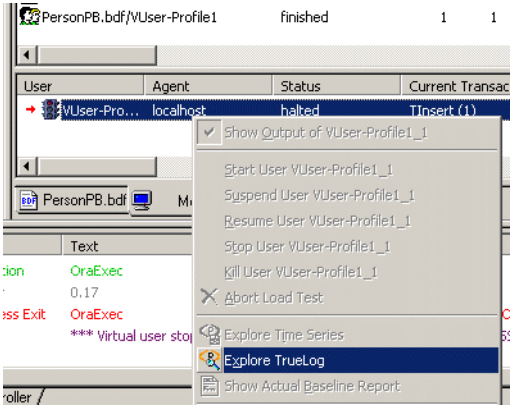
"Tutorial 2: Creating a script with the Recorder" explained the problem of not using a unique primary key within a database table and generating an "unique constraint violation" error.

To handle this situation the result of a database operation must be parsed into a variable. The variable's value must then be used in place of the recorded value as an input parameter for a subsequent database call.

Parsing a result into a variable

Procedure To parse the result of a database operation into a variable:

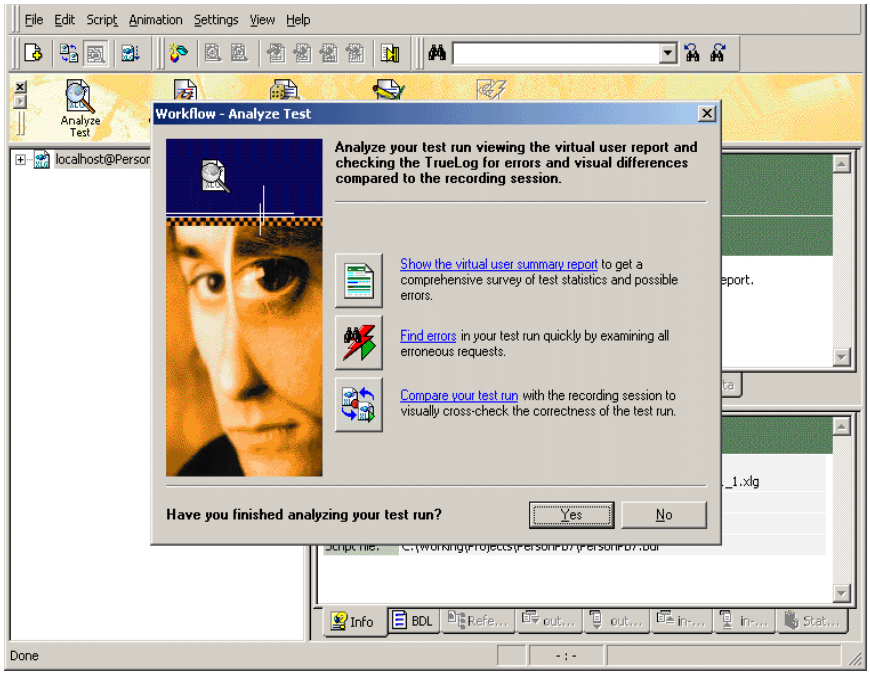
1 After performing a **Try Script** run, right click the VUser Try Script status line in the **Monitor** window and choose **Explore TrueLog**.



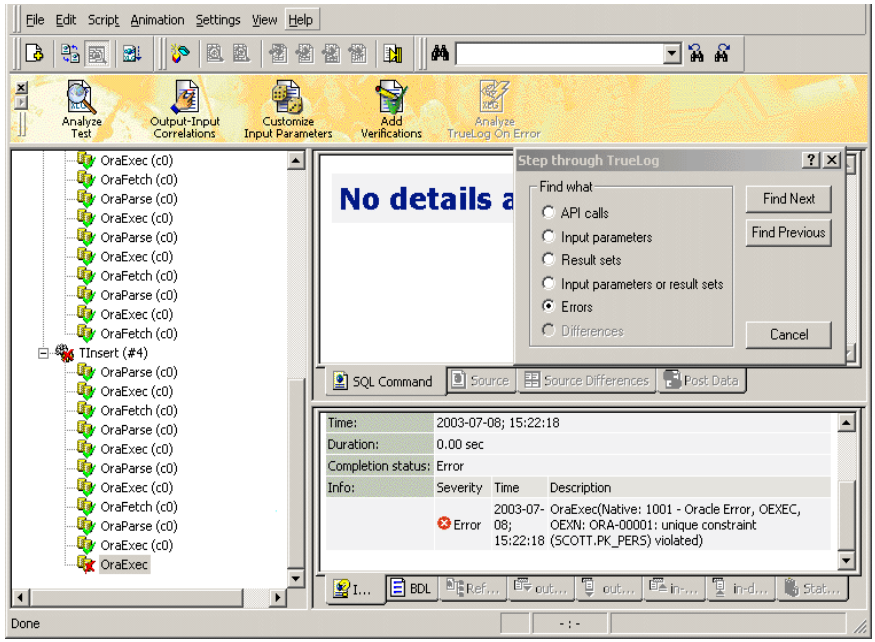
TrueLog Explorer launches, loaded with the replay TrueLog of your **Try Script** run.

2 Click **Analyze Test** on the **TrueLog Explorer Workflow** bar.

The **Workflow - Analyze Test** dialog box opens.

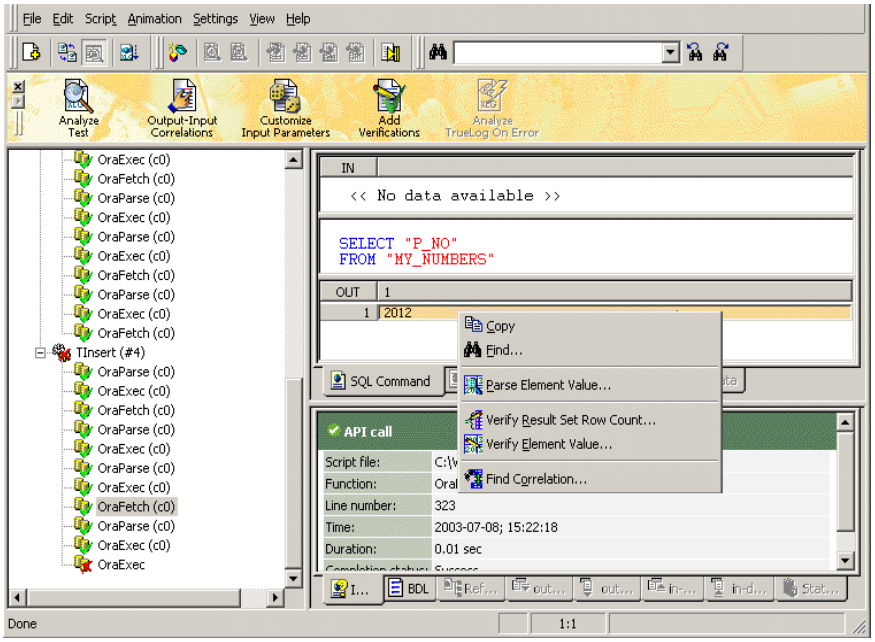


3 Click **Find Errors**.
Click the **Errors** option button on the **Step Through TrueLog** dialog box. Continue clicking **Find Next** until you receive a **No more Errors** message.

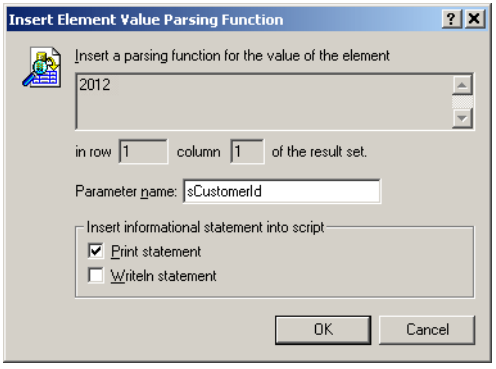


The database operation that caused your **Try Script** run to fail is now selected. On the **Info** tab you can see an error message that describes the reason for the failure.

4 Now select the closest preceding Fetch statement. Select the value in the output parameter table, right-click, and choose **Parse Element Value**.



5 The **Insert Element Value Parsing Function** dialog box displays.



Specify the name of the variable you want the value to be parsed into in the **Parameter name** text box, for example **sCustomerId**, and click **OK**.

Confirm the following message box by clicking **OK**. Click the **BDL** tab in the lower section of TrueLog Explorer. TrueLog Explorer then adds the following lines to your script.

```
sCustomerId := RsGetString("1", 1);
Print("sCustomerId: " + sCustomerId);
```

Additionally TrueLog Explorer declares a global variable in the *dclparam* section of your script.

```
dclparam
  sCustomerId       : string;
```

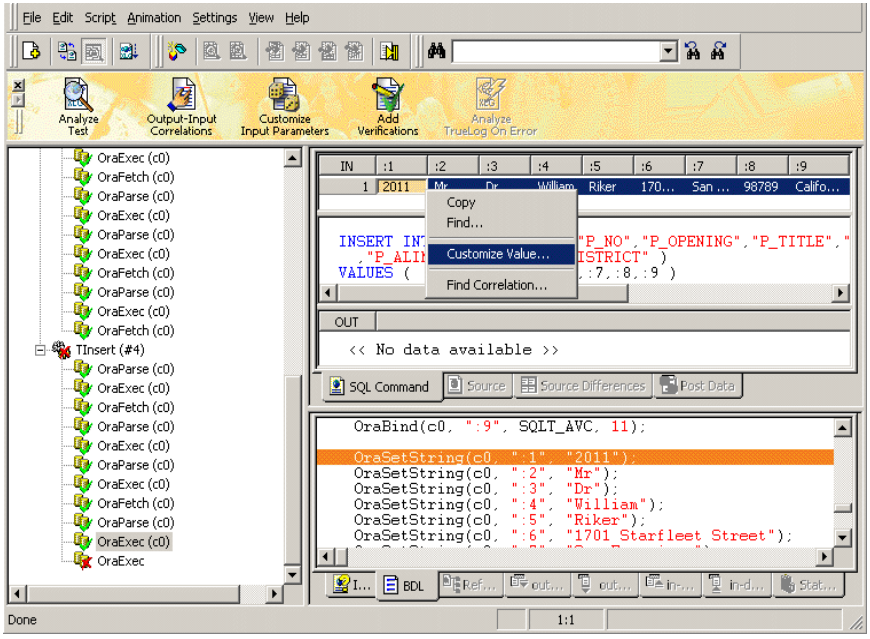
Substitution of an input parameter

Procedure To substitute an input parameter with a variable:

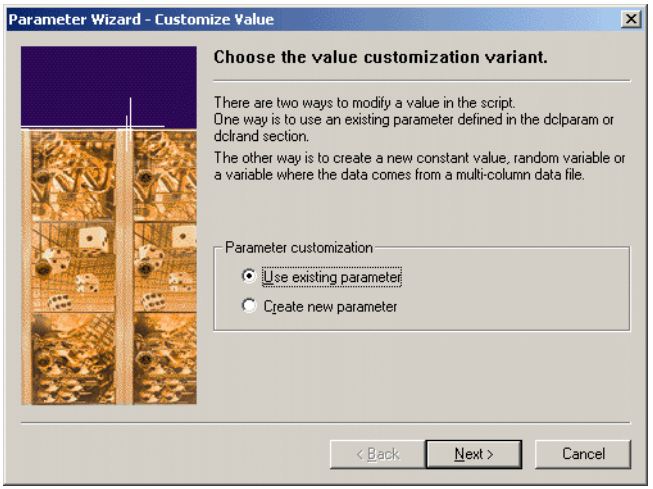
1 After parsing the session relevant data into a variable you can use the value of the variable rather than the formerly passed parameter value as input for a database operation.

In this case you must substitute the parameter value passed to the database operation that caused your **Try Script** run to fail.

To do this, select the database operation node immediately preceding the node that failed, right click the left-most element in the input parameter view and choose **Customize Value**.

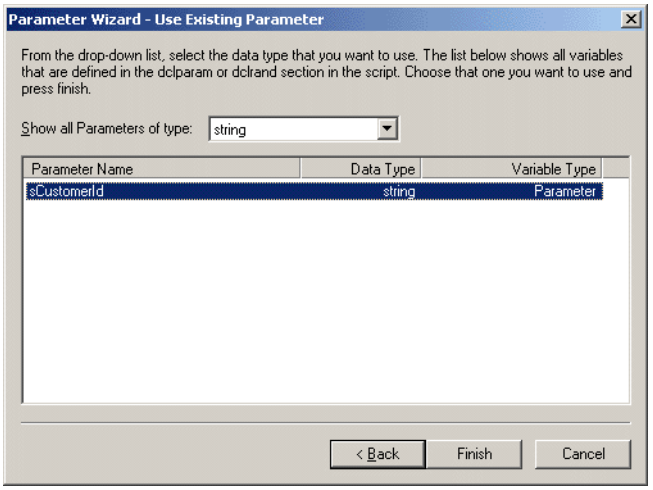


2 The **Parameter Wizard - Customize Value** dialog box opens.



Select **Use existing parameter** and click **Next** to proceed.

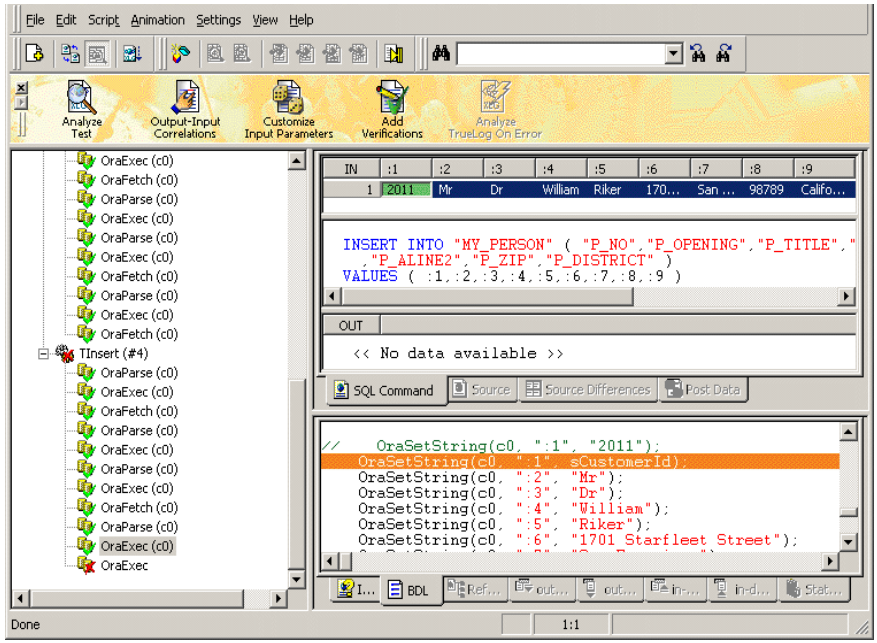
3 The **Parameter Wizard - Use Existing Parameter** dialog box opens.



Select the name of the variable for which you previously created the parsing function, for example **sCustomerId**, and click **Finish**.

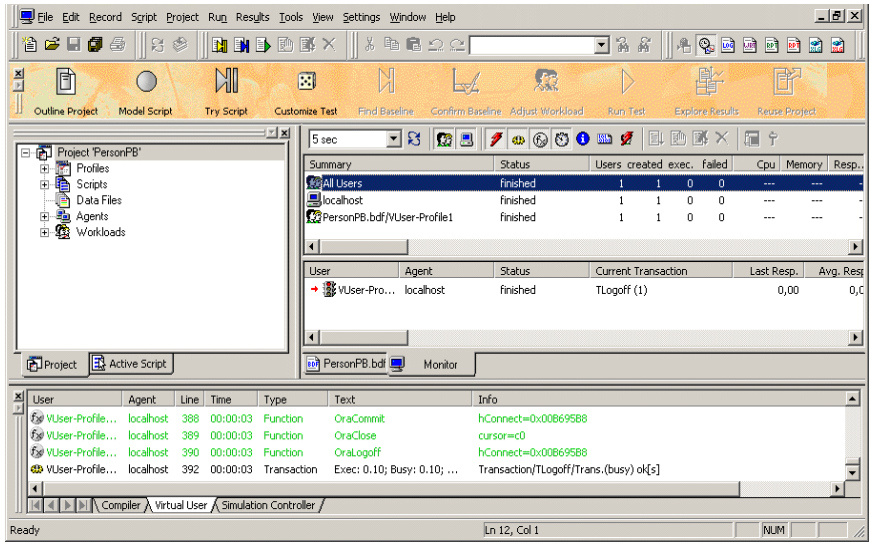
Confirm the following message box by clicking **OK**, and click the BDL tab in the lower section of TrueLog Explorer. TrueLog Explorer then adds the following lines to your script.

```
//    OraSetString(c0, ":1", "2011");
    OraSetString(c0, ":1", sCustomerId);
```



Handling of session dependent, dynamic information is now complete.

Close **TrueLog Explorer**, activate **Silk Performer Workbench**, and do a trial run of the customized script as was shown in Tutorial 2. The **Try Script** run should now proceed without error.



Replacing constant values with random variables

When randomized data is required, random variables are employed to provide realistic user data. In this way, Silk Performer furnishes virtual users with varied personal data, for example name, address, and phone number.

Replacing constant values

Procedure To replace constant values with random variables:

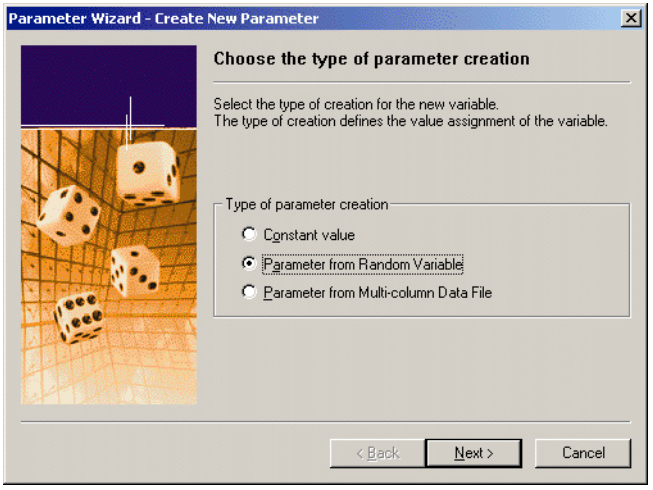
Note Ensure that the PersonPB.bdf script is active in Silk Performer Workbench. To do this, click the PersonPB.bdf tab.

1 In the **Workflow** bar, click the **Customize Test** button.

The **Workflow – Customize Test** dialog box opens.

2 Click the **Create New Parameter** icon.

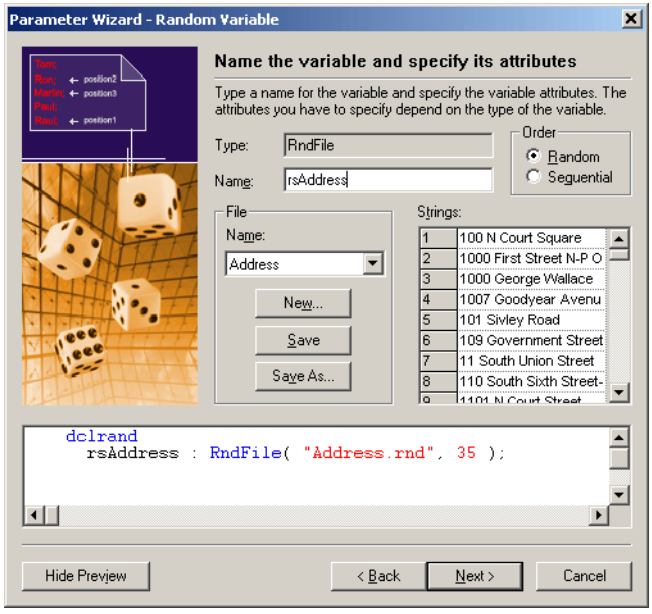
The **Parameter Wizard - Create New Parameter** dialog box opens.



3 Select **Parameter from Random Variable** and click **Next**.
The **Parameter Wizard - Random Variable** dialog box opens.

4 From the **Random type** list box, select the **String from file** type.

5 Click **Next**.



6 Enter a name for your variable in the Name edit field, for example **rsAddress**.
In the **File** area, select **Address** from the **Name** list box.

This random variable selects an address of up to 35 characters in length from a Silk Performer file that contains random addresses.

7 Click **Finish**.

Silk Performer inserts the following lines into your test script:

```
dclrand
  rsAddress : RndFile("Address.rnd", 35);
```

8 Locate the TUpdate transaction.

Do this by expanding the **Transactions** folder in the **Active Script** window and double-clicking **TUpdate**.

9 In the TUpdate transaction, locate the section where the UPDATE statement is executed.

This section looks like the following:

```
/***
TUpdate_SQL007:
  UPDATE "MY_PERSON"
  SET "P_ALINE1" = :1
  WHERE "P_NO" = :2 ;
***/
OraParse(c0, TUpdate_SQL007);

OraBind(c0, ":1", SQLT_AVC, 15);
OraBind(c0, ":2", SQLT_STR);

OraSetString(c0, ":1", "First Avenue");
OraSetString(c0, ":2", "204");

Ora8StmtExecute(ghSvcCtx0, ghStmt0, 1);
```

10 Replace the fourth parameter of the first OraBind function call with **40**.

11 Replace the address that you entered earlier, “First Avenue” in this case, with the **rsAddress** variable.

The code section should now look like the following. The code shown in bold type is what was changed:

```
OraParse(c0, TUpdate_SQL007);

OraBind(c0, ":1", SQLT_AVC, 40);
OraBind(c0, ":2", SQLT_STR);

OraSetString(c0, ":1", rsAddress);
OraSetString(c0, ":2", "204");

OraExec(c0);
```

Specifying which tasks each user is to perform

Usually a load test simulates a number of different user types. In the test script, you must specify which action each virtual user type is to perform.

Assigning tasks to users

Procedure To specify which tasks each user is to perform:

1. Navigate to the **dcluser** section of the PersonPB.bdf script file. Do this by selecting **User Groups/VUser** in the **Active Script** window.

   You will see the following:

```
dcluser
  user
    VUser
  transactions
    TInit           : begin;
    TLogon          : 1;
    TSearch         : 1;
    TUpdate         : 1;
    TInsert         : 1;
    TLogoff         : 1;
```

   In this section, a single user group called **VUser** is defined. By default, this user group performs each of the transactions you created (TLogon, TSearch, TUpdate, TInsert, and TLogoff) once.

2. Edit the **dcluser** section of the script so that it looks like the following:

```
dcluser
  user
    Searcher
  transactions
    TLogon          : begin;
    TSearch         : 5;
    TLogoff         : end;

  user
    Updater
  transactions
    TLogon          : begin;
    TSearch         : 3;
    TUpdate         : 2;
    TLogoff         : end;

  user
    Inserter
  transactions
    TLogon          : begin;
    TSearch         : 1;
    TInsert         : 3;
    TLogoff         : end;
```

   In the edited version of the script, the **Searcher** user group performs the TLogon transaction at the beginning, then the TSearch transaction five times, and finally the TLogoff transaction at the end. The **Updater** user group performs the TLogon transaction at the beginning, then the TSearch transaction three times, the TUpdate transaction twice, and the TLogoff transaction at the end. The **Inserter** user group performs the TLogon transaction at the beginning, the TSearch transaction once, the TInsert transaction three times, and the TLogoff transaction at the end.

3. Select **File/Save** from the menu bar to save your changes.

4. On the **Workflow** toolbar, click the **Customize Test** button.

The **Workflow – Customize Test** dialog box opens.

5 Click **OK** to confirm that you're done customizing the test script.

What you have learned

In Tutorial 3 you learned how to:

- Replace constant values with random variables
- Specify which tasks each virtual user is to perform

Tutorial 4: Replaying a customized script

Once you have created a test script by recording traffic, and then customized the script, you can load test your server by replaying the script. To run a load test you must perform the following steps:

- Find the test baseline
- Confirm the test baseline
- Set up server monitoring for your load test
- Execute the load test

Finding the test baseline

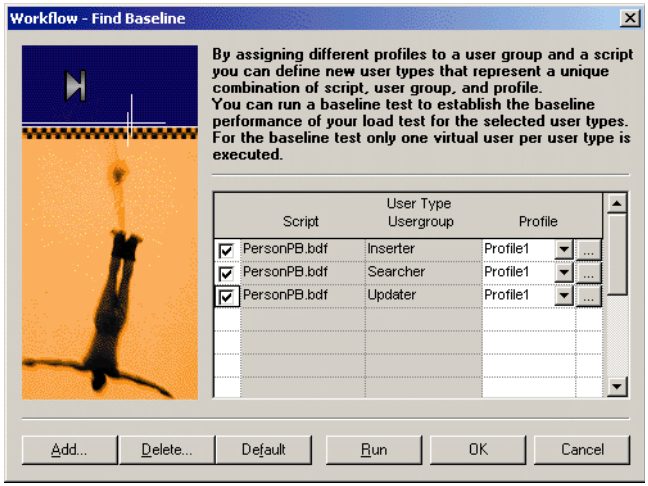
Once you have customized the load-testing script, you must determine baseline performance. This is the ideal performance of your application when it is not under significant pressure. To identify the test baseline, the fully customized script is run with just one user per user group. This will provide a solid basis for comparison when the application is later placed under stress during the load test.

Finding the baseline

Procedure To find the load test baseline:

1 On the **Workflow** toolbar, click the **Find Baseline** button.

The **Workflow – Find Baseline** dialog box opens.



This dialog lists all the user groups that have been set up in your test script.

2 Click **Run**.

Silk Performer runs one virtual user from each user group declared in the test script. While the test runs you can watch its progress in the **Monitor** window.

Confirming the test baseline

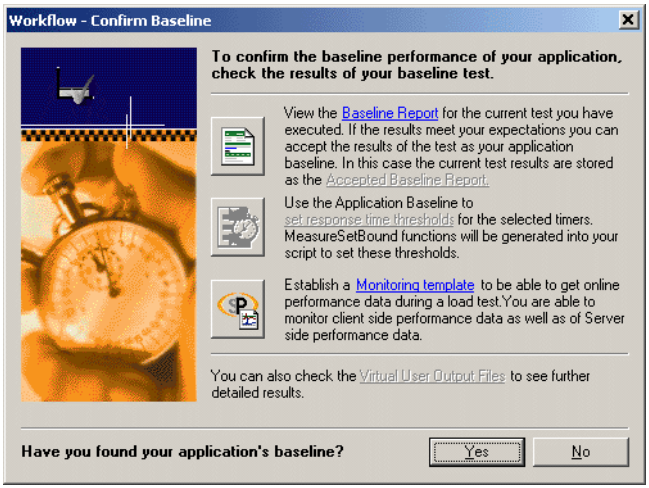
After running the baseline test, you must confirm that the test result reflects the desired performance of the tested server. To do this, inspect the results of the test. If they are satisfactory, the baseline has been identified and can later form the basis for comparison with results from full load tests.

Confirming the baseline

Procedure To confirm the load test baseline:

1 On the **Workflow** toolbar, click the **Confirm Baseline** button.

The **Workflow – Confirm Baseline** dialog box opens.



2 Click the **Baseline Report** button.

The Baseline Report opens.

3 Check the report carefully. In particular, make sure that no errors occurred during the baseline test.

4 Click the **Accept Baseline** button on the Baseline Report.

5 Click **Yes** and **OK** on the following dialog boxes.

6 If your baseline report doesn't indicate any errors, click **Yes** on the **Workflow – Confirm Baseline** dialog box to confirm that you have identified the baseline.

Adjusting the workload

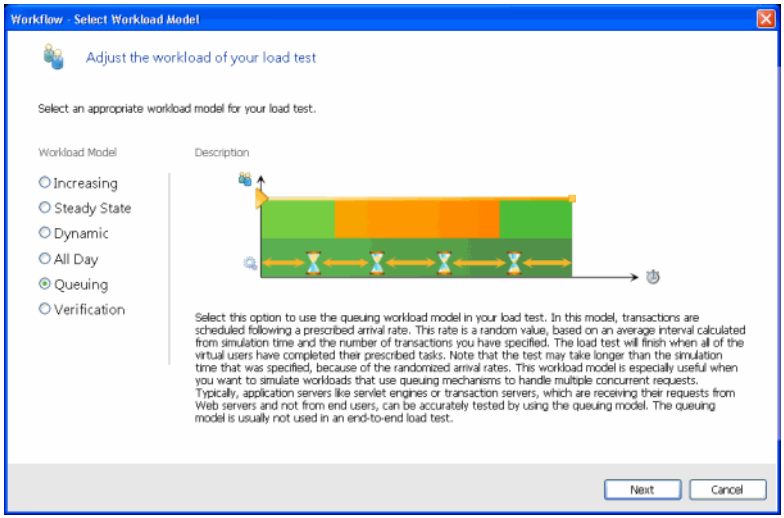
Once you have confirmed your baseline you can select a workload model and prepare the workload to run a full load test.

Adjusting the workload

Procedure To adjust the workload:

1 On the **Workflow** toolbar, click the **Adjust Workload** button.

The **Workflow - Select Workload Model** dialog box opens.



This dialog box shows all possible workload models that can be selected.

2 Select the **Queuing** workload model.

3 Click **Next**.

4 On the **Workflow Assign Agents** dialog box, select an option and click **OK**.

Silk Performer opens the Workload Configuration dialog box.

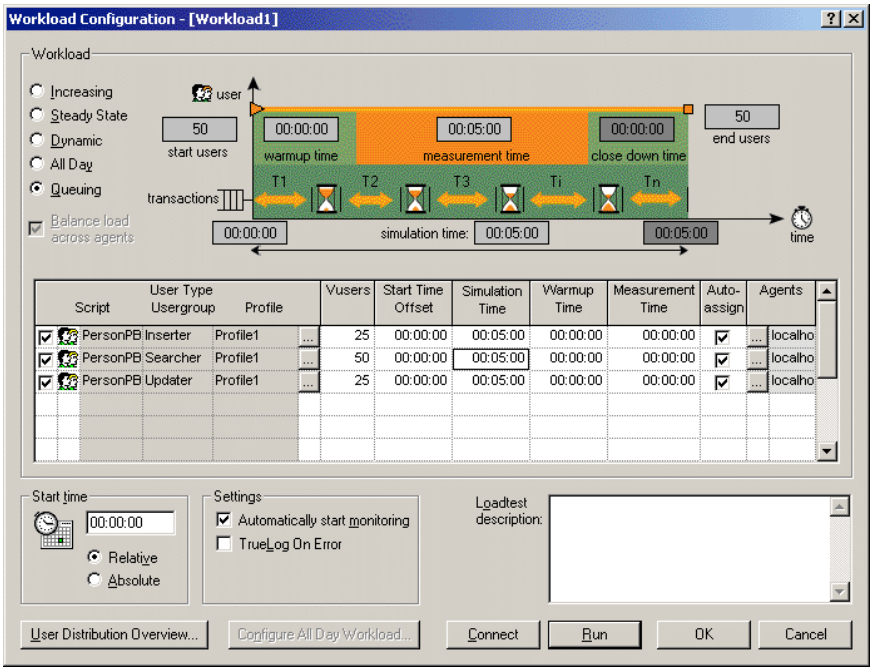
Running the load test

Once you have adjusted the workload, you're ready to run a full load test. The test script is run with multiple virtual users to test the database server. For this tutorial, only your local computer is required. A true load test however requires an appropriate testing environment, which you have to set up on the local area network, that offers a full complement of agent computers to host virtual users.

Running the test

Procedure To run the load test:

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



2 In the **VUsers** column, change the number of Inserter users to **25**, the number of Searcher users to **50**, and the number of Updater users to **25**.

3 For each of these user groups, change the simulation time to **300** seconds (5 minutes).

You are now ready to run the load test. When you run a test Silk Performer's sophisticated monitoring tool, **Performance Explorer**, automatically generates a live graphical display with a default set of test data for the server under test. You can disable this default option by deselecting the setting **Automatically start monitoring**. You can also change the default monitoring settings by pressing the **Active profile** button. For additional information, see Customizing server monitoring. Once monitoring has been set up as required, you can run the test.

4 Disable all user groups that are declared in the OraLoadPers test script.

5 Click **Run** to start the test, or click **OK** to save the settings and run the test later.

When you start the test, Silk Performer runs the load test against the database server with 100 virtual users.

If the default **Automatically start monitoring** option is enabled, **Performance Explorer** will open and generate a live graphical display of the specified test data. You can adapt these views by adding and

deleting specific measurements. To do this, expand the tree menu on the left and drag and drop selected measurements into existing or new view graphs.

Customizing server monitoring for your load test

An important feature provided by Silk Performer is server monitoring during actual load tests. Server monitoring helps you locate and analyze bottlenecks on servers. You can separately examine the performance of the operating system, the network and the server application (which in this case is the Oracle server).

Server monitoring is performed using Performance Explorer, a powerful tool incorporated into Silk Performer. When you run a test, the default procedure has Performance Explorer automatically open a set of views containing a live graphical display of a selection of test data that is relevant to the server under test. You can disable this default option by deselecting the **Automatically start monitoring** setting on the **Workload Configuration** dialog. You can also customize the default monitoring settings and select a particular set of data sources.

Customizing server monitoring

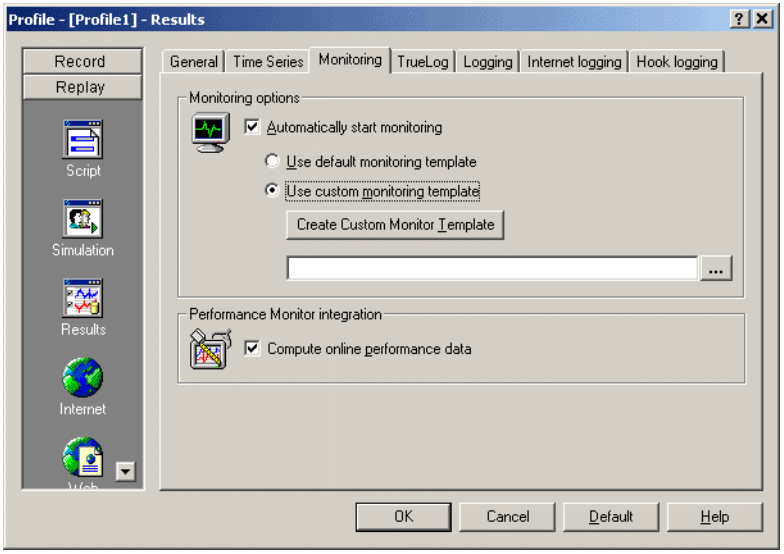
Procedure To customize server monitoring for your load test:

1. On the **Workflow** toolbar, click the **Run Test** button.

   The **Workload Configuration** dialog box opens.

2. In the **Settings** section, click the **Active Profile** button.

   The **Profile - [Profile1] - Results** dialog box opens.



3 In the shortcut list on the left, select the **Replay** category and click the **Results** icon.

4 Select the **Monitoring** tab

The **Automatically start monitoring** option is activated by default. This means that when you run a load test, Performance Explorer opens automatically using a default template that invokes a display of relevant client information.

5 If you want to customize monitoring, enable the **Use custom monitoring template**, and click the **Create Custom Monitor Template** button.

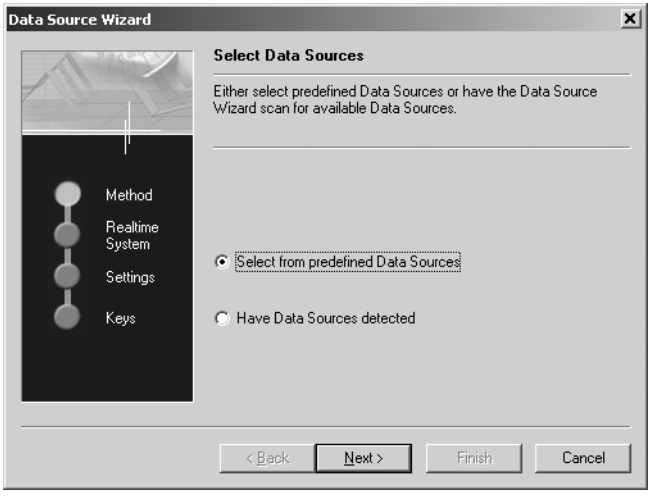
Performance Explorer creates and opens a template with the name of your project, which is the same as the default template.

6 Click **Edit Custom Monitor Template** (Performance Explorer opens)**,** to customize the new template and change the information display. You can close or maintain the default monitoring windows and you can add one or more windows to provide additional information.

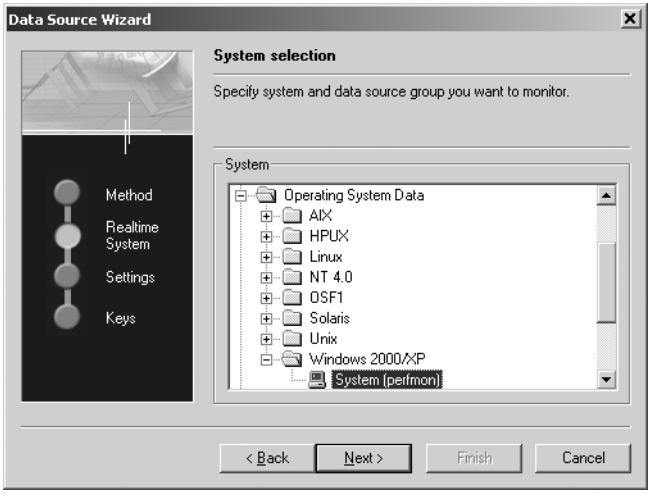
To set up **Performance Explorer** to display additional information, do the following:

a From the menu bar, select **Monitor/Add Data Source**.

The **Data Source Wizard** opens.



b Click **Next**.
The **Data Source Wizard - System selection** dialog box opens.



c In the tree view, expand the folder that corresponds to the operating system on which the database server runs. Select **System**.

d Click **Next**.

e In the **Connection Parameters** dialog box, enter connection parameters such as the host name or IP address of the server, the connection port, the user name, and the password.

The data you enter here depends on the operating system that runs on the computer you are monitoring.

f Click **Next** once you have entered the connection parameters.

g In the next dialog box, select the performance counters you wish to monitor.

Of particular interest are processor and memory utilization of the server under test.

h Click **Finish**.
Performance Explorer opens a new window, displaying real-time data for the performance counters you have selected.

i In the **Monitor** tree view, right-click the name of the computer you are monitoring and select **Add Data Source**.

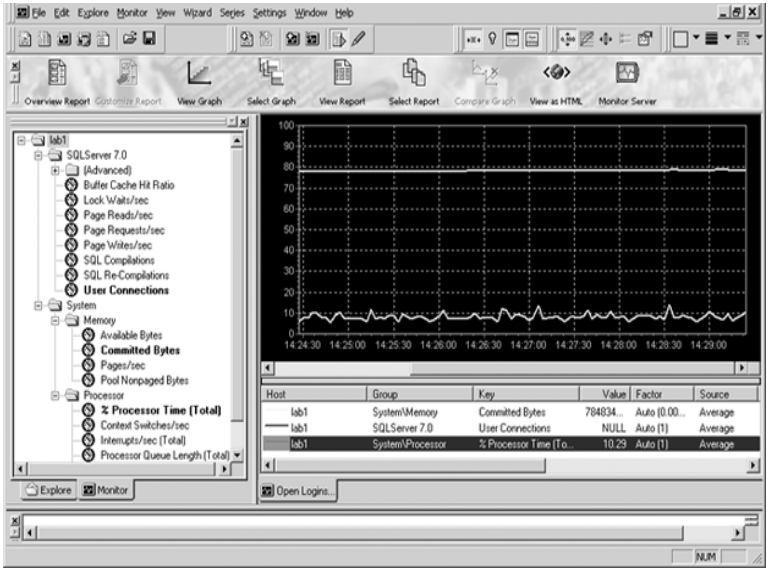
j The Data Source Wizard opens.

k Repeat steps A through E. However this time:

I On the first dialog box, select the name of the Oracle server software rather than **System**.

II On the third dialog box, select which performance counters you wish to monitor.

Later, when you run the load test, the monitor view will look similar to the following:



What you have learned

In Tutorial 4, you learned how to:

- Find a test baseline
- Confirm a test baseline
- Customize monitoring for a load test
- Adjust a workload
- Execute a load test

Tutorial 5: Viewing the results of your load test

Silk Performer incorporates **Performance Explorer**, a powerful graphing and analysis tool, to assist you in working with results information.

Performance Explorer enables you to analyze results information with advanced features for creating statistical reports and displaying performance results in real-time generated graphics. Results information varies depending on the type of application and/or server under test. In general, the following is available:

- **Response time information:** This is the total time it takes to process a given timer; it provides application performance information from the client perspective.
- **Throughput information:** This is the average rate at which a given timer is processed by the server; it allows you to analyze performance from the server perspective.

Overview report of performed measurements

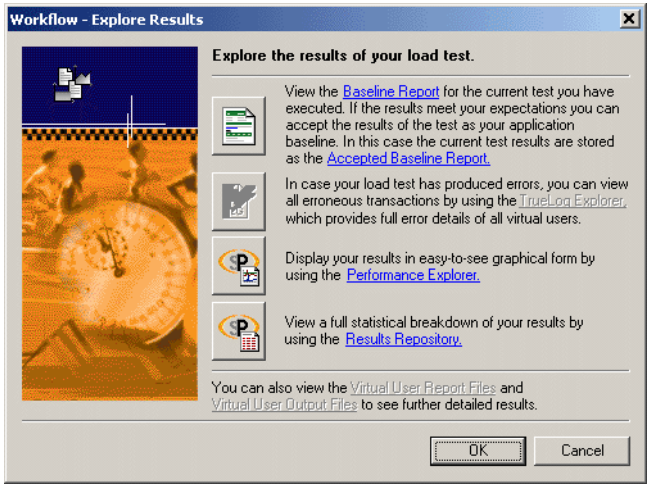
You can consult an HTML overview report that contains results of all performed measurements.

Exploring the overview report

Procedure To explore an overview report:

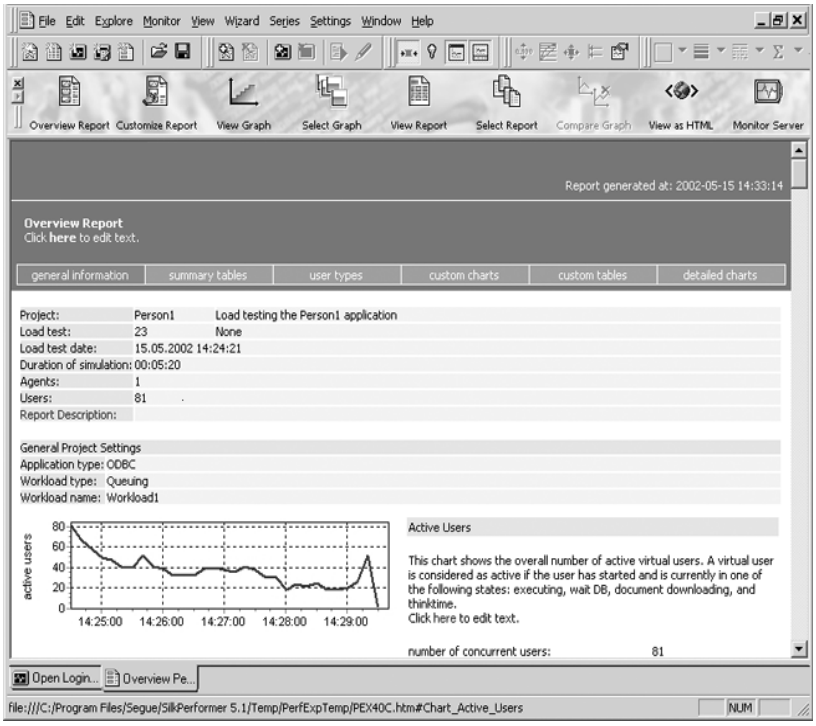
1 On the **Workflow** toolbar, click the **Explore Results** button.

The **Workflow - Explore Results** dialog opens.



2 Click the **Performance Explorer** button.

Performance Explorer opens, and the HTML overview report is generated.



This report offers short explanations of displayed graphs. More detailed information is also available.

Detailed response time measurements

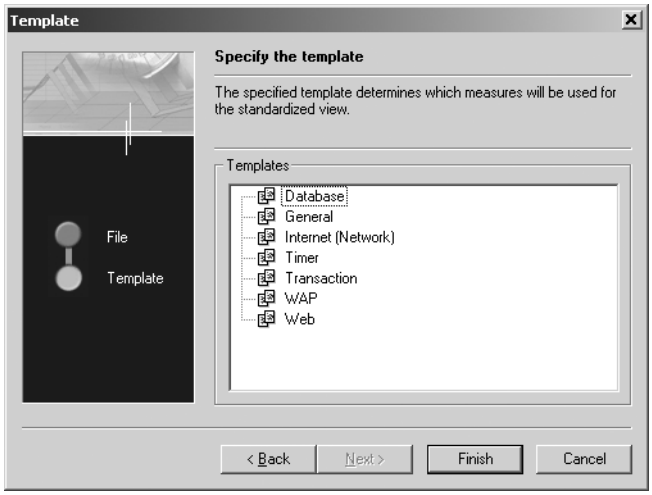
When you analyze the performance of an application from the client point of view, you examine the response times of each transaction and timer. This enables you to find out how users will experience interaction with server objects. A common goal in load testing is to find out if response times for all transactions remain below a specified, critical limit.

Response time details

Procedure To explore detailed response time measurements:

1 On the Performance Explorer **Workflow** bar, click the **Select Graph** button.

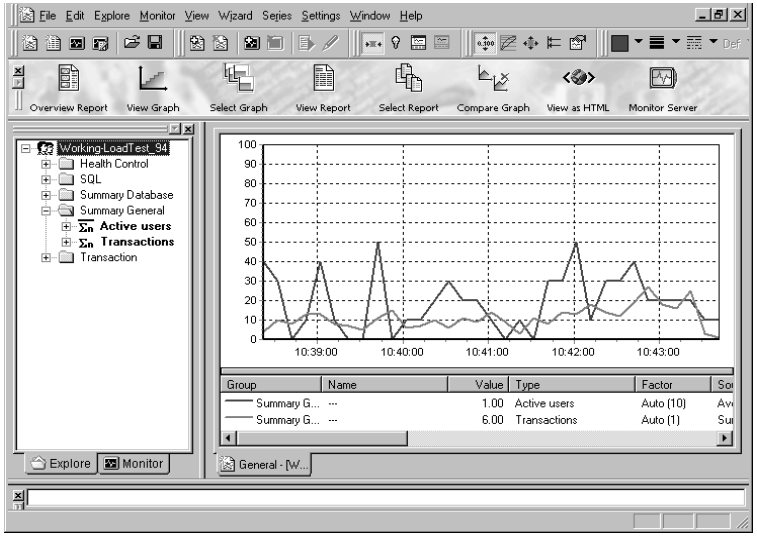
The **Template** dialog opens.



2 In the **Templates** area, select the **Timer** option.

3 Click **Finish**.

Performance Explorer displays a graph of the active users.



You see the default graph with the average number of active users, and one or more default measurements (if any).

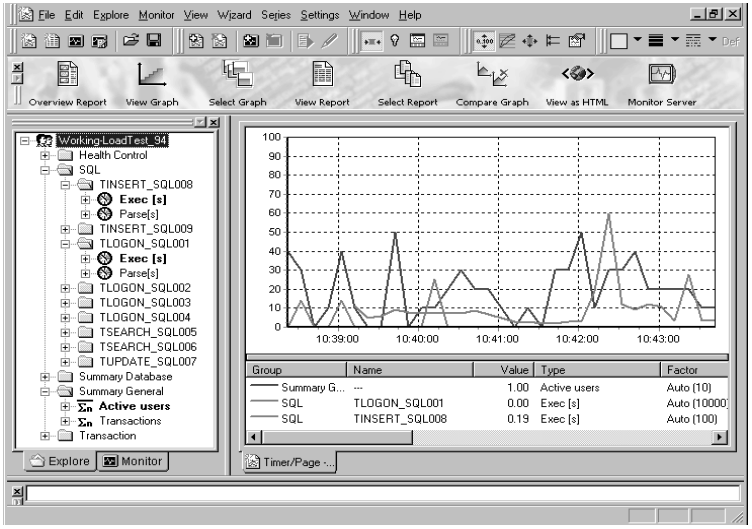
4 Expand the **Explore** tree view.

5 In the **Explore** tree view, select **SQL Commands/TLOGON_SQL001**.

6 Drag the **Exec** measurement into the new view.

7 In the **Explore** tree view, select **SQL Commands/TINSERT_SQL008**.

8 Again, drag the **Exec** measurement into the new view.



The load test you ran in the previous tutorial lasted only five minutes, so there are few measurements, which makes it difficult to draw any conclusions.

Real load tests can last several hours, a few days, or even a week. Such complete tests provide result information that allows you to accurately analyze server performance.

Exploring throughput measurements

When you analyze the performance of a database server from the point of view of the server, you examine the **throughput rate**. Throughput is work done by a server during a specific time interval (e.g., number of requests that a server processes each second, minute, or hour). A typical goal in load testing is to ensure that throughput rate remains above a specified, critical limit.

Throughput rate details

Procedure To explore throughput measurements:

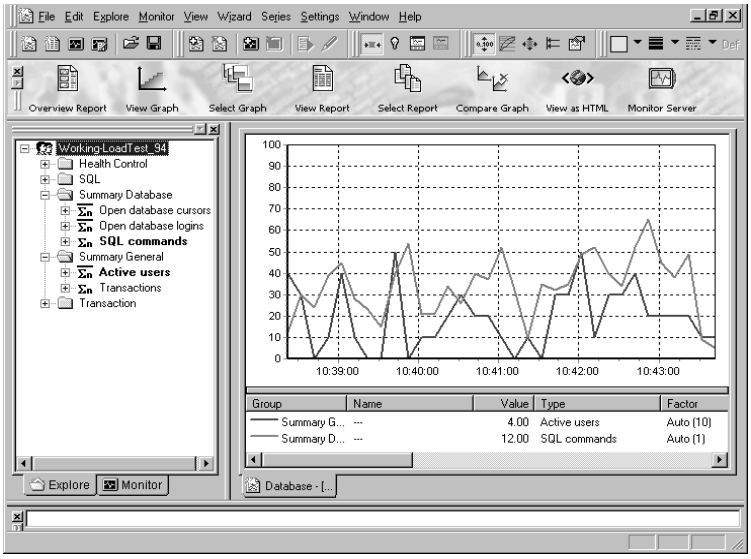
1 On the Performance Explorer **Workflow** bar, click the **Select Graph** button.

The **Template** dialog opens.

2 In the **Templates** area, select the **Database** group.

3 Click **Finish**.

Performance Explorer opens a new chart that resembles the following:



Again, since the load test you ran lasted only five minutes, it's difficult to draw conclusions from the results.

What you have learned

In Tutorial 5, you learned how to:

- Explore response time measurements
- Explore throughput measurements

4

Load Testing an Oracle OCI8 Database

Introduction

This chapter contains a series of tutorials that will get you started using Silk Performer to load test Oracle OCI8 databases.

What you will learn

This chapter contains the following sections:

Overview

Each of the following tutorials provides step-by-step instruction for important Silk Performer tasks. These tutorials should be followed in order, as each tutorial builds on what is covered in the preceding tutorials.

You will learn how to:

- Execute a predefined load test on your local machine.
- Create a test script by recording database traffic using the Recorder.
- Customize a recorded test script.
- Run a customized script, using multiple virtual users.

- Use reporting tools to view select throughput and response time information.

Sample: PersonPB7

These tutorials use a sample database application called PersonPB7. This application is included on the Silk Performer CD-ROM and is installed in the SampleApps\Oracle subfolder of the Silk Performer home directory.

Oracle client

To perform this tutorial **Oracle Client** software must be installed on your computer.

Tutorial 1: Running a predefined test

In this tutorial you will learn how to create a load-testing project and execute sample scripts provided with Silk Performer. You will run a sample script with a single virtual user on your local machine.

You will use the test script OraLoadPers.bdf, which is available at **<Public User Documents>\Silk Performer 16.5\Samples\Database\Oracle**. This script creates the tables that the PersonPB7 application requires and inserts data into your database that is required for the remaining tutorials.

Creating a load-testing project

For each server you load test with Silk Performer, you must create a project file. A project file administers all the settings that are associated with a series of load tests, including test type, workload model, and options that depend on the server type under test. In addition, project files enable you to easily switch between different load-testing projects without having to define new settings each time.

Creating a project

Procedure To create your load-testing project:

1. Open Silk Performer.
2. Close any open projects (Silk Performer opens with the last project you worked on).
3. On the **Workflow** bar, click **Start here**.

   The **Workflow – Outline Project** dialog box opens.
4. In the **Name** text box, enter **PersonPB7** as the name of your load-testing project.
5. Enter an optional project description in **Description**.
6. In the **Type** list, select the **Oracle** option.
7. Click **Next**.

Silk Performer creates a new load-testing project called PersonPB7 with the default settings profile and your local computer as the only agent.

Adding a load-testing script

To run a load test, you must have a test script. Each project contains one or more load-testing scripts. Such scripts define one or a group of virtual users and the actions they perform. Usually, such scripts are modeled on real-world behavior. In this tutorial you will use a **Sample script**, provided by Silk Performer.

Before you can run the load-testing script, you must add it to your project.

Adding a test script

Procedure To add a predefined script to your project:

1. In the menu tree, right-click **Scripts** and select **Add Existing Script**.

   The **Select Script(s)** dialog box opens.

2. Navigate to **<Public User Documents>\Silk Performer 16.5\Samples\Database\Oracle** and select **OraLoadPers.bdf**.

3. Click **Open**.

   Silk Performer adds the **OraLoadPers.bdf** test script to the **Scripts** folder of your current project and displays the script in a new editor window.

Personalizing the sample script

The OraLoadPers script contains three constant declarations that specify the database to which the virtual users connect. Before you can execute the script, you must change these constant values so that the users connect to your database.

Personalizing the test

Procedure To personalize the sample load-testing script:

1. In the **Active Script** window, select **Symbols/Constants/USERNAME**.

   You see the following code:

```
const
  USERNAME       := "user";     // specify YOUR user name here
  PASSWORD       := "password"; // specify YOUR password here
  CONNECTSTRING  := "orclnet2"; // specify the Oracle connect string
```

2. Replace the “user” string with the user name you wish to use to connect to the database.

3. Replace the “password” string with the password you specified for the user.

4 Replace the “orclnet2” string with the connection string that refers to your database.

5 Select **File/Save** from the menu bar to save your changes.

Executing the test script

Three user groups are set up in the OraLoadPers test script:

- **Creator**: A virtual user of this group creates the tables that the PersonPB7 application requires.
- **Loader**: A virtual user of this group loads the sample data into the database so that the PersonPB7 application can be load tested under real-world conditions.
- **Dropper**: A virtual user of this group removes the sample data and deletes the tables that were created earlier for PersonPB7.

To create the tables and insert the sample data for the remaining tutorials, you must first run a virtual user of the Creator group and then a user of the Loader group.

Executing the test script

Procedure To execute the OraLoadPers test script:

1 In the **Active Script** window, expand the **User Groups** folder.

2 In the **User Groups** folder, right-click **Creator** and select **Run User “Creator”**.

Silk Performer runs a virtual user, performing all the actions defined in the test script, that is, creating the tables in the database that the PersonPB7 application requires. While the virtual user runs, you can watch its progress in the **Monitor** window, for example the number of transactions executed, the response time of the last transaction, and the average response time.

3 Monitor the load test and wait until the virtual user has finished executing its tasks.

a In the top part of the **Monitor** window, select an agent computer or a user group that you wish to monitor.

In the bottom part of the **Monitor** window, Silk Performer displays overview information about the virtual users running on the selected agent computer, or belonging to the selected user group.

b In the bottom part of the **Monitor** window, right-click a virtual user for which you want to view detailed information and select **Show Output**.

Silk Performer then displays detailed run-time information about the selected user in the **Virtual User** window at the bottom (e.g., transactions and functions the user executes, and the data the user sends to and receives from the server). The type of information displayed depends on the options selected at the top of the Monitor window.

Display Errors. Select this button to display appropriate error messages indicating the probable causes of user-related errors.

Display Transactions. Select this button to display a message when a user has finished executing a transaction; messages indicate whether or not transactions were successful.

Display Functions. Select this button to display a message for each function call a given user performs, including function name and its parameters.

Display Info. Select this button to display messages containing additional information about the current action a given user performs.

Display Data. Select this button to show data exchanged with the server.

Display all Errors of all Users. Select this button to display error messages for all errors of all users. Each error message indicates user, agent and probable cause.

4 From the menu bar, select **File/Close** to close the **Monitor** window.

5 In the **User Groups** folder, right-click **Loader** and select **Run User "Loader"**.

Silk Performer runs a virtual user that inserts sample data into the previously created tables.

What you have learned

In Tutorial 1, you learned how to:

- Create a load-testing project
- Add a predefined script to a project
- Execute a script with one virtual user
- View progress information for a load test

Tutorial 2: Creating a script with the Recorder

The Silk Performer Recorder allows you to capture database traffic transferred between a client application and a server. After you record database traffic, you can save it as a test script. You can then easily customize the script and replay it to simulate a large number of virtual users. Recording, customizing, and replaying scripts is covered in depth in Tutorials 2, 3, and 4. In Tutorial 5, you will analyze the results of a load test.

This tutorial includes the following steps:

- Setting up an application profile
- Recording traffic between a client application and a server
- Generating a test script based on recorded traffic
- Trying out a generated test script
- Validating a generated test script

Setting up an application profile

When you want the Silk Performer Recorder to capture traffic exchanged between a client application and a server, you must set up a profile for the client application. Profiles specify client application type and what traffic is to be recorded.

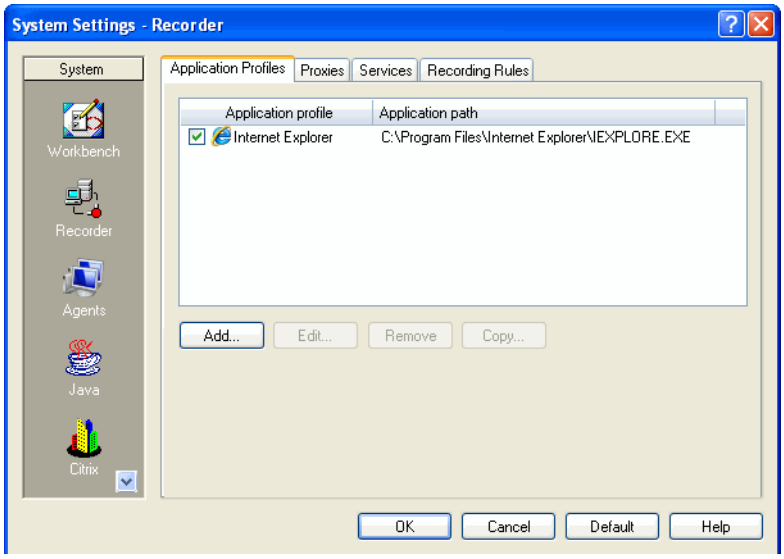
Setting up an application profile

Procedure To set up a profile for the client application:

1 From the menu bar, select **Settings/System**.

The **System Settings – Workbench** dialog box opens.

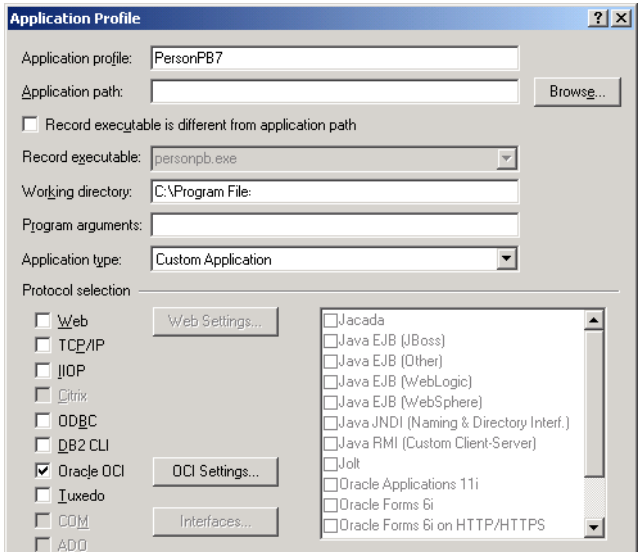
2 In the shortcut list on the left, click the **Recorder** icon.



The list contains all the application profiles that are currently set up on your computer.

3 Click **Add**.

The **Application Profile** dialog box opens.



4 In the **Application profile** text box, enter a name for the application profile, for example **PersonPB7**.

5 Click **Browse** and locate the executable called **PersonPB7.exe**. This file is located in the SampleApps\Oracle subfolder of the Silk Performer home directory.

6 From the **Application type** list box, select the **Custom Application** option.

7 In the **API selection** area, select the **Oracle** option.

8 Click **OCI Settings...**. The **Oracle OCI Settings** dialog box opens.

9 From the list box select **Oci.dll**. Click **OK**.

10 Click **OK**.

Silk Performer adds the new application profile to the profile list in the **System Settings – Recorder** dialog box.

11 Click **OK** to close the dialog box.

Modeling a test script

To load test a database, traffic between the client application and a server must be recorded and described in a test script. The script enables any number of virtual users to perform actions similar to those that were performed during the recording session.

To create as realistic a load test as possible, you will record five separate transactions in a single recording session:

- Logging on to the database
- Searching for customer information
- Updating a customer record
- Inserting a new customer record
- Logging off from the database

When you run the load test, you will create three user groups to which you will assign each of the above tasks in a ratio that simulates real-world behavior.

Modeling a test script

Procedure To model a load-testing script:

1 On the **Workflow** bar, click **Model Script**.

The **Workflow – Model Script** dialog box opens.

2 From the **Application Profile** list box, select **PersonPB7**.

3 Click **Start recording**.

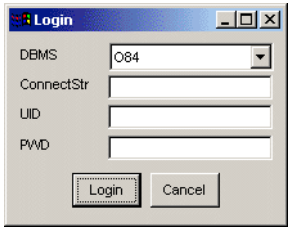
Silk Performer opens the Recorder and starts the PersonPB7 application.

4 In the Recorder, click the **New Transaction** button to insert a new transaction into the test script you are generating.
Confirm the next Silk Performer Recorder message by clicking **Yes.**

The following dialog box opens.

5 Create a new transaction called **TLogon**; then click **OK**. A distinct set of time measurements will be made for each transaction you create. When recording traffic, you should create a new transaction for each distinct user session, from connection to shutdown.

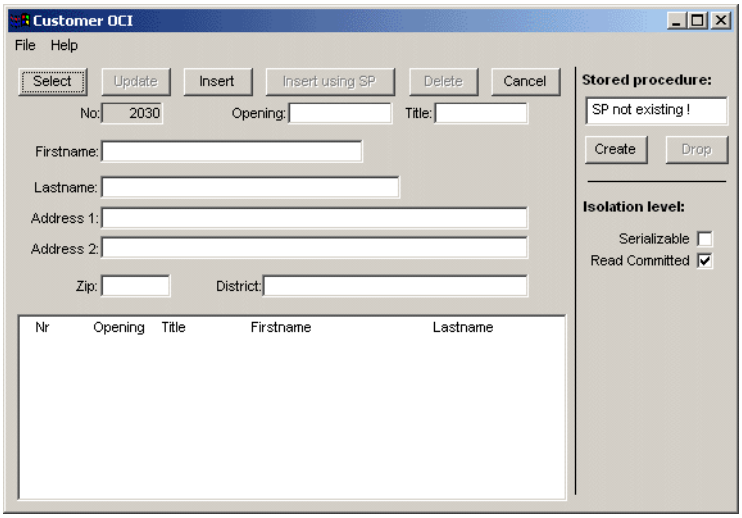
6 In the **Login** dialog box of the PersonPB7 application, select **O84** from the **DBMS** list box.



7 Enter the database connection string, user name, and password for logging on to the database.

8 Click **Login**.

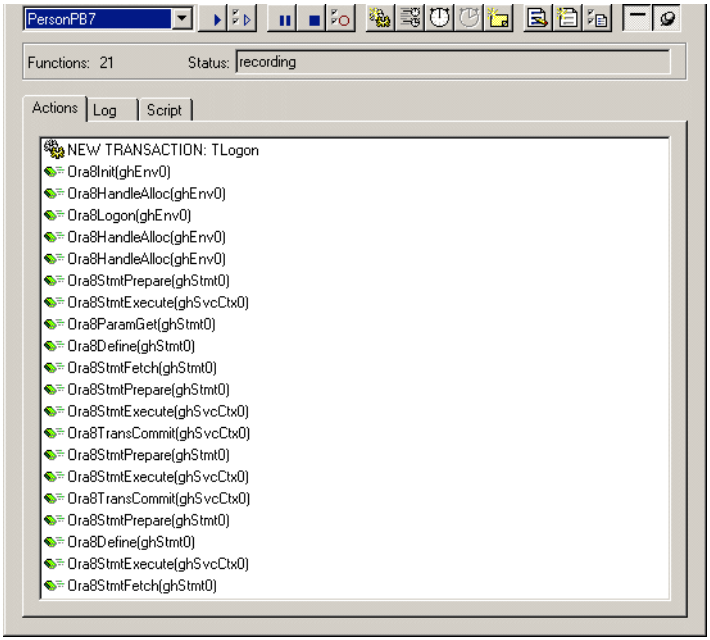
The **Customer** window opens.



The Recorder records the database traffic automatically as you connect to the database.

9 To view your actions and the function calls that the PersonPB7 application performs, click the **Change GUI Size** button.

The Silk Performer **Recorder** window displays as follows.



10 In the Recorder, create a new transaction called **TSearch**.

11 Click **Create** to create a stored procedure.

12 In the **Customer OCI** dialog box, enter a last name for which you wish to search, for example **Williams,** and press **Select**.

13 In the PersonPB7 application, perform the following additional steps. Before you perform each step, create a new transaction with a name that describes the user session, for example **TUpdate** or **TInsert**.

   a Change the customer address

   b Insert a new customer record.

   Note For detailed information on using the PersonPB7 sample application, refer to the *Silk Performer PersonPB Reference.*

14 On the Recorder, create a new transaction called **TLogoff**.

15 In the **Customer OCI** window, select **File/Exit** from the menu bar.

16 On the Recorder, click the **Stop Recording** button.

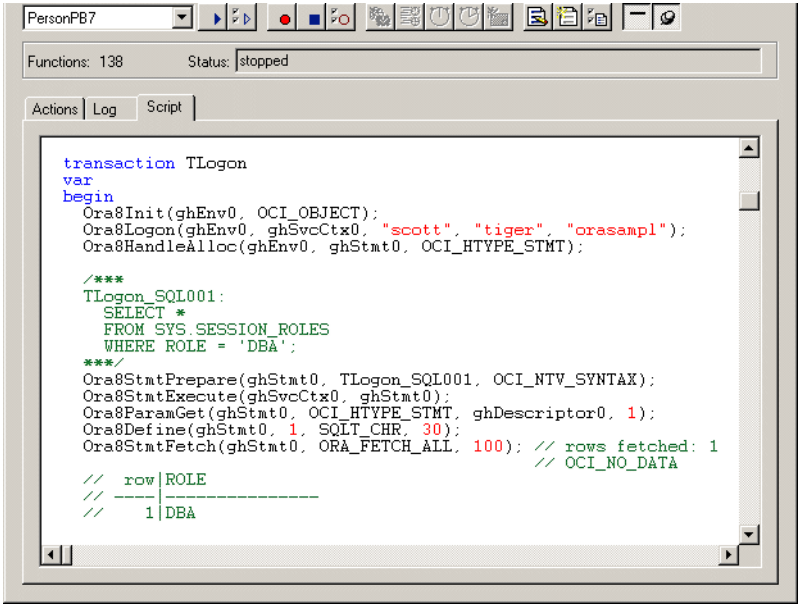
   The **Save As** dialog box opens.

17 In the **File name** text box, enter **PersonPB7**.

18 Click **Save**.

19 You can now decide whether or not to close the Recorder.

If you keep the Recorder open, you can examine the script that was generated.



20 Close the Recorder.

Silk Performer then automatically adds the script to your current load-testing project.

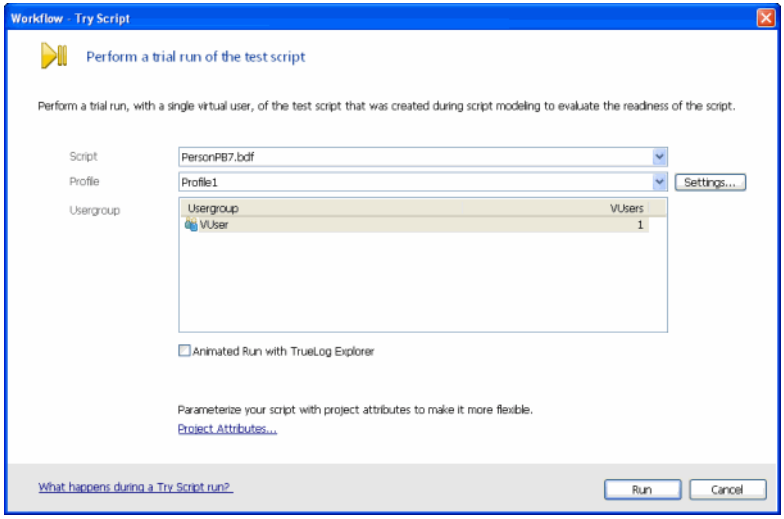
Trying out the generated test script

Next you should do a trial run of the test script that was created. The object of the trial run is to ensure that the script is error free, and that it accurately reproduces the interaction between the client application and the database server.

Trying out the script

Procedure To try out the generated test script:

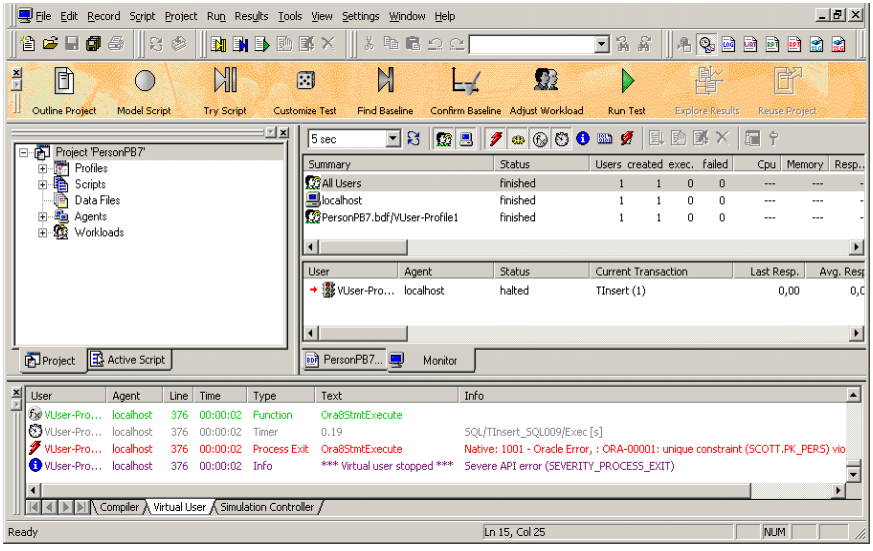
1 In the **Workflow** bar, click the **Try Script** button.



2 From the **Script** list box, select the **PersonPB7.bdf** script.

3 Click **Run**.

Silk Performer runs a single virtual user, called VUser, which performs all the actions that were performed during the recording session. While the single-user test runs, you can watch its progress.



The **Try Script** run failed because of an error. The "… **unique constraint** … **violation** …" error message in the **Virtual User** output tab indicates that the script attempted to insert a record with a primary key that was not unique within the data table.

Your script attempted to use the same value that was used during recording. This is a typical problem when replaying scripts that include session dependent, dynamic information. Tutorial 3 shows you how to customize your script to handle such session dependent, dynamic information.

Specifically the script attempted to add a new person record that didn't have a unique person ID. Person ID's are numbers from 1 to 9999. The highest used person ID is stored by the PersonPB7 sample application in a separate data table. Before adding a new record, PersonPB7 retrieves the value of the highest used person ID, increments the number up by one, and updates the data table. Then it retrieves the new value and uses it as an input parameter for the insert statement.

What you have learned

In Tutorial 2, you learned how to:

- Set up an application profile
- Record traffic between a client application and a database
- Generate a test script based on recorded traffic
- Try out a generated test script

Tutorial 3: Customizing a generated test script

Customizing a generated test script allows you to use the actions performed by a single user to realistically simulate the actions of multiple users.

Three key tasks assist in creating real-world simulations:

- Handling session dependent, dynamic information
- Replacing constant values with random variables
- Specifying the tasks each user is to perform

Note Customizing a test script can be done through either Silk Performer Workbench or TrueLog Explorer. When using Silk Performer Workbench to customize generated scripts you must use Benchmark Description Language (BDL), Silk Performer's high-level scripting language. Although knowledge of BDL is not required for this tutorial, it is recommended that you read the BDL introductory topics in Silk Performer's online help system.

No knowledge of BDL is required to use TrueLog Explorer. TrueLog Explorer performs all script manipulation for you. In this tutorial TrueLog Explorer is

used to handle session relevant information. To learn more about the functionality and use of TrueLog Explorer, please refer to the *TrueLog Explorer Help*.

Handling session dependent, dynamic information

Script customization must be performed whenever session dependent, dynamic information is used by a database application. In most cases customization involves parsing the result value of a database operation into a variable and substituting the input parameter value of another database operation with the parsed value.

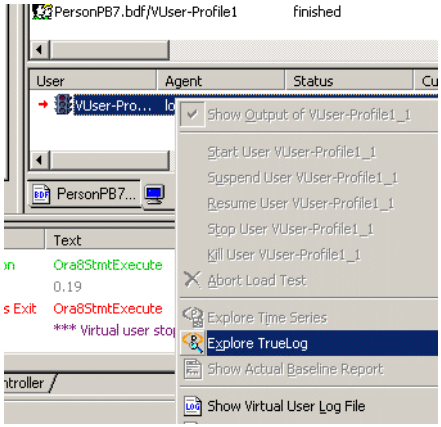
"Tutorial 2: Creating a script with the Recorder" explained the problem of not using a unique primary key within a database table and generating an "unique constraint violation" error.

To handle this situation the result of a database operation must be parsed into a variable. The variable's value must then be used in place of the recorded value as an input parameter for a subsequent database call.

Parsing a result into a variable

Procedure To parse the result of a database operation into a variable:

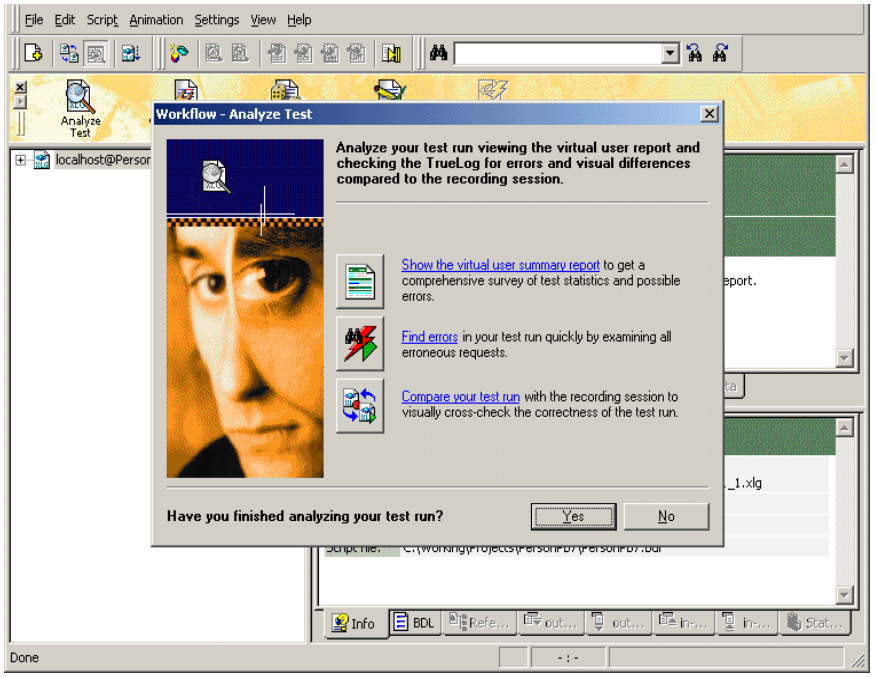
1 After performing a **Try** Script run, right click the VUser Try Script status line in the **Monitor** window and choose **Explore TrueLog**.



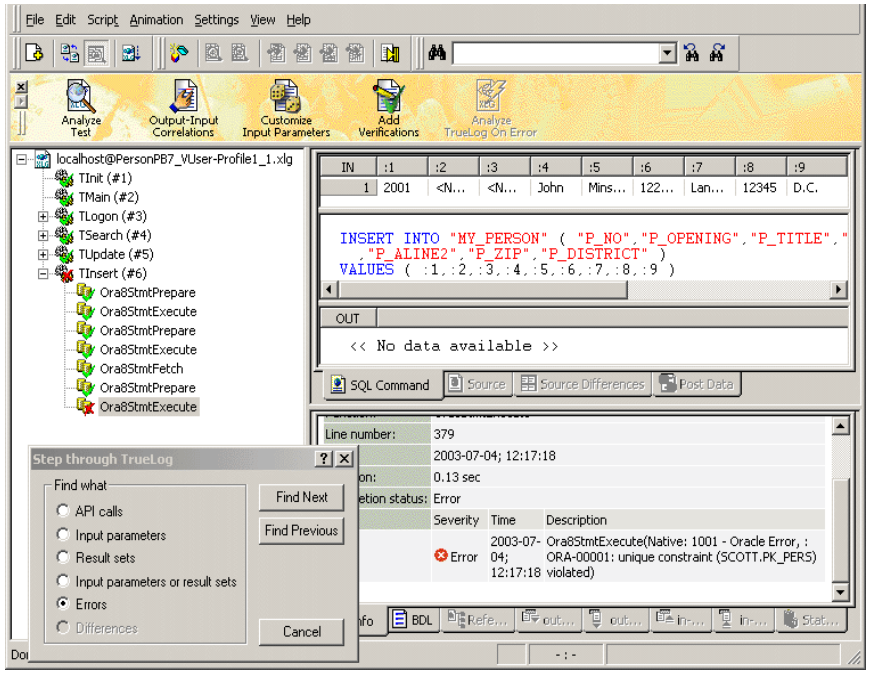
TrueLog Explorer launches, loaded with the replay TrueLog of your **Try Script** run.

2 Click **Analyze Test** on the **TrueLog Explorer Workflow** bar.

The **Workflow - Analyze Test** dialog box opens.

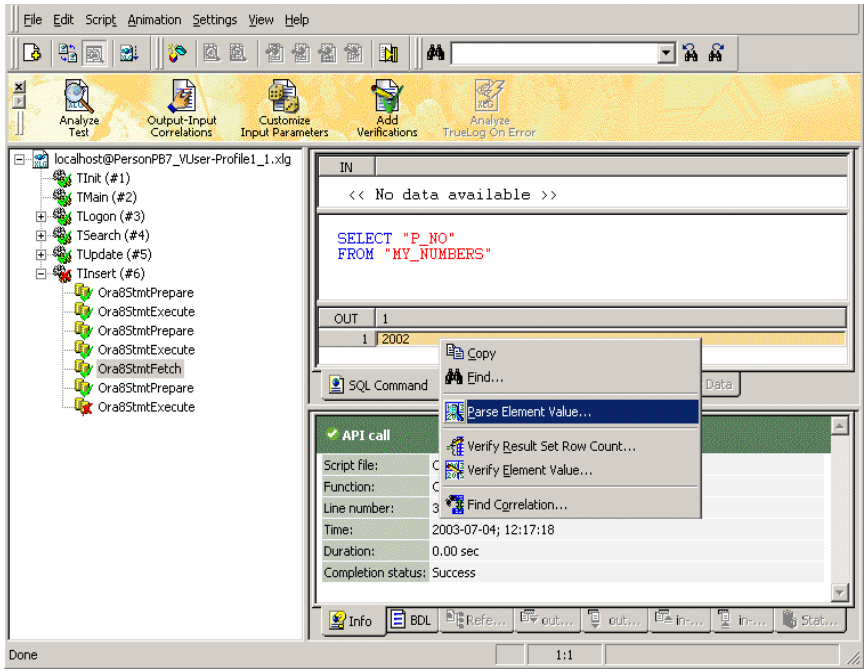


3 Click **Find Errors**. Click the **Errors** option button on the **Step Through TrueLog** dialog box. Continue clicking **Find Next** until you receive a **No more Errors** message.

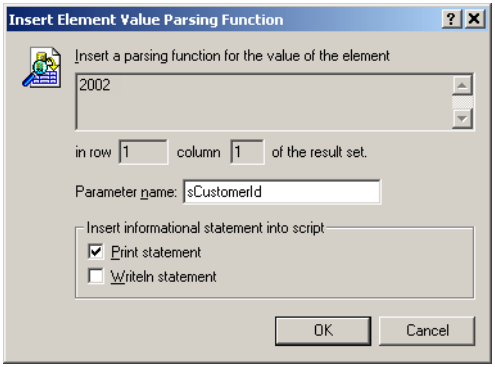


The database operation that caused your **Try Script** run to fail is now selected. On the **Info** tab you can see an error message that describes the reason for the failure.

4 Now select the closest preceding Fetch statement. Select the value in the output parameter table, right-click, and choose **Parse Element Value**.



5 The **Insert Element Value Parsing Function** dialog box opens.



Specify the name of the variable you want the value to be parsed into in the **Parameter name** field for example **sCustomerId**, and click **OK**.

Confirm the following message box by clicking **OK**. Click the **BDL** tab in the lower section of TrueLog Explorer. TrueLog Explorer then adds the following lines to your script.

```
sCustomerId := RsGetString("1", 1);
Print("sCustomerId: " + sCustomerId);
```

Additionally TrueLog Explorer declares a global variable in the *dclparam* section of your script.

```
dclparam
  sCustomerId       : string;
```

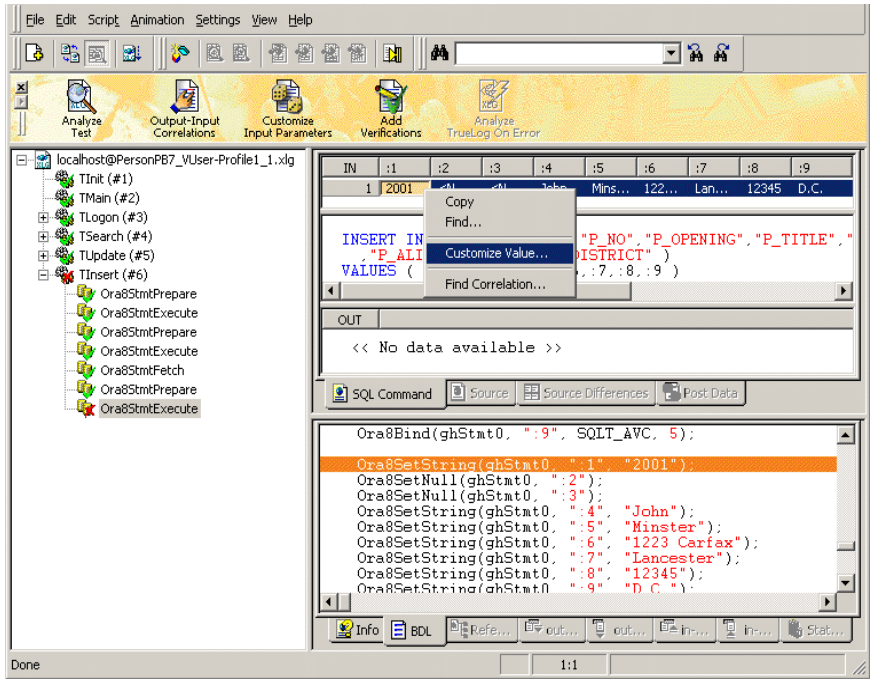
Substitution of an input parameter

Procedure To substitute an input parameter with a variable:

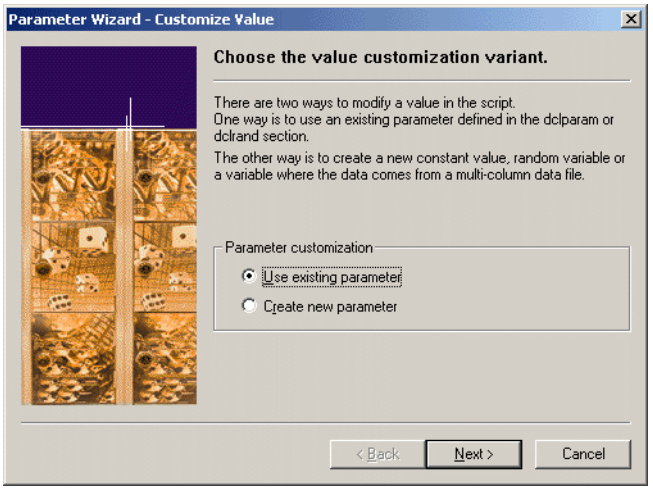
1 After parsing the session relevant data into a variable you can use the value of the variable rather than the formerly passed parameter value as input for a database operation.

In this case you must substitute the parameter value passed to the database operation that caused the **Try Script** run to fail.

To do this, select the database operation node immediately preceding the node that failed, right click the left-most element in the input parameter view and choose **Customize Value**.

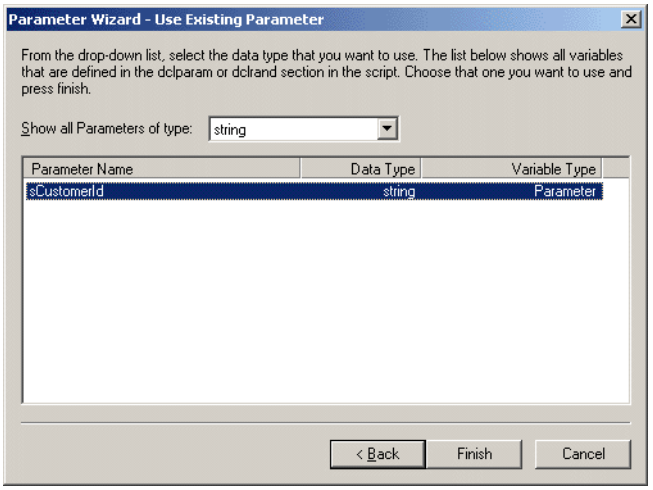


2 The **Parameter Wizard - Customize Value** dialog box opens.



Select **Use existing parameter** and click **Next** to proceed.

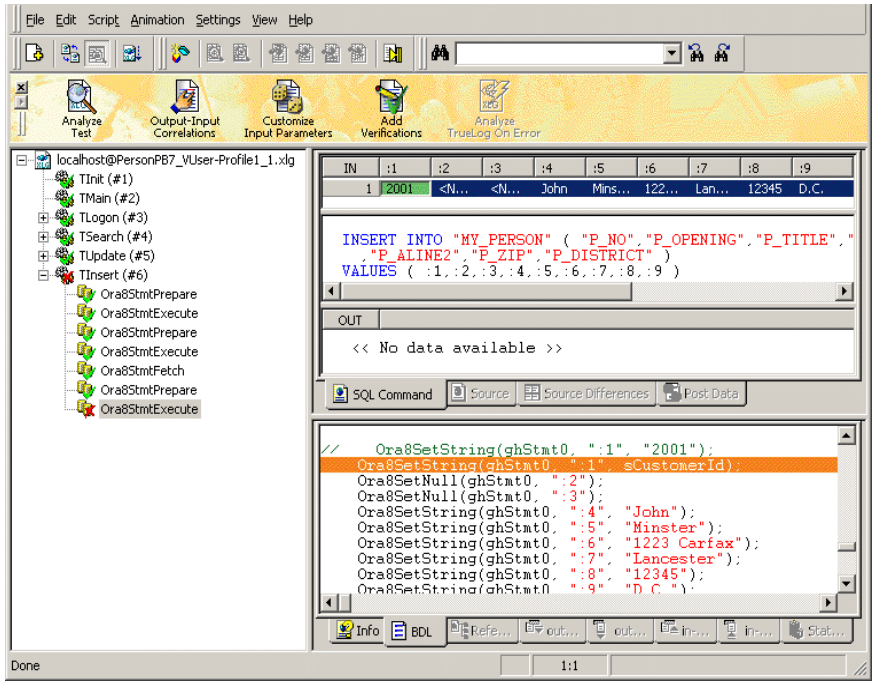
3 The **Parameter Wizard - Use Existing Parameter** dialog box opens.



Select the name of the variable for which you previously created the parsing function, for example **sCustomerId**, and click **Finish**.

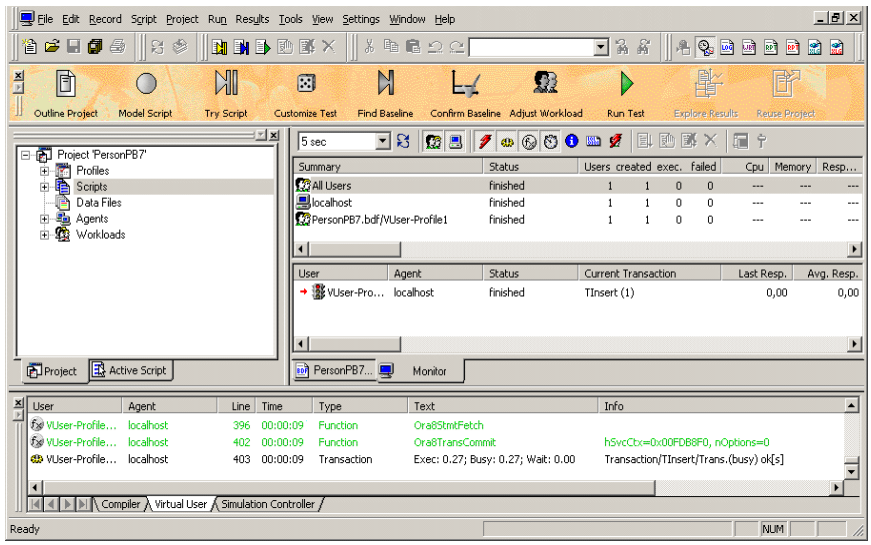
Confirm the following message box by clicking OK, and click the BDL tab in the lower section of TrueLog Explorer. TrueLog Explorer then adds the following lines to your script:

```
//    Ora8SetString(ghStmt0, ":1", "2001");
    Ora8SetString(ghStmt0, ":1", sCustomerId);
```



Handling of session dependent, dynamic information is now complete.

Close **TrueLog Explorer**, activate **Silk Performer Workbench**, and do a trial run of the customized script as was shown in **Tutorial 2**. The **Try Script** run should now proceed without error.



Replacing constant values with random variables

When randomized data is required, random variables are employed to provide realistic user data. In this way, Silk Performer furnishes virtual users with varied personal data, for example name, address, and phone number.

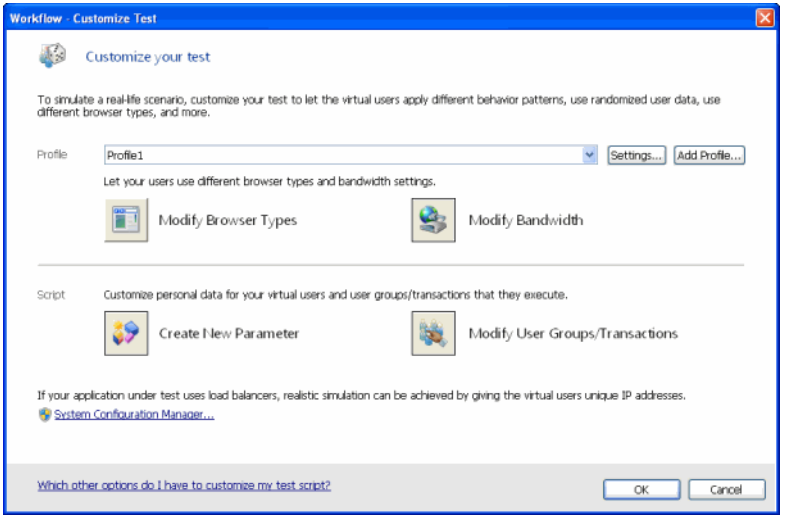
Replacing constant values

Procedure To replace constant values with random variables:

> **Note** Ensure that the PersonPB7.bdf script is active in Silk Performer Workbench. To do this, click the PersonPB7.bdf tab.

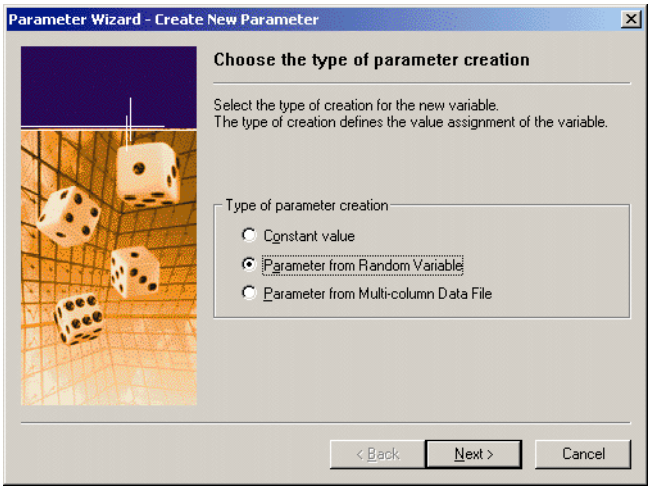
1 In the **Workflow** bar, click the **Customize Test** button.

The **Workflow – Customize Test** dialog box opens.

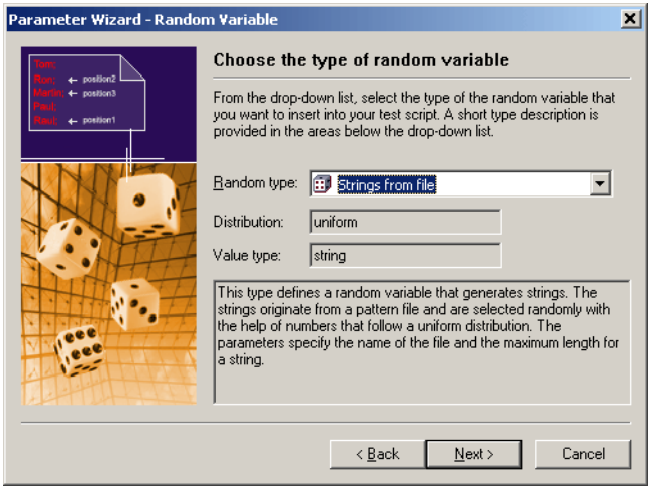


2 Click the **Create New Parameter** icon.

The **Parameter Wizard - Create New Parameter** dialog box opens.

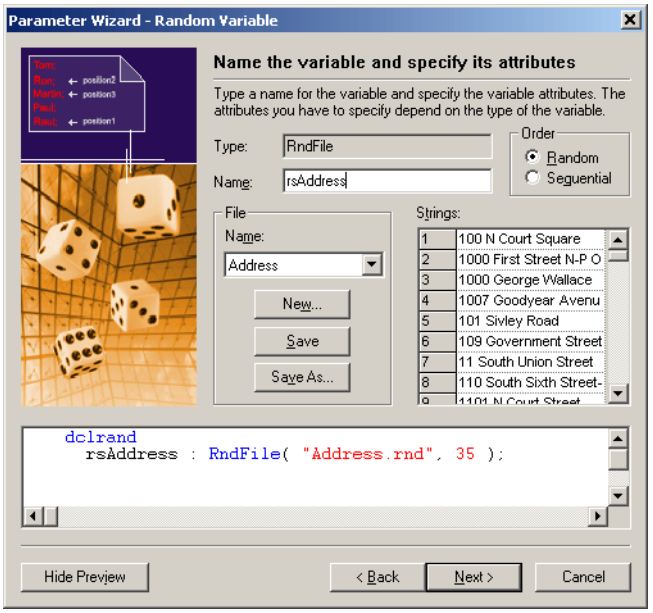


3 Select **Parameter from Random Variable** and click **Next**.
The **Parameter Wizard - Random Variable** dialog box opens.



4 From the **Random type** list box, select the **String from file** type.

5 Click **Next**.



6 Enter a name for your variable in the Name edit field, for example **rsAddress**.
In the **File** area, select **Address** from the **Name** list box.

This random variable selects an address of up to 35 characters in length from a Silk Performer file that contains random addresses.

7 Click **Finish**.

Silk Performer inserts the following lines into your test script.

```
dclrand
  rsAddress : RndFile("Address.rnd", 35);
```

8 Locate the TUpdate transaction.

Do this by expanding the **Transactions** folder in the **Active Script** window and double-clicking **TUpdate**.

9 In the TUpdate transaction, locate the section where the UPDATE statement is executed.

This section looks like the following.

```
/***
TUpdate_SQL007:
  UPDATE "MY_PERSON"
  SET "P_ALINE1" = :1
  WHERE "P_NO" = :2 ;
***/
Ora8StmtPrepare(ghStmt0, TUpdate_SQL007, OCI_NTV_SYNTAX);
```

```
Ora8Bind(ghStmt0, ":1", SQLT_AVC, 15);
Ora8Bind(ghStmt0, ":2", SQLT_CHR, 3);

Ora8SetString(ghStmt0, ":1", "First Avenue");
Ora8SetString(ghStmt0, ":2", "204");

Ora8StmtExecute(ghSvcCtx0, ghStmt0, 1);
```

10 Replace the fourth parameter of the first Ora8Bind function call with **40**.

11 Replace the address that you entered earlier (“First Avenue” in this case) with the **rsAddress** variable.

The code section should now look like the following. The code shown in bold type is what was changed.

```
Ora8StmtPrepare(ghStmt0, TUpdate_SQL007, OCI_NTV_SYNTAX);

Ora8Bind(ghStmt0, ":1", SQLT_AVC, 40);
Ora8Bind(ghStmt0, ":2", SQLT_CHR, 3);

Ora8SetString(ghStmt0, ":1", rsAddress);
Ora8SetString(ghStmt0, ":2", "204");

Ora8StmtExecute(ghSvcCtx0, ghStmt0, 1);
```

Specifying which tasks each user is to perform

Usually a load test simulates a number of different user types. In the test script, you must specify which action each virtual user type is to perform.

Assigning tasks to users

Procedure To specify which tasks each user is to perform:

1 Navigate to the **dcluser** section of the PersonPB7.bdf script file. Do this by selecting **User Groups/VUser** in the **Active Script** window.

You will see the following:

```
dcluser
  user
    VUser
  transactions
    TInit           : begin;
    TLogon          : 1;
    TSearch         : 1;
    TUpdate         : 1;
    TInsert         : 1;
    TLogoff         : 1;
```

In this section, a single user group called **VUser** is defined. By default, this user group performs each of the transactions you created (TLogon, TSearch, TUpdate, TInsert, and TLogoff) once.

2 Edit the **dcluser** section of the script so that it looks like the following.

```
dcluser
  user
    Searcher
  transactions
    TLogon          : begin;
```

```
    TSearch         : 5;
    TLogoff         : end;

  user
    Updater
  transactions
    TLogon          : begin;
    TSearch         : 3;
    TUpdate         : 2;
    TLogoff         : end;

  user
    Inserter
  transactions
    TLogon          : begin;
    TSearch         : 1;
    TInsert         : 3;
    TLogoff         : end;
```

In the edited version of the script, the **Searcher** user group performs the TLogon transaction at the beginning, then the TSearch transaction five times, and finally the TLogoff transaction at the end. The **Updater** user group performs the TLogon transaction at the beginning, the TSearch transaction three times, the TUpdate transaction twice, and the TLogoff transaction at the end. The **Inserter** user group performs the TLogon transaction at the beginning, the TSearch transaction once, the TInsert transaction three times, and the TLogoff transaction at the end.

3. Select **File/Save** from the menu bar to save your changes.
4. On the **Workflow** toolbar, click the **Customize Test** button.

   The **Workflow – Customize Test** dialog box opens.
5. Click **OK** to confirm that you're done customizing the test script.

What you have learned

In Tutorial 3, you learned how to:

- Replace constant values with random variables
- Specify which tasks each virtual user is to perform

Tutorial 4: Replaying a customized script

Once you have created a test script by recording traffic, and then customized the script, you can load test your server by replaying the script. To run a load test, you must perform the following steps:

- Find the test baseline
- Confirm the test baseline
- Set up server monitoring for your load test
- Execute the load test

Finding the test baseline

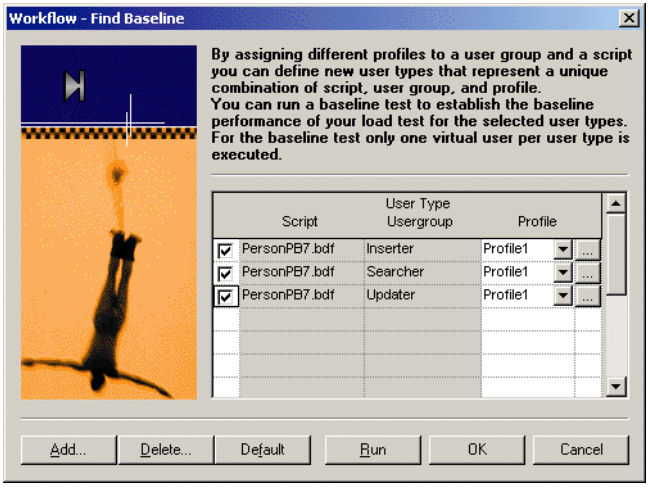
Once you have customized the load-testing script, you must determine baseline performance. This is the ideal performance of your application when it is not under significant pressure. To identify the test baseline, the fully customized script is run with just one user per user group. This will provide a solid basis for comparison when the application is later placed under stress during the load test.

Finding the baseline

Procedure To find the load test baseline:

1. On the **Workflow** toolbar, click the **Find Baseline** button.

   The **Workflow – Find Baseline** dialog box opens.



   This dialog box lists all the user groups that have been set up in your test script.

2. Click **Run**.

   Silk Performer runs one virtual user from each user group declared in the test script. While the test runs, you can watch its progress in the **Monitor** window.

Confirming the test baseline

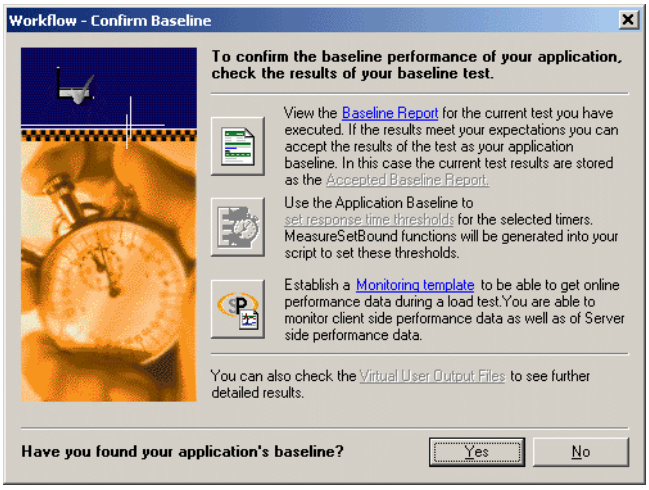
After running the baseline test, you must confirm that the test result reflects the desired performance of the tested server. To do this, inspect the results of the test. If they are satisfactory, the baseline has been established and can later form the basis for comparison with results from full load tests.

Confirming the baseline

Procedure To confirm the load test baseline:

1 On the **Workflow** toolbar, click the **Confirm Baseline** button.

The **Workflow – Confirm Baseline** dialog box opens.



2 Click the **Baseline Report** button.

The Baseline Report opens.

3 Check the report carefully. In particular, make sure that no errors occurred during the baseline test.

4 Click the **Accept Baseline** button on the Baseline Report.

5 Click **Yes** and **OK** on the following dialog boxes.

6 If your baseline report doesn't indicate any errors, click **Yes** on the **Workflow – Confirm Baseline** dialog box to confirm that you have identified the baseline.

Adjusting the workload

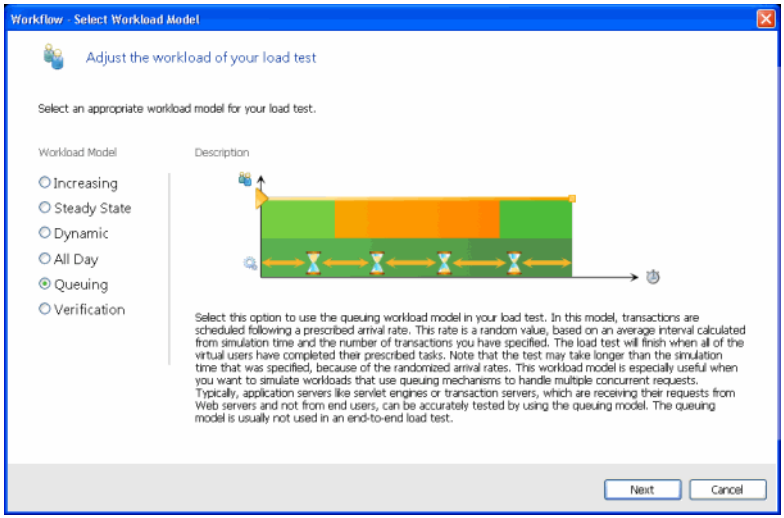
Once you have confirmed your baseline you can select a workload model and prepare the workload to run a full load test.

Adjusting the workload

Procedure To adjust the workload:

1 On the **Workflow** toolbar, click the **Adjust Workload** button.

The **Workflow - Select Workload Model** dialog box opens.



This dialog box shows all possible workload models that can be selected.

2 Select the **Queuing** workload model.

3 Click **Next**.

4 On the **Workflow Assign Agents** dialog box, select an option and click **OK**.

Silk Performer opens the Workload Configuration dialog box.

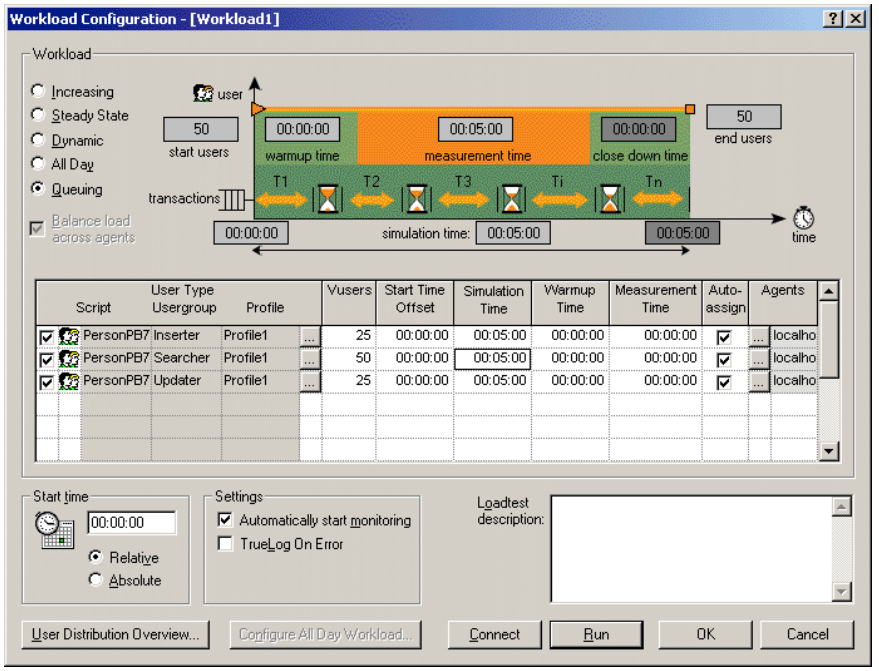
Running the load test

Once you have adjusted the workload, you're ready to run a full load test. The test script is run with multiple virtual users to test the database server. For this tutorial, only your local computer is required. A true load test however requires an appropriate testing environment, set up on the local area network, that offers a full complement of agent computers to host virtual users.

Running the test

Procedure To run the load test:

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



2 In the **VUsers** column, change the number of Inserter users to **25**, the number of Searcher users to **50**, and the number of Updater users to **25**.

3 For each of these user groups, change the simulation time to **300** seconds (5 minutes).

You are now ready to run the load test. When you run a test Silk Performer's sophisticated monitoring tool, **Performance Explorer**, automatically generates a live graphical display with a default set of test data for the server under test. You can disable this default option by deselecting the **Automatically start monitoring** setting. You can also change the default monitoring settings by pressing the **Active profile** button (See Customizing server monitoring). Once monitoring has been set up as required, you can run the test.

4 Disable all user groups that are declared in the OraLoadPers test script.

5 Click **Run** to start the test, or click **OK** to save the settings and run the test later.

When you start the test, Silk Performer runs the load test against the database server with 100 virtual users.

If the default **Automatically start monitoring** option is enabled, **Performance Explorer** will open and generate a live graphical display of the specified test data. You can adapt these views by adding and

deleting specific measurements. To do this expand the tree menu on the left and drag and drop selected measurements into existing or new view graphs.

Customizing server monitoring for your load test

An important feature provided by Silk Performer is server monitoring during actual load tests. Server monitoring helps you locate and analyze bottlenecks on servers. You can separately examine the performance of the operating system, the network and the server application (which in this case is the Oracle server).

Server monitoring is performed using Performance Explorer, a powerful tool incorporated into Silk Performer. When you run a test, the default procedure has Performance Explorer automatically open a set of views containing a live graphical display of a selection of test data that is relevant to the server under test. You can disable this default option by deselecting the **Automatically start monitoring** setting on the **Workload Configuration** dialog. You can also customize the default monitoring settings and select a particular set of data sources.

Customizing server monitoring

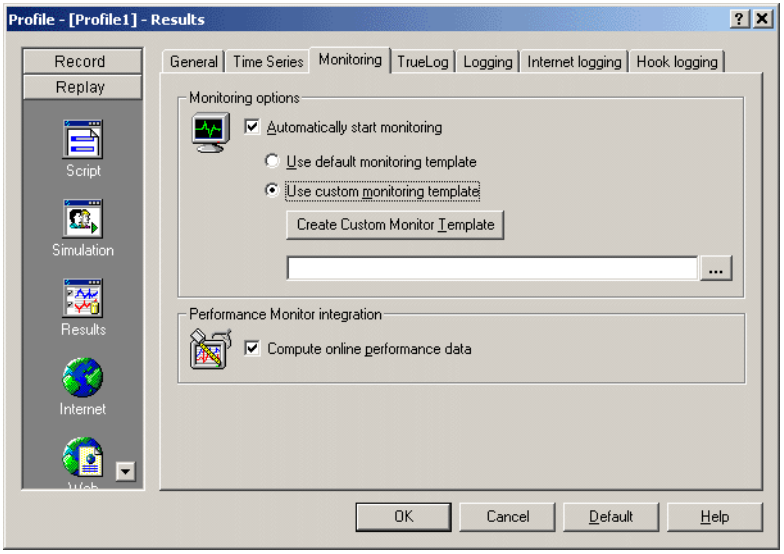
Procedure To customize server monitoring for your load test:

1. On the **Workflow** toolbar, click the **Run Test** button.

   The **Workload Configuration** dialog box opens.

2. In the **Settings** section, click the **Active Profile** button.

   The **Profile - [Profile1] - Results** dialog box opens.



3 In the shortcut list on the left, select the **Replay** category and click the **Results** icon.

4 Select the **Monitoring** tab

The **Automatically start monitoring** option is activated by default. This means that when you run a load test, Performance Explorer opens automatically using a default template that invokes a display of relevant client information.

5 If you want to customize monitoring, enable the **Use custom monitoring template**, and click the **Create Custom Monitor Template** button.

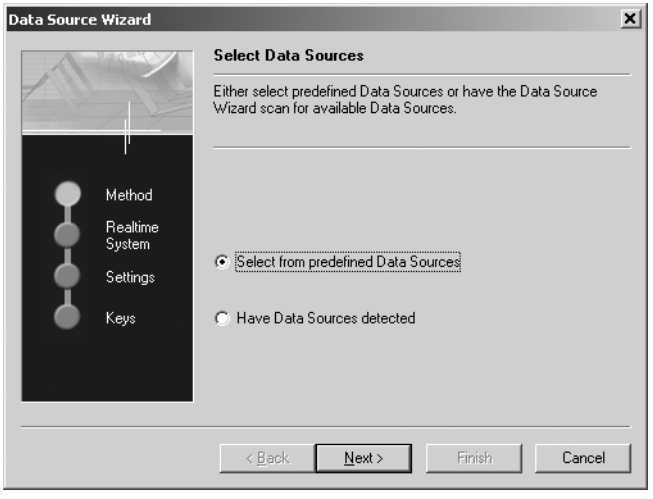
Performance Explorer creates and opens a template with the name of your project, which is the same as the default template.

6 Click **Edit Custom Monitor Template** (Performance Explorer opens) to customize the new template and change the information display. You can close or maintain the default monitoring windows and you can add one or more windows to provide additional information.

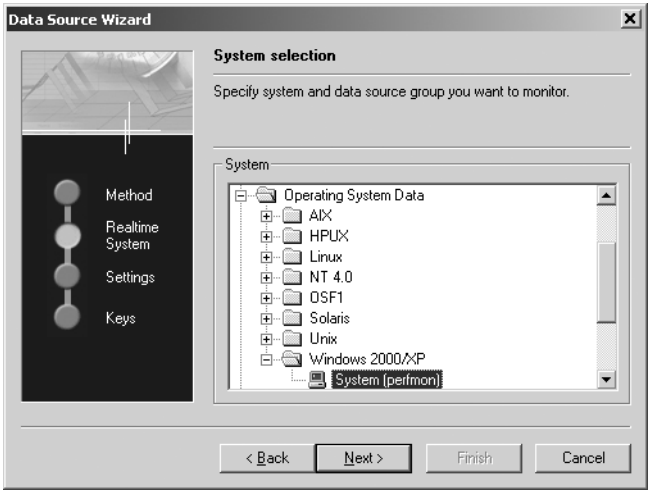
To set up **Performance Explorer** to display additional information, do the following:

a From the menu bar, select **Monitor/Add Data Source**.

The **Data Source Wizard** opens.



b Click **Next**.
The **Data Source Wizard - System selection** dialog box opens.



c In the tree view, expand the folder that corresponds to the operating system on which the database server runs. Select **System**.

d Click **Next**.

e In the **Connection Parameters** dialog box, enter connection parameters, like the host name or IP address of the database server, the connection port, the user name, and the password.

f The data you enter here depends on the operating system that runs on the computer you are monitoring.

g Click **Next** once you have entered the connection parameters.

h In the next dialog box, select the performance counters you wish to monitor.

Of particular interest are processor and memory utilization of the server under test.

i Click **Finish**.

j Performance Explorer opens a new window, displaying real-time data for the performance counters you have selected.

k In the **Monitor** tree view, right-click the name of the computer you are monitoring and select **Add Data Source**.

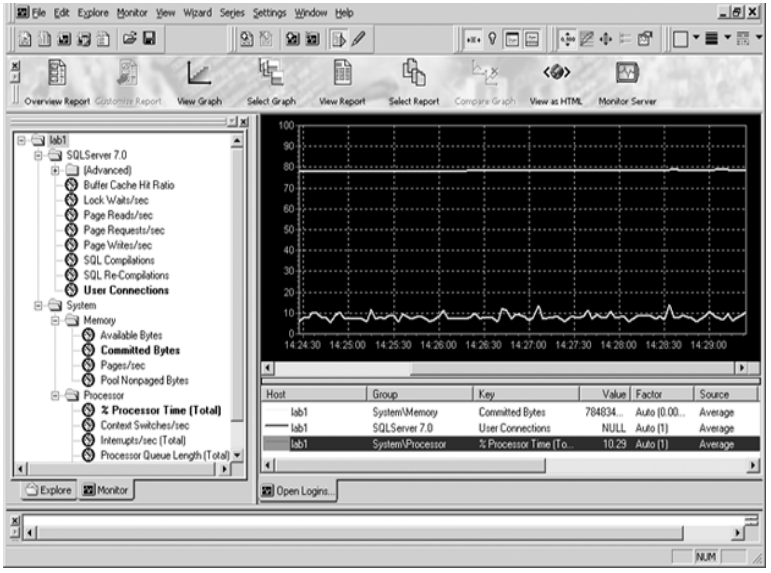
l The Data Source Wizard opens.

m Repeat steps A through E. However this time:

I On the first dialog box, select the name of the Oracle server software rather than **System**.

II On the third dialog box, select which performance counters you wish to monitor.

Later, when you run the load test, the monitor view will look similar to the following.



What you have learned

In Tutorial 4, you learned how to:

- Find a test baseline
- Confirm a test baseline
- Customize monitoring for a load test
- Adjust a workload
- Execute a load test

Tutorial 5: Viewing the results of a load test

Silk Performer incorporates **Performance Explorer**, a powerful graphing and analysis tool, to assist you in working with results information.

Performance Explorer enables you to analyze results information with advanced features for creating statistical reports and displaying performance results in real-time generated graphics. Results information varies depending on the type of application and/or server under test. In general, the following is available:

- **Response time information**: This is the total time it takes to process a given timer; it provides application performance information from the client perspective.
- **Throughput information**: This is the average rate at which a given timer is processed by the server; it allows you to analyze performance from the server perspective.

Overview report of performed measurements

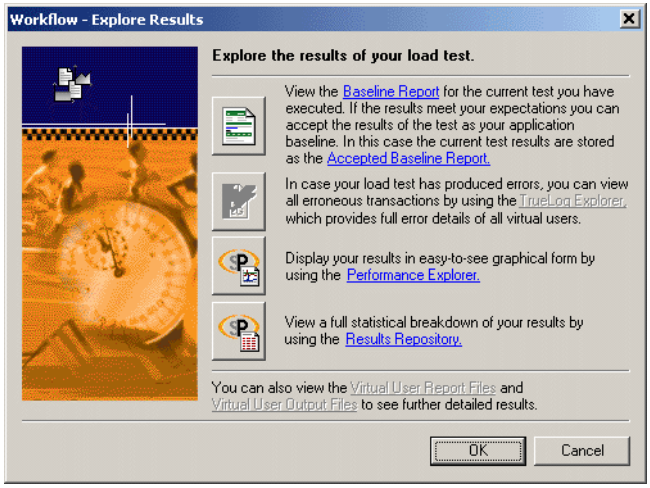
You can consult an HTML overview report that contains results of all performed measurements.

Exploring the overview report

Procedure To explore an overview report:

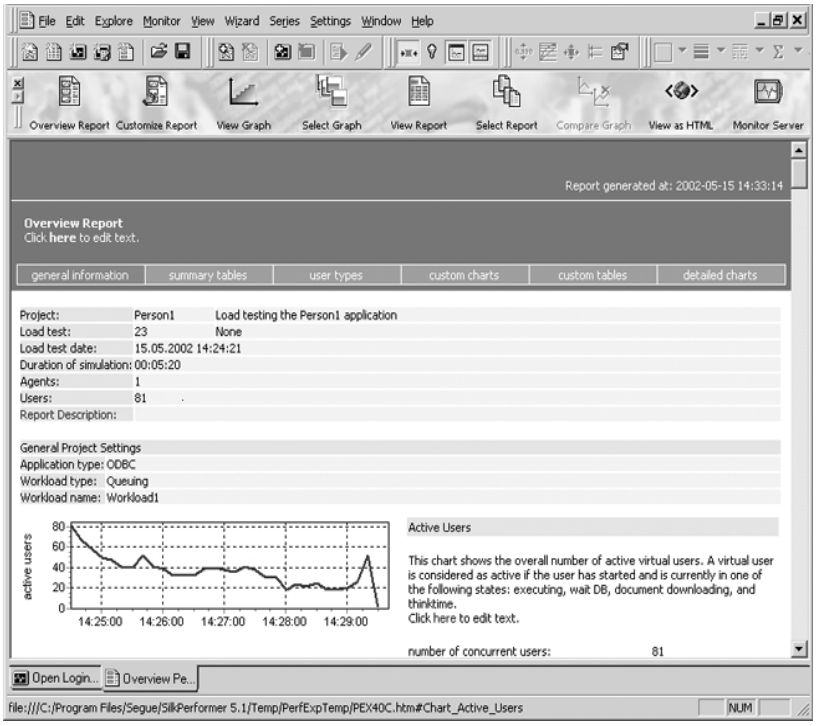
1 On the **Workflow** toolbar, click the **Explore Results** button.

The **Workflow - Explore Results** dialog box opens.



2 Click the **Performance Explorer** button.

Performance Explorer opens, and the HTML overview report is generated.



This report offers short explanations of displayed graphs. More detailed information is also available.

Detailed response time measurements

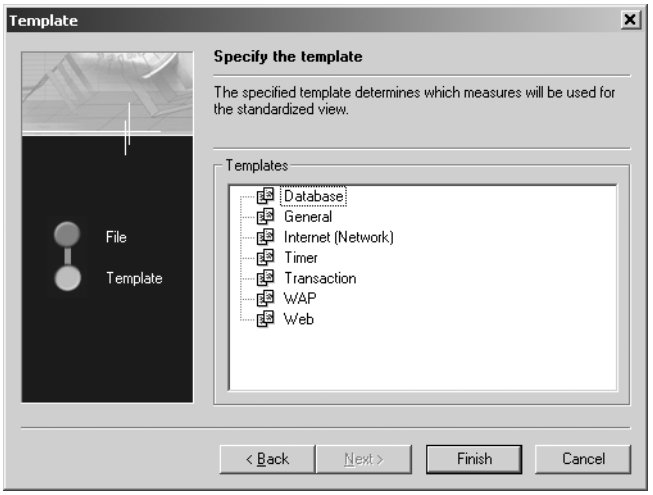
When you analyze the performance of an application from the client point of view, you examine the response times of each transaction and timer. This enables you to find out how users will experience interaction with server objects. A common goal in load testing is to find out if response times for all transactions remain below a specified, critical limit.

Response time details

Procedure To explore detailed response time measurements:

1 On the Performance Explorer **Workflow** bar, click the **Select Graph** button.

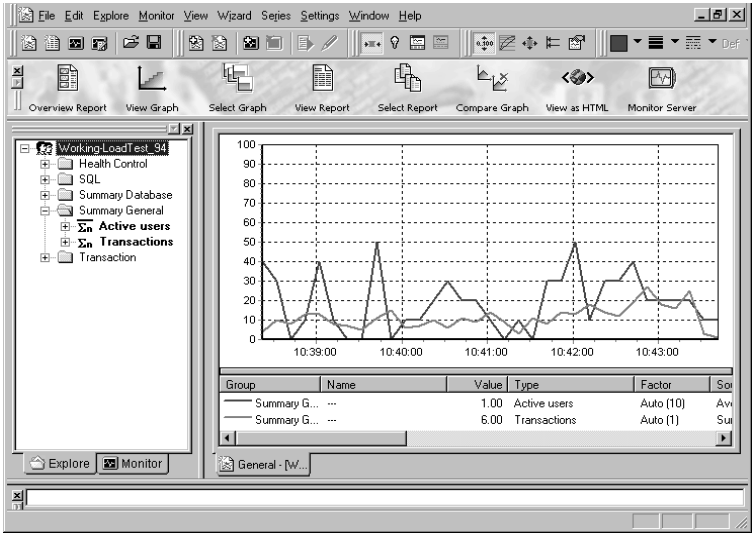
The **Template** dialog box opens.



2 In the **Templates** area, select the **Timer** option.

3 Click **Finish**.

Performance Explorer displays a graph of the active users.



You see the default graph with the average number of active users, and one or more default measurements (if any).

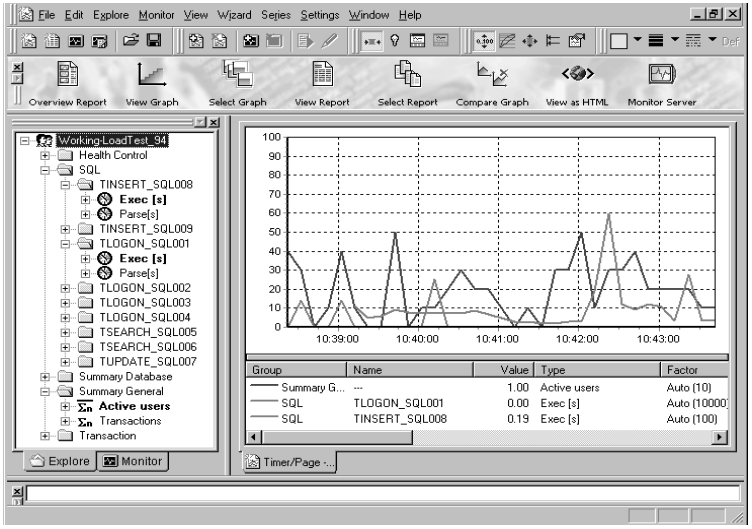
4 Expand the **Explore** tree view.

5 In the **Explore** tree view, select **SQL Commands/TLOGON_SQL001**.

6 Drag the **Exec** measurement into the new view.

7 In the **Explore** tree view, select **SQL Commands/TINSERT_SQL008**.

8 Again, drag the **Exec** measurement into the new view.



The load test you ran in the previous tutorial lasted only five minutes, so there are few measurements, which makes it difficult to draw any conclusions.

Real load tests can last several hours, a few days, or even a week. Such complete tests provide results information that allows you to accurately analyze server performance.

Exploring throughput measurements

When you analyze the performance of a database server from the point of view of the server, you examine **throughput rate**. Throughput is work done by a server during a specific time interval (e.g., number of requests that a server processes within a second, minute, or hour). A typical goal in load testing is to ensure that throughput rate remains above a specified, critical limit.

Throughput rate details

Procedure To explore throughput measurements:

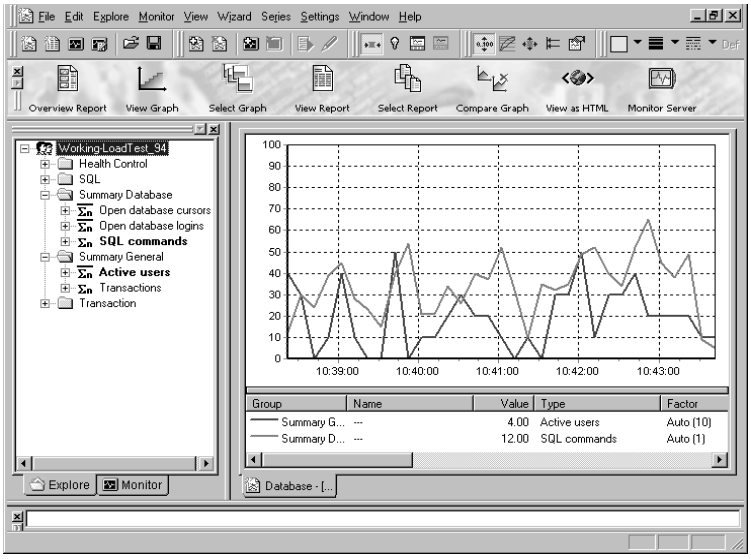
1 On the Performance Explorer **Workflow** bar, click the **Select Graph** button.

The **Template** dialog box opens.

2 In the **Templates** area, select the **Database** group.

3 Click **Finish**.

Performance Explorer opens a new chart that resembles the following.



Again, since the load test you ran lasted only five minutes, it's difficult to draw conclusions from the results.

What you have learned

In Tutorial 5, you learned how to:

- Explore response time measurements
- Explore throughput measurements

5

Load Testing a CORBA Application

Introduction

This chapter contains a series of tutorials that will help you to start load-testing CORBA applications with Silk Performer.

What you will learn

This chapter contains the following sections:

Overview

Each of the following tutorials provides step-by-step instructions for important Silk Performer tasks. You should perform the tutorials in order, since each tutorial builds on what you have learned in the previous ones.

You will learn how to do the following:

- Execute a predefined load test on your local machine.
- Create a test script by recording IIOP traffic using the Recorder.
- Customize the recorded test script.

- Run the customized script, using multiple virtual users.
- Use reporting tools to view throughput and response time information.

Sample: VisiBroker Bank application

These tutorials use the sample CORBA Bank application that accompanies Inprise VisiBroker. This application is included on the VisiBroker CD-ROM and should have been installed in the examples\basic\bank_agent subfolder of the VisiBroker home directory as part of the installation and setup.

Tutorial 1: Running a sample load test

In this tutorial you will learn to create a load-testing project and execute sample scripts that are provided with Silk Performer. You will run one of the sample scripts with a single virtual user on your local machine.

You will use the test script VBrokerBank01.bdf, which is included in the **<Public User Documents>\Silk Performer 16.5\Samples\CORBA\VisiBroker** folder. This sample was created using VisiBroker 4.5.

Creating a load-testing project

For each server you load test with Silk Performer, you have to create a project file. A project file administers all the settings that are associated with a series of load tests, like the test type, the workload model, and a number of options that depend on the type of server being tested.

Creating a project

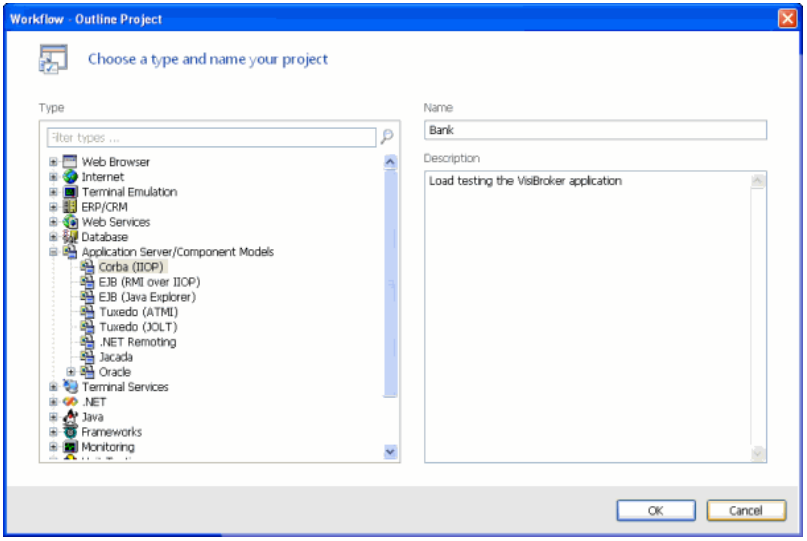
Procedure To create your load-testing project:

1. Open Silk Performer.

   If you did not close the project you last worked on, Silk Performer will open with that project.
2. Close any open projects.

3 In the **Workflow** bar, click **Start here**.

The **Workflow - Outline Project** dialog box opens.



4 In the **Name** text box, enter **Bank** as the name for your load-testing project.

5 Enter an optional project description in **Description**.

6 In the **Type** list, select the **CORBA (IIOP)** option.

7 Click **OK**.

Silk Performer creates a new load-testing project called Bank with the default settings profile and your local computer as the only agent.

Adding a load-testing script

Each project contains a number of load-testing scripts. A test script defines different groups of virtual users and all the actions that these users have to perform. To be able to execute a load-testing script, you have to add it to your project.

Adding a test script

Procedure To add a predefined script to your project:

1 In the menu tree, right-click **Scripts** and select **Add Existing Script**.

The **Select Script(s)** dialog box opens.

2 Navigate to **<Public User Documents>\Silk Performer 16.5\Samples\CORBA\VisiBroker** and select the **VBrokerBank01.bdf** script.

3 Click **Open**.

Silk Performer adds the **VBrokerBank01.bdf** test script to the **Scripts** folder of your current project and displays the script in a new editor window.

Executing the test script

In the VBrokerBank01 test script, the VB_Bank01 user group is set up. Virtual users in this group will create a new account, query the current balance of the account, and write the balance to the output file.

Executing the test script

Procedure To execute the VBrokerBank01 test script:

1 Make sure the **VisiBroker Bank sample** server is running before you start this procedure, otherwise the script will fail.

2 In the **Active Script** window, expand the **User Groups** folder.

3 In the **User Groups** folder, right-click **VB_Bank01** and select **Run User "VB_Bank01"**.

Silk Performer runs a virtual user, performing all the actions defined in the test script. While the virtual user is being run, you can watch progress in the **Monitor** window, for example, the number of transactions executed, the response time of the last transaction, and the average response time.

4 Monitor the load test and wait until the virtual user has finished executing its tasks.

a In the top part of the **Monitor** window, select an agent computer or a user group that you want to monitor.

In the bottom part of the **Monitor** window, Silk Performer displays overview information about all the virtual users running on the selected agent computer or belonging to the selected user group.

b In the bottom part of the **Monitor** window, right-click a virtual user for which you want to view detailed information and select **Show Output**.

Silk Performer displays detailed run-time information about the selected user in the **Virtual User** window at the bottom, for example, the transactions and functions the user executes, and the data the user sends to and receives from the server. The type of information displayed depends on the options that you select at the top of the Monitor window.

Display Errors. Select this button to display an appropriate error message indicating the probable reason whenever a virtual-user-related error occurs.

Display Transactions. Select this button to display a message when a user has finished executing a transaction; the message indicates whether the transaction was successful or not.

Display Functions. Select this button to display a message for each function call a given user performs, including the function name and all its parameters.

Display Info. Select this button to display messages containing additional information about the current action a given user performs.

Display Data. Select this button to show data exchanged with the server.

Display all Errors of all Users. Select this button to display error messages for all errors of all users. Each error message will indicate user, agent and probable cause of the error.

This information will be displayed in addition to the information chosen for the selected user.

What you have learned

In Tutorial 1, you learned how to:

- Create a load-testing project
- Add a predefined script to the project
- Execute the script with one virtual user
- View progress information for the load test

Tutorial 2: Creating a script with the Recorder

The Silk Performer Recorder allows you to capture and record the IIOP traffic between the client CORBA object and the server objects. After you record traffic, you can save it as a test script. You can then easily customize the script and replay the script to simulate a large number of virtual users. Recording, customizing, and replaying a script will be covered separately in Tutorials 2, 3, and 4. In Tutorial 5, you will analyze the results of the load test you ran.

This tutorial includes the following steps:

- Setting up an application profile
- Recording traffic between a client application and a server
- Generating a test script based on the recorded traffic

- Trying out the generated test script
- Validating the generated test script

Setting up an application profile

Whenever you want the Silk Performer Recorder to capture and record IIOP traffic that is exchanged between CORBA objects, you have to set up a profile for the client object. This profile specifies the type of the client and what type of traffic to record.

If you are using VB for Java 3.X, you have to enter an additional application profile for the Java VM. (Application path will point to the local installation of the virtual machine).

Setting up an application profile

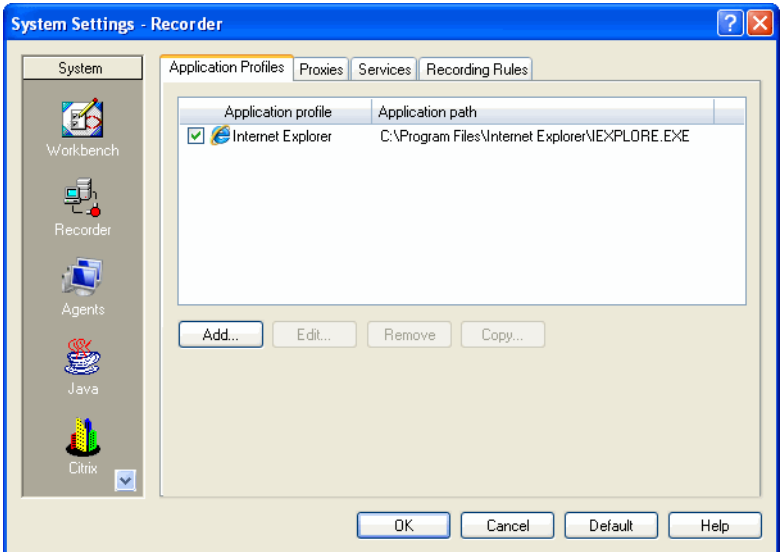
Procedure To set up a profile for the client:

1 From the menu bar, select **Settings/System**.

The **System Settings – Workbench** dialog box opens.

2 In the shortcut list on the left side, click the **Recorder** icon.

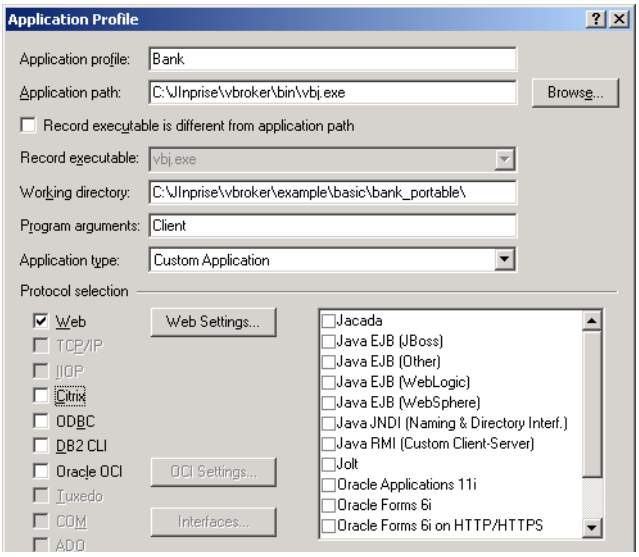
You see the following.



The list contains all the application profiles that are currently set up on your computer.

3 Click the **Add** button.

The **Application Profile** dialog box opens.



4 In the **Application profile** text box, enter a name for the application profile, for example, **Bank**.

5 Click **Browse** and locate the executable, called **VBJ.exe**. This file is located in the **BIN** subfolder of the VisiBroker home directory.

6 In the **Working directory** text box, enter the path of the sample you work with.

7 In the **Program arguments** text box, enter **Client**.

8 From the **Application type** list box, select the **Custom Application** option.

9 In the **API selection** area, select the **WinSock** option.

10 Click **Web Settings...**. The **Web Settings** dialog box opens.

11 From the list box, select **ws2_32.dll**. Click **OK**.

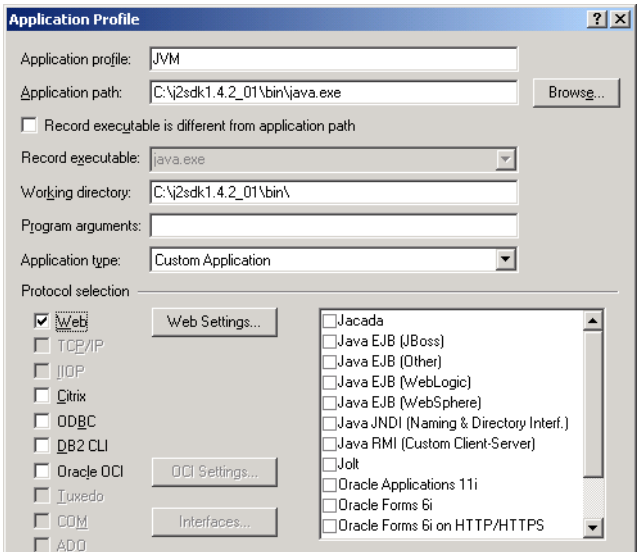
12 Click **OK**.

Silk Performer adds the new application profile to the profile list in the **System Settings – Recorder** dialog box.

Since VBJ launches a **Java Virtual Machine**, you also have to enter an application profile for **java.exe**.

13 Click the **Add** button.

The **Application Profile** dialog box opens.



14 In the **Application profile** text box, enter JVM for the application profile, for example, **JVM**.

15 Click **Browse** and in the folder of the Java virtual Machine locate and select **Java.exe**.

16 From the **Application type** list box, select the **Custom Application** option.

17 In the **API selection** area, select the **WinSock** option.

18 Click **Web Settings...**. The **Web Settings** dialog box opens.

19 From the list box, select **ws2_32.dll**. Click **OK**.

20 Click **OK**.

Silk Performer adds the new application profile to the profile list in the **System Settings – Recorder** dialog box.

21 Click **OK** to close the dialog box.

Modeling a test script

To load test the Bank application, the IIOP traffic between the client object and the server objects needs to be recorded and then described in a test script. The script finally enables any number of virtual users to perform actions similar to those that you performed during the recording session.

Modeling a test script

Procedure To model a load-testing script:

1. In the **Project** window, right-click **Project 'Bank'** and select **Add Data File**.

   The **Select Data File(s)** dialog box opens.

2. From the **Files of type** list box, select **CORBA IDL Files (*.idl)**.

3. Browse to the **Bank.idl** file and click **Add**.

   Silk Performer adds the Bank.idl file to your current project. The Recorder will require this file to interpret the traffic that is exchanged between the bank client and the server objects and to generate a high-level test script that can easily be customized.

4. In the **Workflow** bar, click **Model Script**.

   The **Workflow – Model Script** dialog box opens.

5. From the **Application Profile** list box, select **Bank**. Make sure the server is running!

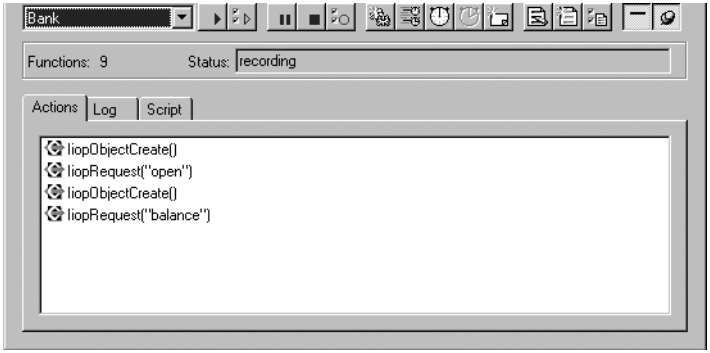
6. In **URL**, enter the command line parameter client (program argument in application profile).

7. Click **Start recording**.

   Silk Performer opens the Recorder and starts the Bank client application. The Recorder automatically records the IIOP traffic that the client application sends to and receives from the server objects.

8. To view the function calls that the Bank application performs, click the **Change GUI Size** button.

   The **Silk Performer Recorder** window displays as follows.



9. When the execution of the Bank client application is done, click the **Stop Recording** button in the Recorder.

   The **Save As** dialog box opens.

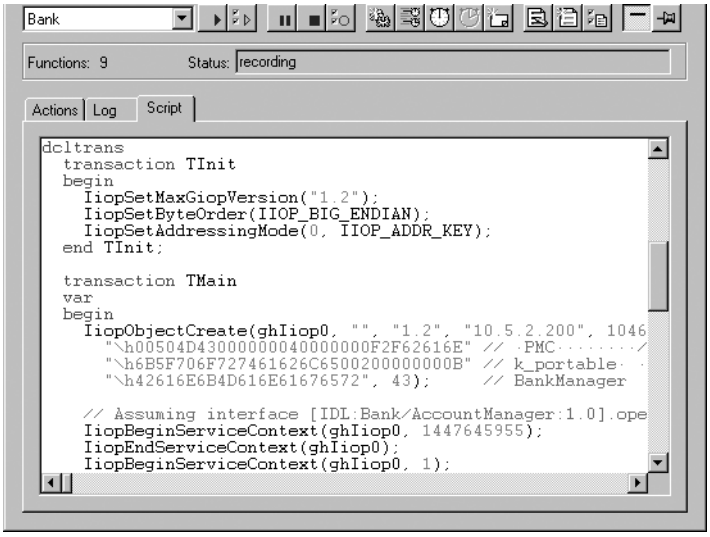
10 In the **File name** text box, enter **Bank**.

11 Click **Save**.

Silk Performer automatically adds the script to your current load-testing project.

12 You can now decide whether to close the Recorder.

If you keep it open, you can examine the script that has been generated.



13 Close the Recorder.

Trying out the generated test script

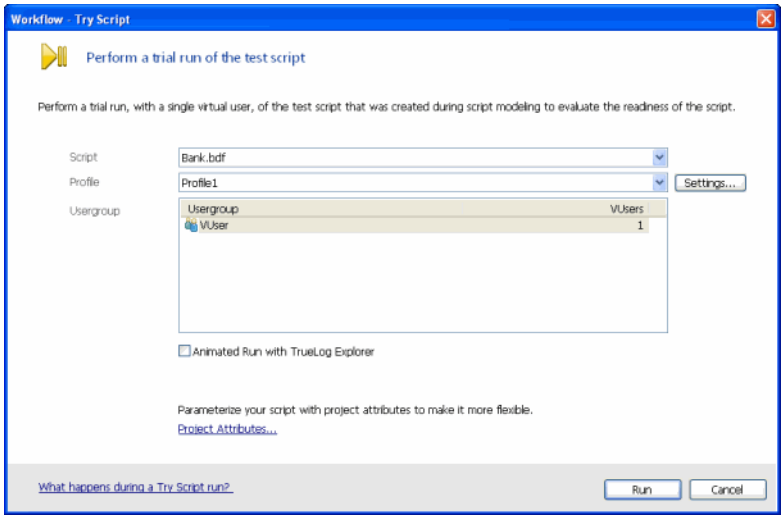
Next you should do a trial run of the test script that was created. The object of the trial run is to ensure that the script is free from error, and that it will reproduce accurately the interaction between the client and the server objects.

Trying out the script

Procedure To try out the generated test script:

1 In the **Workflow** bar, click **Try Script**.

The **Workflow – Try Script** dialog box opens.



2 From the **Script** list box, select the **Bank.bdf** script.

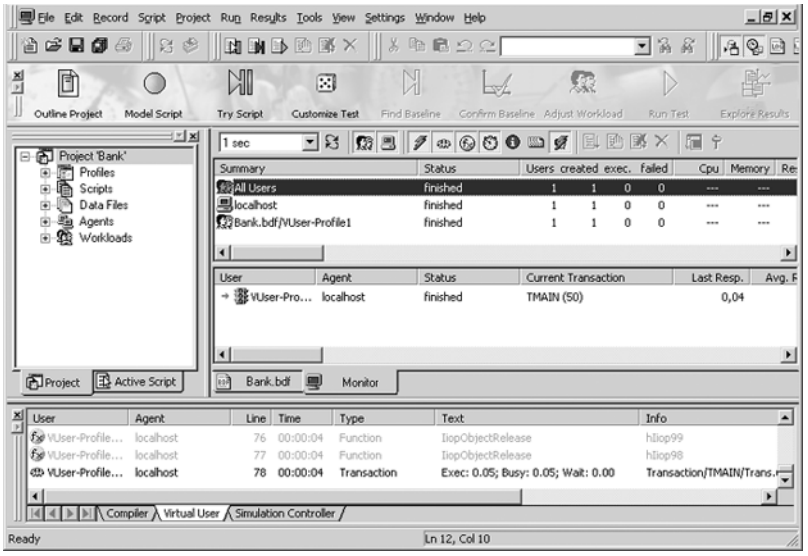
3 Make sure the **VUser** group is selected in the **Usergroup** list.

4 Disable the **Animated Run with TrueLog Explorer** option.

5 Click **Run**.

Silk Performer runs a single virtual user, called VUser, which performs all the actions that you have performed during the previous recording session.

While the single-user test is running, you can watch progress.



What you have learned

In Tutorial 2, you learned how to:

- Set up an application profile
- Record traffic between a client application and a server
- Generate a test script based on the recorded traffic
- Try out the generated test scripts

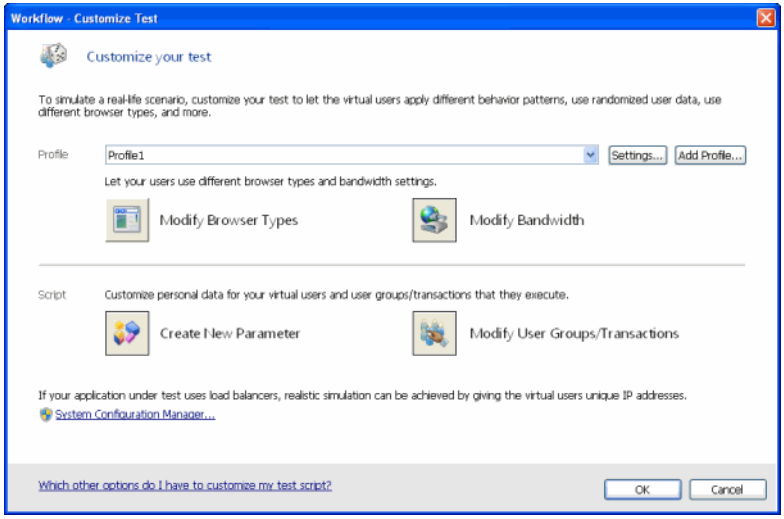
Tutorial 3: Customizing the generated test script

Where randomized data is necessary, random variables are employed to provide realistic user data. In this way, Silk Performer furnishes the virtual users with varied personal data – like name, address, phone number, and so on.

Replacing constant values

Procedure To replace constant values with random variables:

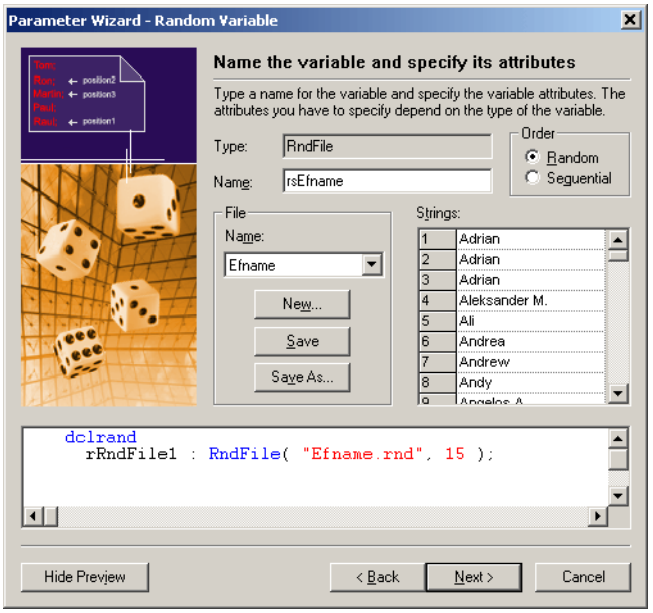
1. In the **Workflow** bar, click **Customize Test**.



2. Click the **Create New Parameter** icon.
   The **Parameter Wizard - Create New Parameter** dialog box opens.
3. Select **Parameter from Random Variable** and click **Next**.
   The **Parameter Wizard - Random Variable** dialog box opens.

4. From the **Random type** list box, select the **String from file** type.
5. Click **Next**.

The following dialog box opens.



6 In the **File** area, select **Efname** from the **Name** list box.

This random variable will draw from a Silk Performer file containing first names and randomly select a name up to 15 characters in length.

7 Click **Finish**.

Silk Performer inserts the following lines into your test script.

```
dclrand
  rsEfname : RndFile("Efname.rnd", 15);
```

8 Repeat steps 2-6, but instead of **Efname** select **Elname** from the **Name** list box in the Random Variable Wizard.

This random variable will draw from a Silk Performer file containing last names and randomly select a name up to 15 characters in length.

The **dclrand** section of the test script should now look like the following.

```
dclrand
  rsEfname : RndFile("Efname.rnd", 15);
  rsElname : RndFile("Elname.rnd", 12);
```

9 Locate the TMain transaction.

You can do this by expanding the **Transactions** folder in the **Active Script** window and double-clicking **TMain**.

10 Within the TMain transaction, declare a local string variable called **sName**.

You do this by inserting the following variable declaration after the **var** keyword.

```
sName: string;
```

11 At the beginning of the transaction, insert the following code.

```
sName := rsEfname + " " + rsElname;
```

This will generate a random name, consisting of a first and a last name, separated by a blank. The beginning of the TMain transaction should now look like the following. The code shown in bold type is what you have inserted.

```
transaction TMain
var
  sName: string;
begin
  sName := rsEfname + " " + rsElname;

  IiopObjectCreate(ghIiop0, "", "1.0",
                   "192.168.20.128", 1026,
    "\h01504D43000000001C00000049444C3A"
    "\h42616E6B2F4163636F756E744D616E61"
    "\h6765723A312E30000D00000042616E6B"
    "\h204D616E6167657200", 57);
```

12 Locate the section of the TMain transaction where a request for the “open” operation is sent.

This code section looks similar to the following.

```
IiopSetString(ghIiop0, "Default Name"); // name
IiopRequest(ghIiop0, "open");
```

13 Replace the name with the string variable **sName**.

The BDL code should now look like the following.

```
IiopSetString(ghIiop0, sName); // name
IiopRequest(ghIiop0, "open");
```

14 When you are done making changes to the script file, select **File/Save** from the menu bar to save your changes.

Specifying which tasks each user will perform

Usually a load test simulates a number of different types of users. In the test script, you have to specify which action each type of virtual user will perform.

Assigning tasks to users

Procedure To specify which tasks each user will perform:

1 Navigate to the **dcluser** section of the Bank.bdf script file. You can do this by selecting **User Groups/VUser** in the **Active Script** window.

You see the following.

```
dcluser
  user
    VUser
  transactions
    TInit           : begin;
    TMain           : 1;
```

In this section, a single user group is defined, called VUser. By default, this user group will perform the transactions that the Recorder created, TInit and TMain, once each.

2 Edit the **dcluser** section of the script so it looks like the following.

```
dcluser
  user
    Customer
  transactions
    TInit           : begin;
    TMain           : 50;
```

In the edited version of the script, each user of the Customer group performs the TInit transaction at the beginning and then the TMain transaction 50 times.

3 Select **File/Save** from the menu bar to save your changes.

4 In the **Workflow** toolbar, click the **Customize Test** button.

The **Workflow – Customize Test** dialog box opens.

5 Click **OK** to confirm that you are done customizing the test script.

What you have learned

In Tutorial 3, you learned how to:

- Replace constant values with random variables
- Specify which tasks each virtual user will perform

Tutorial 4: Replaying the customized script

After you have created a test script by recording traffic and then customizing the script, you can load test your server by replaying the script. To run a load test, you must perform the following steps:

- Find the test baseline
- Confirm the test baseline
- Set up server monitoring for your load test
- Execute the load test

Finding the test baseline

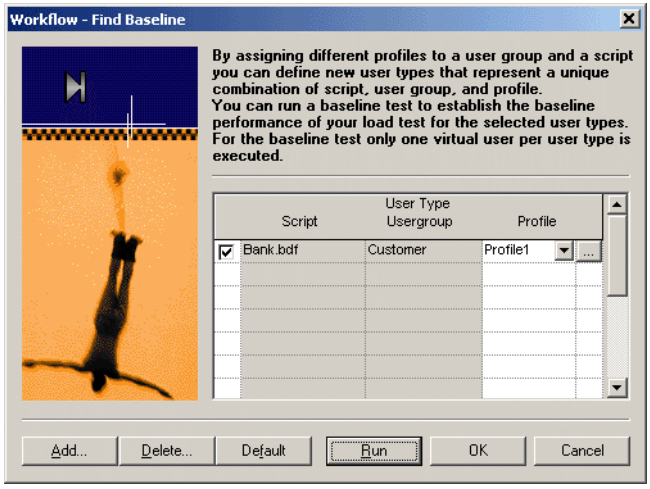
After you have customized the load-testing script, you should ascertain the baseline performance – that is, the basic, ideal performance – of the server objects under test. The fully customized test script is run with just one user per user group, and the results from this unstressed performance from the server objects form the basis for comparison with the results of the full load test later.

Finding the baseline

Procedure To find the load test baseline:

1. In the **Workflow** toolbar, click the **Find Baseline** button.

   The **Workflow – Find Baseline** dialog box opens.



   This dialog box lists all the user groups you have set up in your test script.

2. Click **Run**.

   Silk Performer runs one virtual user from each user group that you have declared in the test script. While the test is running, you can watch progress in the **Monitor** window.

Confirming the test baseline

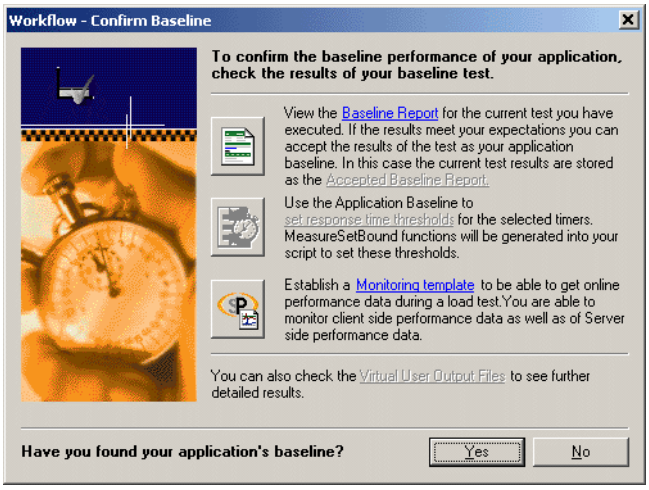
After this test run, you have to confirm that the test baseline does actually reflect the desired performance by the application under test. This is done by inspecting the results from that test. If the results are satisfactory, the baseline established will form the basis for comparison with the results from the full load test later.

Confirming the baseline

Procedure To confirm the load test baseline:

1 In the **Workflow** toolbar, click the **Confirm Baseline** button.

The **Workflow – Confirm Baseline** dialog box opens.



2 Click the **Baseline Report** button.

The Baseline Report opens.

3 Check the report carefully. In particular, make sure that no errors have occurred during your baseline test.

4 Click the **Accept Baseline** button in the Baseline Report.

5 Click **Yes** and **OK** by the Message boxes that comes up.

6 If your baseline report indicates no errors, click **Yes** in the **Workflow – Confirm Baseline** dialog box to confirm that you have found the baseline.

Setting up server monitoring for your load test

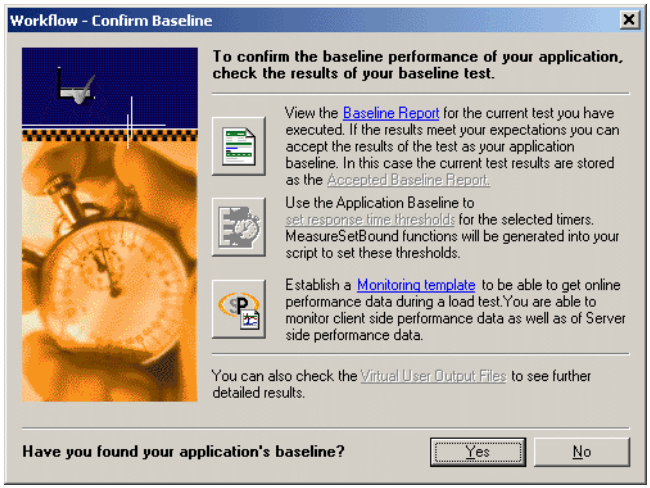
Monitoring the server under test helps you to find out whether there are bottlenecks on the server objects, and, if so, where exactly they are.

Setting up server monitoring

Procedure To set up server monitoring for your load test:

1 In the **Workflow** toolbar, once again click the **Confirm Baseline** button.

The **Workflow – Confirm Baseline** dialog box opens.



2 Click the **Monitoring template** button.

The **Profile - [Profile1] - Results** dialog box opens to the **Monitoring** tab.

The option **Automatically start monitoring** is activated by default. It means, that when you run a load test, the Performance Explorer will open automatically using a default template that invokes a display of relevant client information.

3 If you want to customize monitoring, enable the **Use custom monitoring template**, and click the **Create Customer Monitor Template** button.

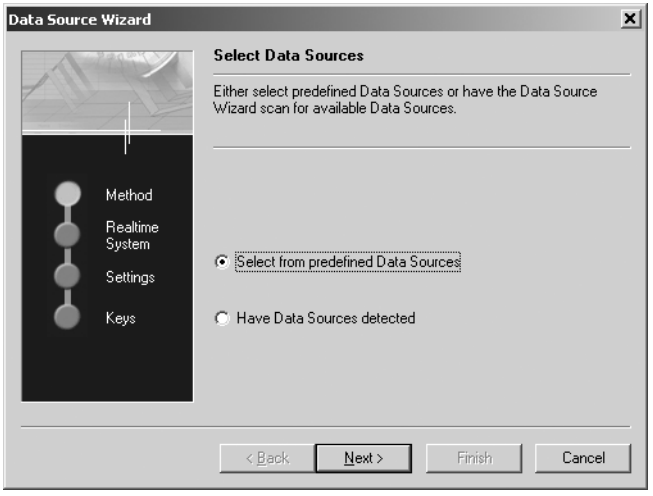
Performance Explorer will create and open a template with the name of your project, which is equal to the default template.

4 Click **Edit Customer Monitor Template,** to customize the new template and change the display of information. You can close or maintain the default monitoring windows and you can add one or more windows to provide additional information.

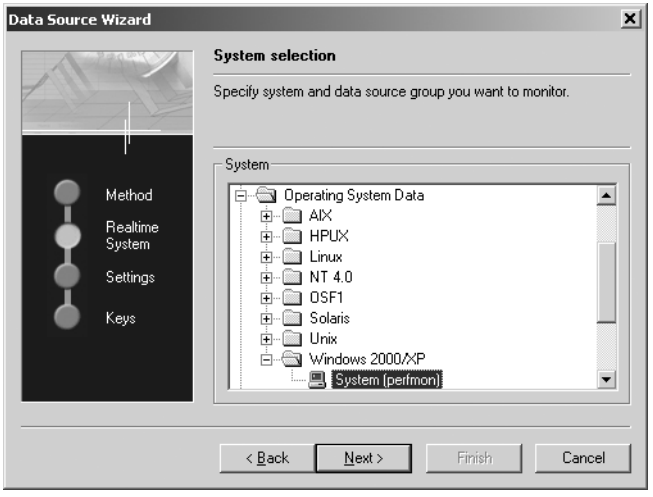
To set up Performance Explorer to display additional information, proceed as follows.

a Click the **Monitor Server** button in the Performance Explorer workflow bar

The **Data Source Wizard** opens.



b Click **Next**.
The **Data Source Wizard - System selection** dialog box opens.



c In the tree view, expand the folder that corresponds to the operating system on which the Bank server is running. Within the folder, select **System**.

d Click **Next**.

e In the next dialog box, enter connection parameters, like the host name or IP address of the server, the connection port, the user name, and the password.

The data you have to enter here depends on the operating system that is running on the computer you are monitoring.

f Click **Next** when you are done entering connection parameters.

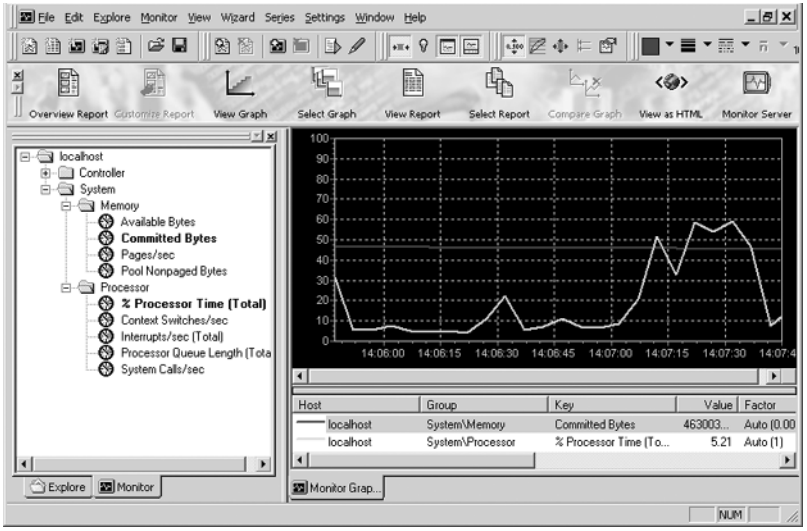
g In the next dialog box, select the performance counters you want to monitor.

Of particular interest are the processor and memory utilization of the server being tested.

h Click **Finish**.

The Performance Explorer opens a new window and displays real-time data for the performance counters you have selected.

Later, when you are running the load test, the monitor view will look similar to the following.



Adjusting the workload

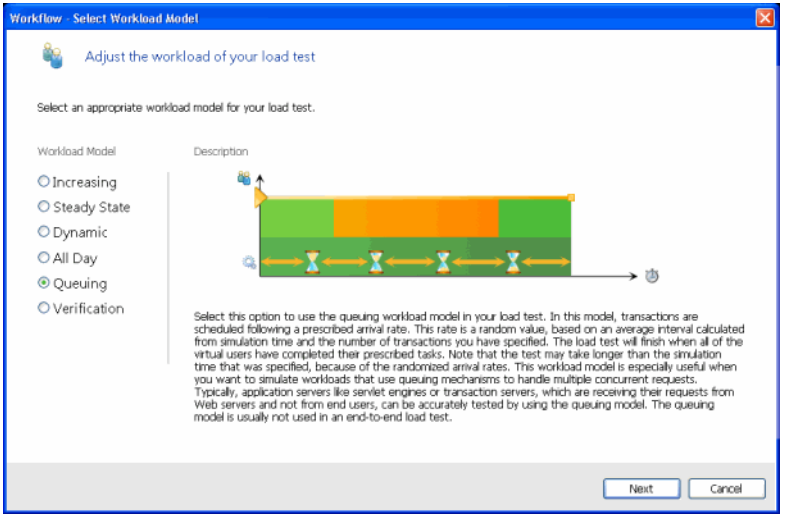
After you have confirmed your baseline, you now select your workload model and prepare the workload to run a full load test.

Adjusting the workload

Procedure To adjust the workload:

1 In the **Workflow** toolbar, click **Adjust Workload**.

The **Workflow - Select Workload Model** dialog box opens.



This dialog box shows all possible workload models, which can be selected.

2 Select **Queuing** workload model.

3 Click **Next**.

4 On the **Workflow Assign Agents** dialog box, select an option and click **OK**.

Silk Performer now opens the Workload Configuration dialog box.

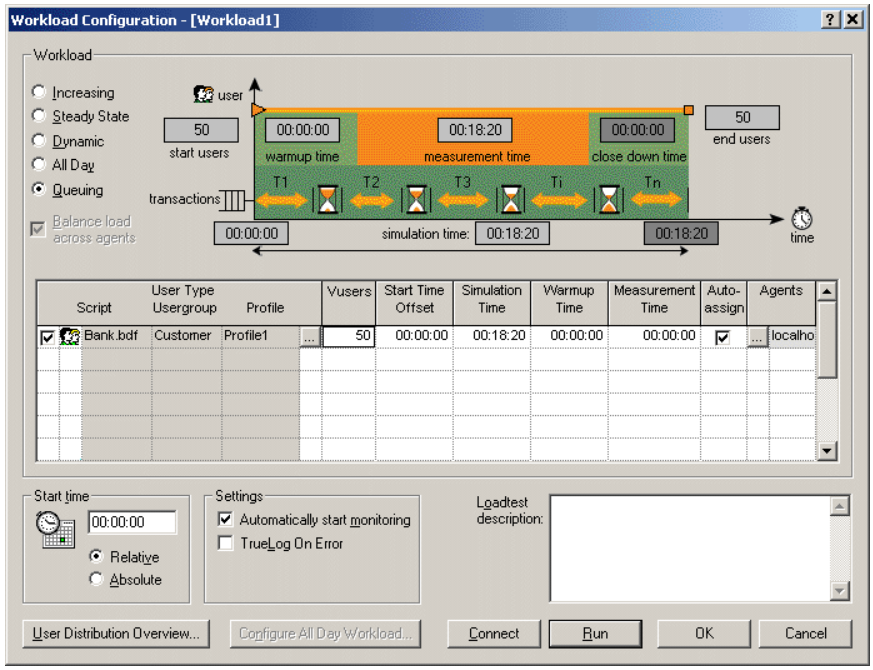
Running the load test

After you have adjust the workload, you run the full load test. The test script is run with multiple virtual users in order to test the Bank server. A large load test would need the appropriate testing environment to be set up in the local area network, including a full complement of agent computers to host the virtual users. For this tutorial, however, you use only your local computer.

Running the test

Procedure To run the load test:

1 The **Workload Configuration** dialog box is open.



2 In the **VUsers** column, change the number of Customer users to **50**.

3 For the Customer user group, change the simulation time to **300** seconds (5 minutes).

4 Disable all user groups that are declared in the VBrokerBank01 test script.

5 Click **Run**.

Silk Performer runs a load test with 50 virtual users, load testing the server application.

Performance Explorer opens automatically and you can watch progress and the server being tested in the **Monitor** window.

What you have learned

In Tutorial 4, you learned how to:

- Find the test baseline
- Confirm the test baseline
- Set up server monitoring for your load test
- Adjusting the workload
- Execute the load test

Tutorial 5: Viewing the results of your load test

Silk Performer incorporates **Performance Explorer**, a powerful graphing and analysis tool to help you work with your results information.

Performance Explorer provides a comprehensive array of results information, which can be displayed using advanced features for creating statistical reports and displaying performance results in real time generated graphics. Results information may vary, depending on the type of application and/or server being tested. In general it falls into the following categories:

- Response time information: This is the total time to process a given timer; it provides application performance information from the point of view of the client.
- Throughput information: This is the rate at which a given timer is processed on average by the server; it allows you to analyze performance from the point of view of the server.

Overview report of performed measurements

You can consult an HTML overview report that contains results of all performed measurements.

Exploring overview report

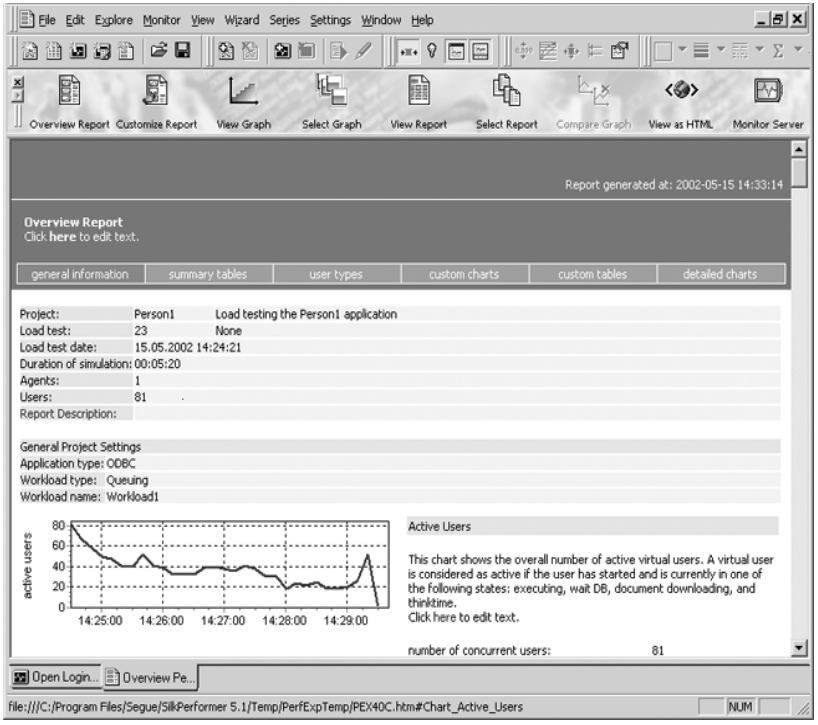
Procedure To explore overview report:

1 In the **Workflow** toolbar, click the **Explore Results** button.

The **Workflow - Explore Results** dialog box opens.

2 Click the **Performance Explorer** button.

Performance Explorer opens, and the HTML overview report is generated.



You may consult this report which includes short explanations of the displayed graphs. If you want you can view more detailed information.

Detailed response time measurements

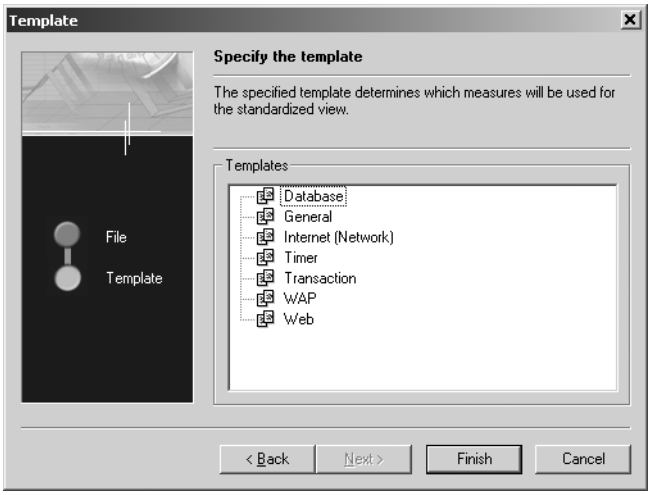
When you analyze performance of an application from the client point of view, you examine the response times for all the transactions and individual timers. This will enable you to find out how users will experience their interaction with server objects. A common goal in load testing is to find out if response times for all transactions remain below a specified, critical limit.

Response time details

Procedure To explore detailed response time measurements:

1 In the Performance Explorer **Workflow** bar, click the **Select Graph** button.

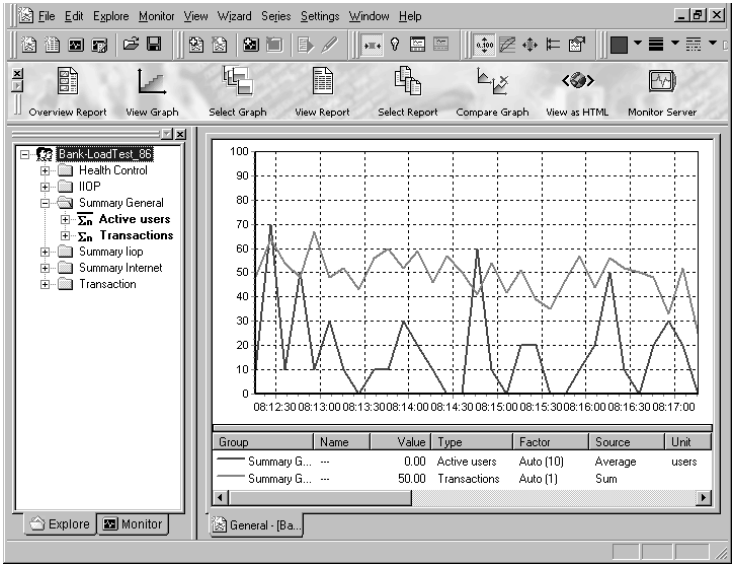
The **Template** dialog box opens.



2 In the **Templates** list, select the **Timer** option.

3 Click **Finish**.

Performance Explorer opens a new chart with the number of active virtual users displayed.



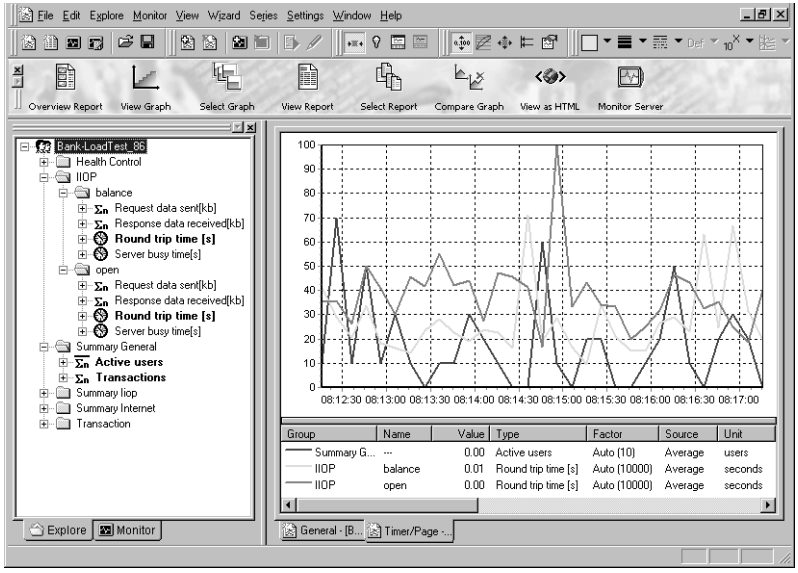
You see the default graph with the average number of active users, and one or more default measurements (if any).

4 Expand the **Explore** tree view.

5 In the **Explore** tree view, select **IIOP/balance**, drag the **Round trip time** measurement, and drop it on the graph.

6 In the **Explore** tree view, select **IIOP/open**, drag the **Round trip time** measurement, and drop it on the graph.

Your graph should now look like the following.



The load test that you ran in the previous tutorial lasted only a few minutes, so there are only a few measurements, which makes it difficult to draw any conclusions.

In real-life you will run load tests that last several hours, a few days, or even a week. These tests will provide results information that will allow you analyze the performance of your server accurately.

Exploring throughput measurements

When you analyze the performance of an application from the point of view of the server, you examine the throughput rate. Throughput is the work done by the server within a specific time interval, for example, the number of requests that the server objects process within a second, a minute, or an hour. A common goal in load testing is to ensure that the throughput rate is above a specified, critical limit.

Throughput rate details

Procedure To explore throughput measurements:

1. From the menu bar, select **File/New/Graph**.

   The Performance Explorer opens a new, blank graph.

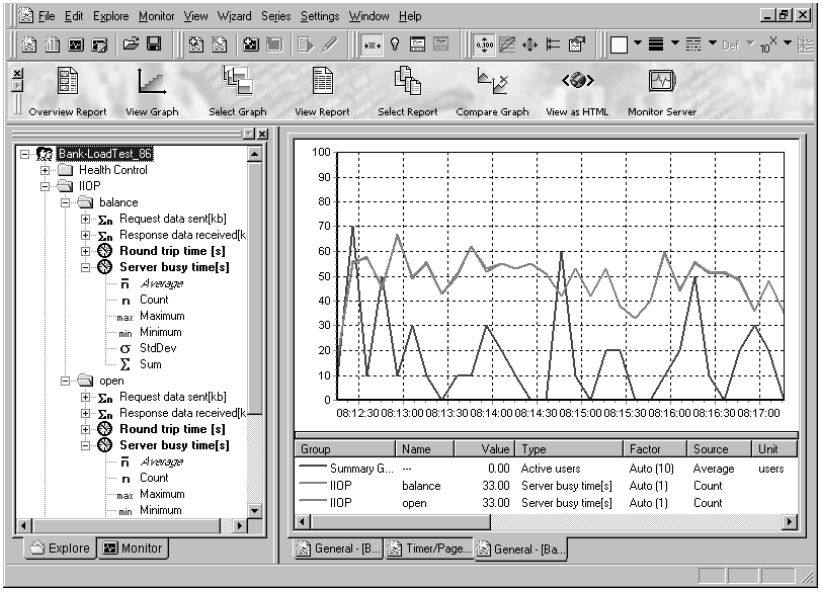
2. In the **Explore** tree view, expand both the **Summary General** and the **IIOP** measurement groups.

3. In the **Summary General** measurement group, drag the **Active users** measurement and drop it on the graph.

4. In the **IIOP** measurement group, select **balance/Server busy time**, drag the **Count** measurement, and drop it on the graph.

5. In the **IIOP** measurement group, select **open/Server busy time**, drag the **Count** measurement, and drop it on the graph.

   You see a graph that displays the number of operations performed by the server objects. The graph should look similar to the following.



Again, since the load test that you ran lasted only five minutes, it is difficult to derive conclusions from the results.

What you have learned

In Tutorial 5, you learned how to:

- Explore response time measurements
- Explore throughput measurements

6

Load Testing a TUXEDO Application

Introduction

This chapter contains a series of tutorials that will help you to start load-testing TUXEDO applications with Silk Performer.

What you will learn

This chapter contains the following sections:

Overview

Each of the following tutorials provides step-by-step instructions for important Silk Performer tasks. You should perform the tutorials in order, since each tutorial builds on what you have learned in the previous ones.

You will learn how to do the following:

- Execute a predefined load test on your local machine.
- Create a test script by recording function calls to the TUXEDO System using the Recorder.
- Customize the recorded test script.

- Run the customized script, using multiple virtual users.
- Use reporting tools to view throughput and response time information.

Sample: Bank application

These tutorials use a sample TUXEDO System application called bankapp. This application is included on your BEA TUXEDO System CD-ROM and should have been installed in the apps\bankapp subfolder of the TUXEDO System home directory as part of the installation and setup.

Tutorial 1: Running a sample load test

In this tutorial you will learn to create a load-testing project and execute sample scripts that are provided with Silk Performer. You will run one of the sample scripts with a single virtual user on your local machine.

You will use the test script bankappn.bdf, which is included in the **<Public User Documents>\Silk Performer 16.5\Samples\Middleware\TUXEDO** folder.

Creating a load-testing project

For each server you load test with Silk Performer, you have to create a project file. A project file administers all the settings that are associated with a series of load tests, like the test type, the workload model, and a number of options that depend on the type of server being tested.

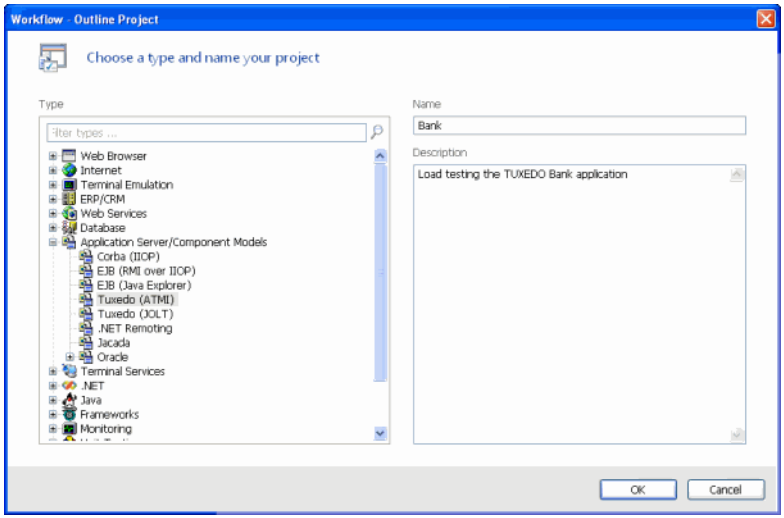
Creating a project

Procedure To create your load-testing project:

1. Open **Silk Performer.**

   If you did not close the project you last worked on, Silk Performer will open with that project.
2. Close any open projects.
3. In the **Workflow** bar, click **Start here**.

The **Workflow - Outline Project** dialog box opens.



4 In the **Name** text box, enter **Bank** as the name for your load-testing project.

5 Enter an optional project description in **Description**.

6 In the **Type** list, select **Tuxedo**.

7 Click **OK**.

Silk Performer creates a new load-testing project called Bank with the default settings profile and your local computer as the only agent.

Adding a load-testing script

Each project contains a number of load-testing scripts. A test script defines different groups of virtual users and all the actions that these users have to perform. To be able to execute a load-testing script, you have to add it to your project.

Adding a test script

Procedure To add a predefined script to your project:

1 In the menu tree, right-click **Scripts** and select **Add Existing Script**.

The **Select Script(s)** dialog box opens.

2 Navigate to **<Public User Documents>\Silk Performer 16.5\Samples\Middleware\TUXEDO** and select the **bankappn.bdf** script.

3 Click **Open**.

Silk Performer adds the **bankappn.bdf** test script to the **Scripts** folder of your current project and displays the script in a new editor window.

Personalizing the sample script

The bankappn script contains three constant declarations that specify the directory where the TUXEDO System client software is installed on the agent computer and the address of the Bank application server. Before you execute the script, you have to change the constant values with your personal settings.

Personalizing the test

Procedure To personalize the sample load-testing script:

1. In the **Active Script** window, select **Symbols/Constants/TUXDIR**.

   You see the following code section.

```
const
  TUXDIR    := "d:\\tuxedo";              // TUXEDO environment setting
  WSNADDR   := "//lab5:3000";             // host/port where WSL listens
  FLDTBLDIR := "d:\\tuxedo\\apps\\allserv"; // location of field tables
```

2. Replace the directory “d:\\tuxedo” with the directory where the TUXEDO System client software is installed on your local computer.
3. Replace the address “//lab5:3000” with the address of the Bank application server.
4. Replace the first part of the FLDTBLDIR constant (“d:\\tuxedo”) with the directory where the TUXEDO System client software is installed on your local computer.
5. Select **File/Save** from the menu bar to save your changes.

Executing the test script

In the bankappn test script, the TuxBankUsr user group is set up. Virtual users in this group will log on to the TUXEDO System, make inquiries about their bank account, deposit money into and withdraw money from the account, and log off from the TUXEDO System.

Executing the test script

Procedure To execute the bankappn test script:

1. In the **Active Script** window, expand the **User Groups** folder.
2. In the **User Groups** folder, right-click **TuxBankUsr** and select **Run User “TuxBankUsr”**.

Silk Performer runs a virtual user, performing all the actions defined in the test script. While the virtual user is being run, you can watch progress in the **Monitor** window, for example, the number of transactions executed, the response time of the last transaction, and the average response time.

3 Monitor the load test and wait until the virtual user has finished executing its tasks.

a In the top part of the **Monitor** window, select an agent computer or a user group that you want to monitor.

In the bottom part of the **Monitor** window, Silk Performer displays overview information about all the virtual users running on the selected agent computer or belonging to the selected user group.

b In the bottom part of the **Monitor** window, right-click a virtual user for which you want to view detailed information and select **Show Output**.

Silk Performer displays detailed run-time information about the selected user in the **Virtual User** window at the bottom, for example, the transactions and functions the user executes, and the data the user sends to and receives from the server. The type of information displayed depends on the options that you select at the top of the Monitor window.

Display Errors. Select this button to display an appropriate error message indicating the probable reason whenever a user-related error occurs.

Display Transactions. Select this button to display a message when a user has finished executing a transaction; the message indicates whether the transaction was successful or not.

Display Functions. Select this button to display a message for each function call a given user performs, including the function name and all its parameters.

Display Info. Select this button to display messages containing additional information about the current action a given user performs.

Display Data. Select this button to show data exchanged with the server.

Display all Errors of all Users. Select this button to display error messages for all errors of all users. Each error message will indicate user, agent and probable cause of the error.

This information will be displayed in addition to the information chosen for the selected user.

What you have learned

In Tutorial 1, you learned how to:

- Create a load-testing project
- Add a predefined script to the project
- Execute the script with one virtual user
- View progress information for the load test

Tutorial 2: Creating a script with the Recorder

The Silk Performer Recorder allows you to capture function calls that a TUXEDO client application performs. After you record function calls, you can save them as a test script. You can then easily customize the script and replay the script to simulate a large number of virtual users. Recording, customizing, and replaying a script will be covered separately in Tutorials 2, 3, and 4. In Tutorial 5, you will analyze the results of the load test you ran.

This tutorial includes the following steps:

- Setting up an application profile
- Recording the function calls that the client application performs
- Generating a test script based on the recorded traffic
- Trying out the generated test script
- Validating the generated test script

Setting up an application profile

Whenever you want the Silk Performer Recorder to capture and record the function calls that a client application performs, you have to set up a profile for the client application. This profile specifies the type of the client application and what type of function calls to record.

Setting up an application profile

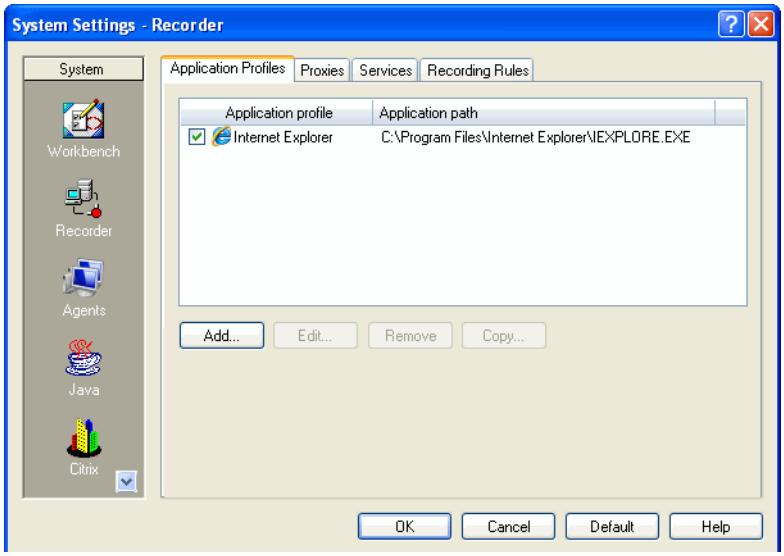
Procedure To set up a profile for the client application:

1. From the menu bar, select **Settings/System**.

   The **System Settings – Workbench** dialog box opens.

2. In the shortcut list on the left side, click the **Recorder** icon.

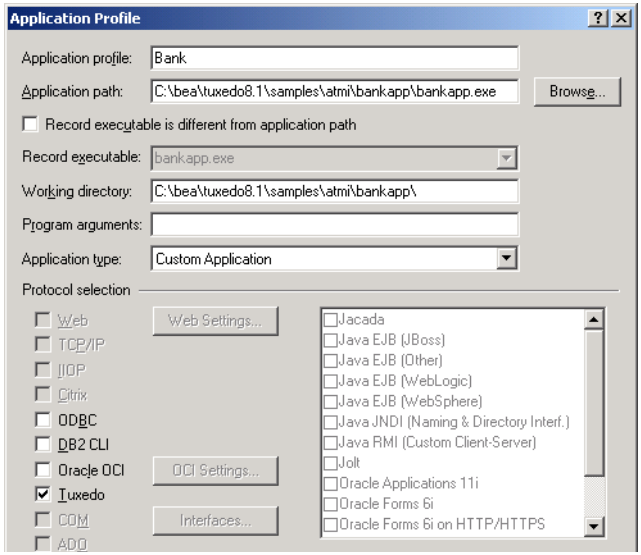
You see the following.



The list contains all the application profiles that are currently set up on your computer.

3 Click **Add**.

The **Application Profile** dialog box opens.



4 In the **Application profile** text box, enter a name for the application profile, for example **Bank**.

5 Click **Browse** and locate the executable, called bankappn.exe. This file is located in the apps\bankapp subfolder of the TUXEDO System home directory.

6 From the **Application type** list box, select the **Custom Application** option.

7 In the **API selection** area, select the **Tuxedo** option.

8 Click **OK**.

Silk Performer adds the new application profile to the profile list in the **System Settings – Recorder** dialog box.

9 Click **OK** to close the dialog box.

Modeling a test script

To load test the Bank server, the function calls that the client application performs need to be recorded and then described in a test script. The script finally enables any number of virtual users to perform actions similar to those that you performed during the recording session.

In order to create as realistic a load test as possible, you will record four separate transactions in a single recording session:

- Logging on to the Bank server
- Opening a new account
- Querying the balance of an existing account
- Logging off from the Bank server

When you run the load test, you will create two user groups to which you will assign each of these tasks in a ratio that simulates real-world behavior.

Modeling a test script

Procedure To model a load-testing script:

1 In the **Workflow** bar, click **Model Script**.

The **Workflow – Model Script** dialog box opens.

2 From the **Application Profile** list box, select **Bank**.

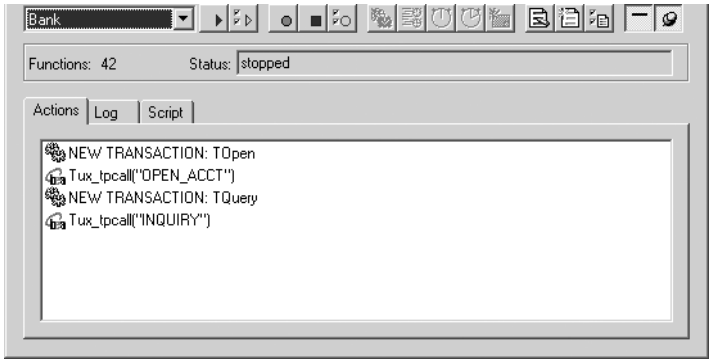
3 Click **Start recording**.

Silk Performer opens the Recorder and starts the Bank client application.

As you connect to the Bank server, the Recorder automatically records the function calls that the client application performs.

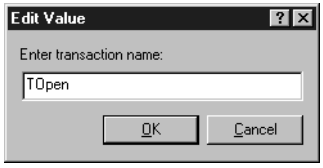
4 To view your actions as well as the function calls that the Bank application performs, click the **Change GUI Size** button.

The **Silk Performer Recorder** window displays as follows.



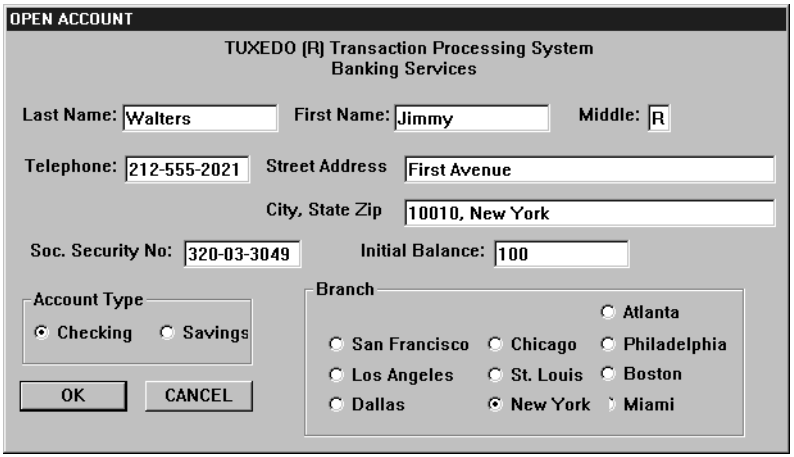
5 In the Recorder, click the **New Transaction** button to insert a new transaction into the test script you are generating.

The following dialog box opens.



6 Create a new transaction called **TOpen**; then click **OK**. A distinct set of time measurements will be made for each transaction you create. When recording traffic, you should create a new transaction for each distinct user session, from connection to shutdown.

7 In the Bank application, open a new account. Enter the necessary information as shown in the following dialog box; then click **OK**.



8 In the Recorder, create a new transaction called **TQuery**.

9 In the Bank application, make an inquiry about an existing account.

10 Close the Bank application.

11 In the Recorder, click the **Stop Recording** button.

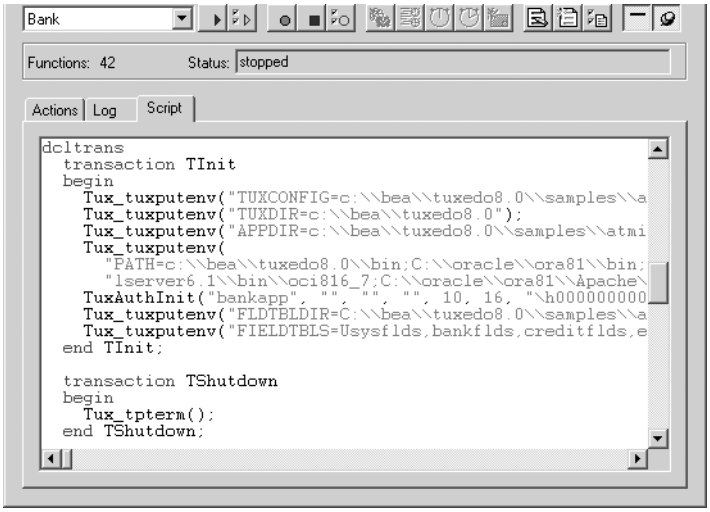
The **Save As** dialog box opens.

12 In the **File name** text box, enter **Bank**.

13 Click **Save**.

14 You can now decide whether to close the Recorder.

If you keep it open, you can examine the script that has been generated.



15 Close the Recorder.

Silk Performer automatically adds the script to your current load-testing project.

Trying out the generated test script

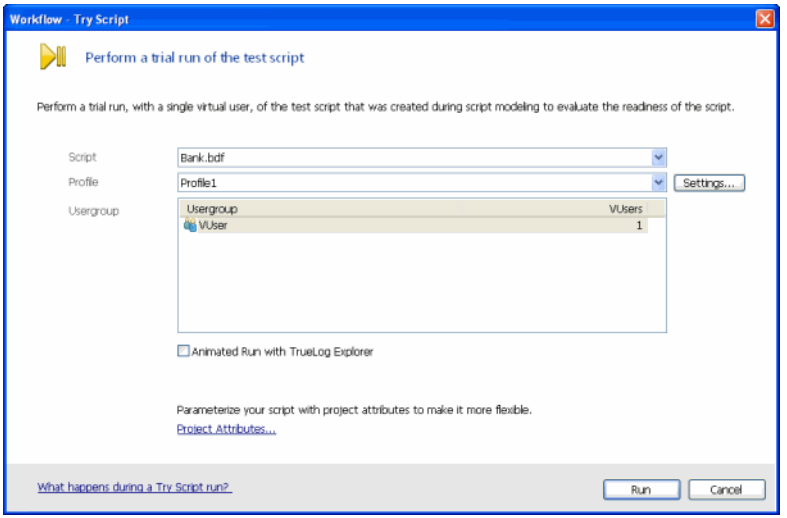
Next you should do a trial run of the test script that was created. The object of the trial run is to ensure that the script is free from error, and that it will reproduce accurately the interaction between the client application and the Bank server.

Trying out the script

Procedure To try out the generated test script:

1 In the **Workflow** bar, click the **Try Script** button.

The **Workflow – Try Script** dialog box opens.



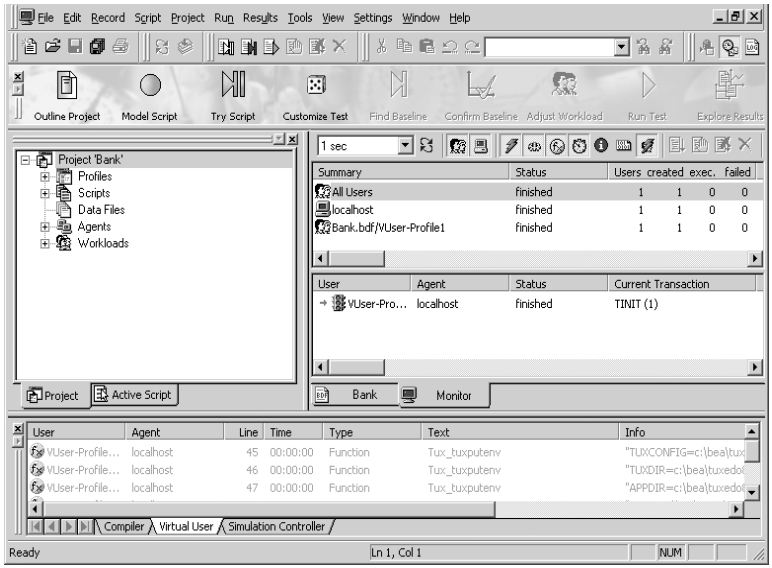
2 From the **Script** list box, select the **Bank.bdf** script.

3 Make sure the **VUser** group is selected in the **Usergroup** list.

4 Disable the **Animated Run with TrueLog Explorer** option.

5 Click **Run**.

Silk Performer runs a single virtual user, called VUser, which performs all the actions that you have performed during the previous recording session. While the single-user test is running, you can watch the progress.



What you have learned

In Tutorial 2, you learned how to:

- Set up an application profile
- Record traffic between a client application and a server
- Generate a test script based on the recorded traffic
- Try out the generated test scripts

Tutorial 3: Customizing the generated test script

Customizing the test script you have generated allows you to take the actions performed by a single user and create from them a realistic simulation of multiple users. Two key tasks can help you create a real-world simulation:

- Replacing constant values with random variables
- Specifying which tasks each virtual user will perform

Note You must use Benchmark Description Language (BDL), Silk Performer's high-level scripting language, to customize generated test scripts. Although knowledge of BDL is not required for this

tutorial, it might be helpful to browse the introductory BDL Reference topics in the Silk Performer Online Help for more information about BDL before you begin.

Replacing constant values with random variables

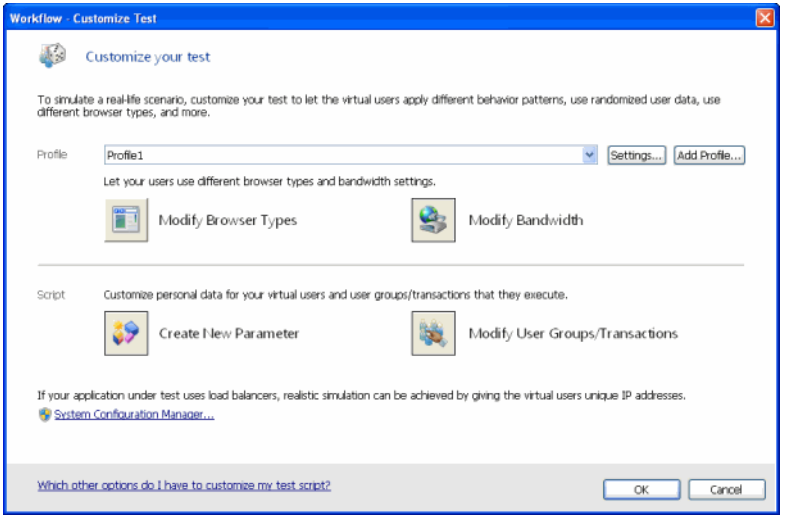
Where randomized data is necessary, random variables are employed to provide realistic user data. In this way, Silk Performer furnishes the virtual users with varied personal data – like name, address, phone number, and so on.

Replacing constant values

Procedure To replace constant values with random variables:

1 In the **Workflow** bar, click **Customize Test**.

The **Workflow – Customize Test** dialog box opens.



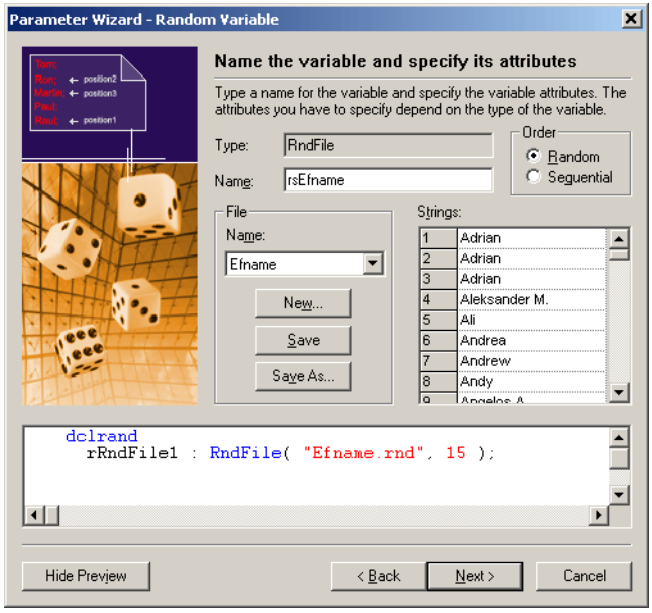
2 Click the **Create New Parameter** icon.
The **Parameter Wizard - Create New Parameter** dialog box opens.

3 Select **Parameter from Random Variable** and click **Next**.
The **Parameter Wizard - Random Variable** dialog box opens.

4 From the **Random type** list box, select the **String from file** type.

5 Click **Next**.

The following dialog box opens.



6 In the **File** area, select **Efname** from the **Name** list box.

This random variable will draw from a Silk Performer file containing first names and randomly select a name up to 15 characters in length.

7 Click **Finish**.

Silk Performer inserts the following lines into your test script.

```
dclrand
  rsEfname : RndFile("Efname.rnd", 15);
```

8 Repeat steps 1-6, but instead of **Efname** select **Elname** from the **Name** list box in the Random Variable Wizard.

This random variable will draw from a Silk Performer file containing last names and randomly select a name up to 15 characters in length.

The **dclrand** section of the test script should now look like the following.

```
dclrand
  rsEfname : RndFile("Efname.rnd", 15);
  rsElname : RndFile("Elname.rnd", 15);
```

9 Locate the TOpen transaction.

You can do this by expanding the **Transactions** folder in the **Active Script** window and double-clicking **TOpen**.

10 Within the TOpen transaction, locate the section where the parameters for the OPEN_ACCT service call are copied into a typed buffer.

The section looks similar to the following.

```
Tux_tpbegin(30, 0);
TuxGetBuffer(hBuffer0, "FML", NULL, 1024);
TuxSetInt(hBuffer0, ID_8296, 1);
TuxSetInt(hBuffer0, ID_16496, ord('C'));
TuxSetInt(hBuffer0, ID_16499, ord('R'));
TuxSetString(hBuffer0, ID_41068, "212-555-0293");
TuxSetString(hBuffer0, ID_41069, "First Avenue     New York, NY 10010");
TuxSetString(hBuffer0, ID_41071, "345-23-1092");
TuxSetString(hBuffer0, ID_41073, "Walters");
TuxSetString(hBuffer0, ID_41074, "Jimmy");
TuxSetString(hBuffer0, ID_41162, "1000.00");
Tux_tpcall("OPEN_ACCT", hBuffer0, 0, hBuffer0, olen, TPNOFLAGS);
```

11 Replace the first name that you have entered during the recording session with the **rsEfname** random variable.

12 Replace the last name that you have entered during the recording session with the **rsElname** random variable.

13 Immediately before the Tux_tpcall function call, insert the following function call.

```
MeasureStart("Open Account");
```

This will start a time measurement for the TUXEDO service call.

14 Immediately after the Tux_tpcall function call, insert the following function call.

```
MeasureStop("Open Account");
```

This will stop the time measurement for the TUXEDO service call. The section should now look like the following. The code shown in bold type is what you have changed.

```
Tux_tpbegin(30, 0);
TuxGetBuffer(hBuffer0, "FML", NULL, 1024);
TuxSetInt(hBuffer0, ID_8296, 1);
TuxSetInt(hBuffer0, ID_16496, ord('C'));
TuxSetInt(hBuffer0, ID_16499, ord('R'));
TuxSetString(hBuffer0, ID_41068, "212-555-0293");
TuxSetString(hBuffer0, ID_41069, "First Avenue     New York, NY 10010");
TuxSetString(hBuffer0, ID_41071, "345-23-1092");
TuxSetString(hBuffer0, ID_41073, rsElname);
TuxSetString(hBuffer0, ID_41074, rsEfname);
TuxSetString(hBuffer0, ID_41162, "1000.00");
MeasureStart("Open Account");
Tux_tpcall("OPEN_ACCT", hBuffer0, 0, hBuffer0, olen, TPNOFLAGS);
MeasureStop("Open Account");
```

15 Locate the TQuery transaction.

You can do this by expanding the **Transactions** folder in the **Active Script** window and double-clicking **TQuery**.

16 In this transaction, immediately before the Tux_tpcall function call, insert the following function call.

```
MeasureStart("Query Balance");
```

This will start a time measurement for the TUXEDO service call.

17 Immediately after the Tux_tpcall function call, insert the following function call.

```
MeasureStop("Open Account");
```

This will stop the time measurement for the TUXEDO service call. The section should now look like the following.

```
Tux_tpbegin(30, 0);
TuxGetBuffer(hBuffer0, "FML", NULL, 4096);
TuxSetInt(hBuffer0, ID_8302, 10023);
MeasureStart("Query Balance");
Tux_tpcall("INQUIRY", hBuffer0, 0, hBuffer0, olen, TPNOFLAGS);
MeasureStop("Query Balance");
//TuxGetInt(hBuffer0, ID_8302); // 10023
//TuxGetString(hBuffer0, ID_40966, ostring, sizeof(ostring));
//TuxGetString(hBuffer0, ID_41161, ostring, sizeof(ostring));
Tux_tpcommit(0);
```

18 When you are done making changes to the script file, select **File/Save** from the menu bar to save your changes.

Specifying which tasks each user will perform

Usually a load test simulates a number of different types of users. In the test script, you have to specify which action each type of virtual user will perform.

Assigning tasks to users

Procedure To specify which tasks each user will perform:

1. Navigate to the **dcluser** section of the Bank.bdf script file. You can do this by selecting **User Groups/VUser** in the **Active Script** window.

   You see the following.

   ```
   dcluser
     user
       VUser
     transactions
       TInit           : begin;
       TOpen           : 1;
       TQuery          : 1;
       TShutdown       : end;
   ```

   In this section, a single user group is defined, called VUser. By default, this user group will perform the transactions you created, TInit, TOpen, TQuery, and TShutdown, once each.

2. Edit the **dcluser** section of the script so it looks like the following.

   ```
   dcluser
     user
       Creator
     transactions
       TInit           : begin;
       TOpen           : 3;
       TShutdown       : end;
   ```

```
user
  Querist
transactions
  TInit           : begin;
  TQuery          : 5;
  TShutdown       : end;
```

In the edited version of the script, each user of the Creator group performs the TInit transaction at the beginning, the TOpen transaction three times, and the TShutdown transaction at the end. Each user of the Querist group performs the TInit transaction at the beginning, the TQuery transaction five times, and the TShutdown transaction at the end.

3 Select **File/Save** from the menu bar to save your changes.

4 In the **Workflow** toolbar, click **Customize Test**.

The **Workflow – Customize Test** dialog box opens.

5 Click **OK** to confirm that you are done customizing the test script.

What you have learned

In Tutorial 3, you learned how to:

- Replace constant values with random variables
- Specify which tasks each virtual user will perform

Tutorial 4: Replaying the customized script

After you have created a test script by recording function calls that the client application performs and then customizing the script, you can load test your server by replaying the script. To run a load test, you must perform the following steps:

- Find the test baseline
- Confirm the test baseline
- Set up server monitoring for your load test
- Execute the load test

Finding the test baseline

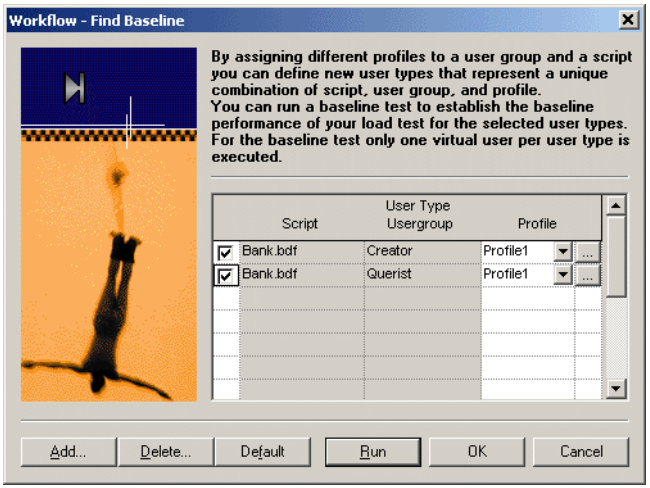
After you have customized the load-testing script, you should ascertain the baseline performance – that is, the basic, ideal performance – of the server under test. The fully customized test script is run with just one user per user group, and the results from this unstressed performance from the application form the basis for comparison with the results of the full load test later.

Finding the baseline

Procedure To find the load test baseline:

1 In the **Workflow** toolbar, click the **Find Baseline** button.

The **Workflow – Find Baseline** dialog box opens.



This dialog box lists all the user groups you have set up in your test script.

2 Click **Run**.

Silk Performer runs one virtual user from each user group that you have declared in the test script. While the test is running, you can watch progress in the **Monitor** window.

Confirming the test baseline

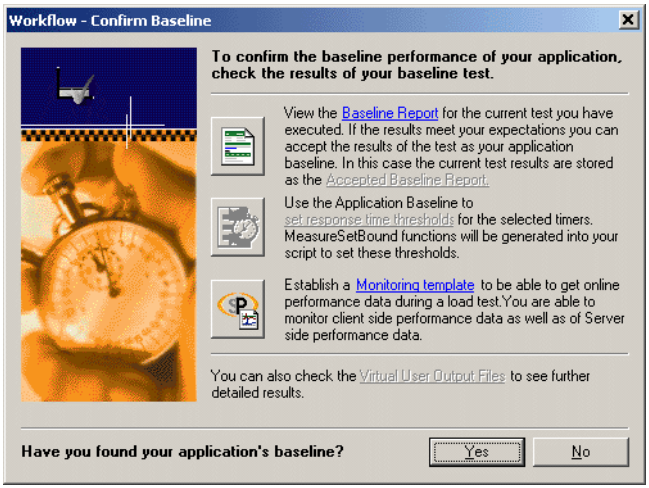
After this test run, you have to confirm that the test baseline does actually reflect the desired performance by the application under test. This is done by inspecting the results from that test. If the results are satisfactory, the baseline established will form the basis for comparison with the results from the full load test later.

Confirming the baseline

Procedure To confirm the load test baseline:

1 In the **Workflow** toolbar, click the **Confirm Baseline** button.

The **Workflow – Confirm Baseline** dialog box opens.



2 Click the **Baseline Report** button.

The Baseline Report opens.

3 Check the report carefully. In particular, make sure that no errors have occurred during your baseline test.

4 Click **Accept Baseline** button in the Baseline Report.

5 Click **Yes** and **OK** by the Message boxes that comes up.

6 If your baseline report indicates no errors, click **Yes** in the **Workflow – Confirm Baseline** dialog box to confirm that you have found the baseline.

Setting up server monitoring for your load test

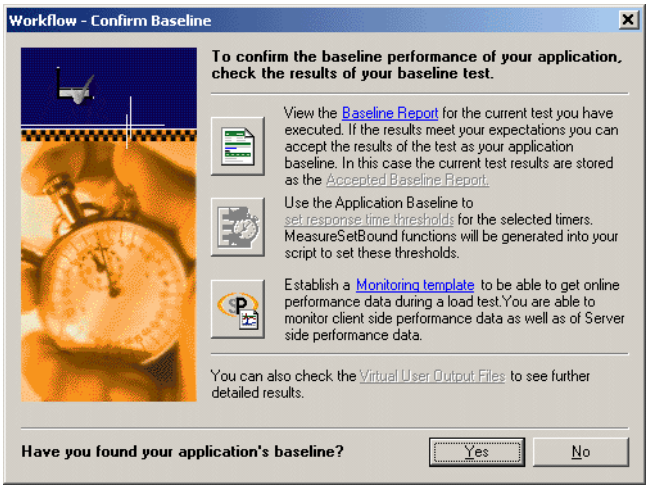
Monitoring the server under test helps you to find out whether there are bottlenecks on the server, and, if so, where exactly they are.

Setting up server monitoring

Procedure To set up server monitoring for your load test:

1 In the **Workflow** toolbar, once again click the **Confirm Baseline** button.

The **Workflow – Confirm Baseline** dialog box opens.



2 Click the **Monitoring template** button.

The **Profile - [Profile1] - Results** dialog box opens on the **Monitoring** tab.

The option **Automatically start monitoring** is activated by default. It means, that when you run a load test, the Performance Explorer will open automatically using a default template that invokes a display of relevant client information.

3 If you want to customize monitoring, enable the **Use custom monitoring template**, and click the **Create Customer Monitor Template** button.

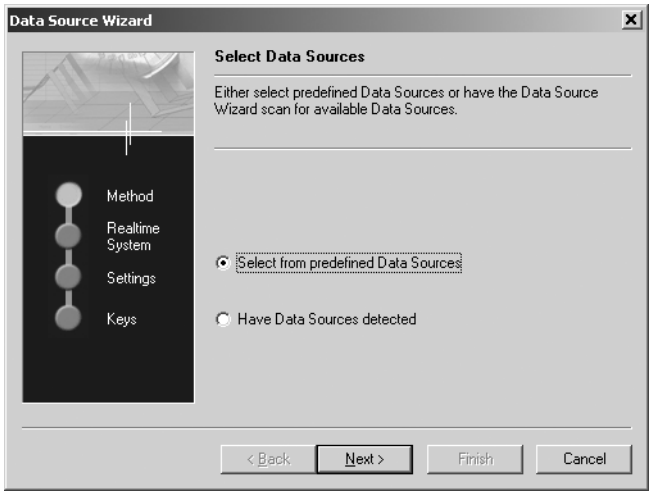
Performance Explorer will create and open a template with the name of your project, which is equal to the default template.

4 Click **Edit Customer Monitor Template**, to customize the new template and change the display of information. You can close or maintain the default monitoring windows and you can add one or more windows to provide additional information.

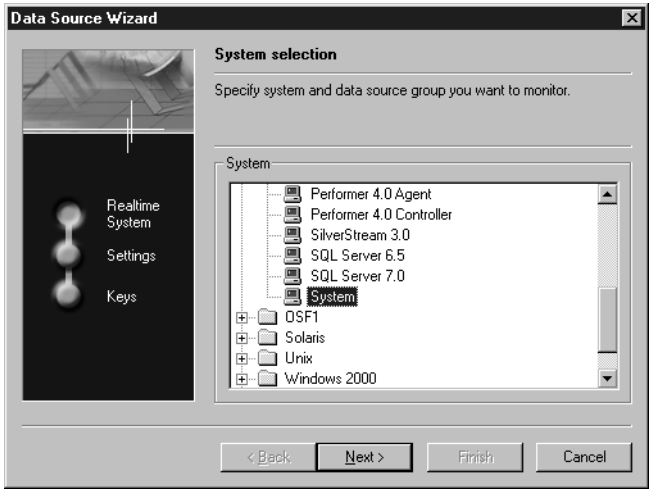
To set up Performance Explorer to display additional information, proceed as follows:

a Click the **Monitor Server** button in the Performance Explorer workflow bar

The **Data Source Wizard** opens.



b Click **Next**.
The **Data Source Wizard - System selection** dialog box opens.



c In the tree view, expand the folder that corresponds to the operating system on which the Bank server is running. Within the folder, select **System**.

d Click **Next**.

e In the next dialog box, enter connection parameters, like the host name or IP address of the server, the connection port, the user name, and the password.

f The data you have to enter here depends on the operating system that is running on the computer you are monitoring.

g Click **Next** when you are done entering connection parameters.

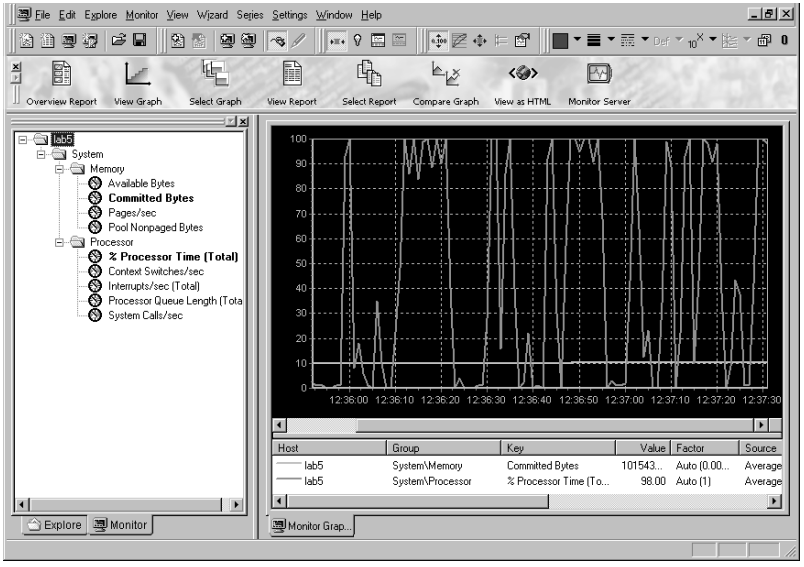
h In the next dialog box, select the performance counters you want to monitor.

i Of particular interest are the processor and memory utilization of the server being tested.

j Click **Finish**.

The Performance Explorer opens a new window and displays real-time data for the performance counters you have selected.

Later, when you are running the load test, the monitor view will look similar to the following.



Adjusting the workload

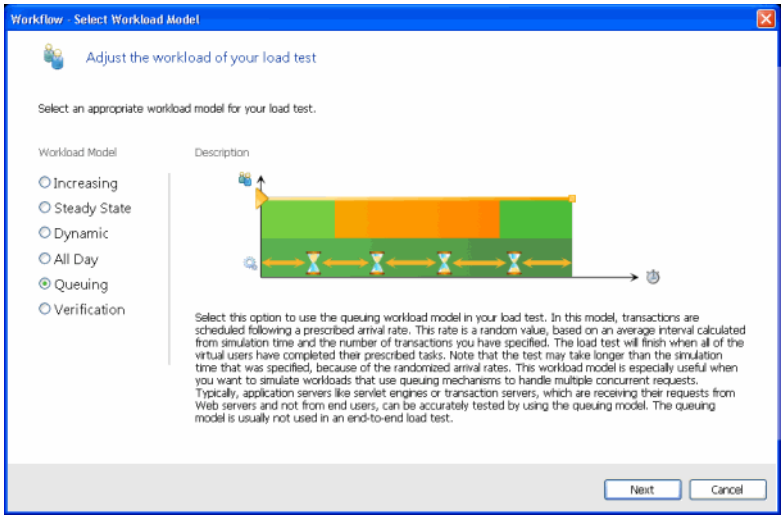
After you have confirmed your baseline, you now select your workload model and prepare the workload to run a full load test.

Adjusting the workload

Procedure To adjust the workload:

1 In the **Workflow** toolbar, click the **Adjust Workload** button.

The **Workflow - Select Workload Model** dialog box opens.



This dialog box shows all possible workload models, which can be selected.

2 Select **Queuing** workload model.

3 Click **Next**.

4 On the **Workflow Assign Agents** dialog box, select an option and click **OK**.

Silk Performer now opens the Workload Configuration dialog box.

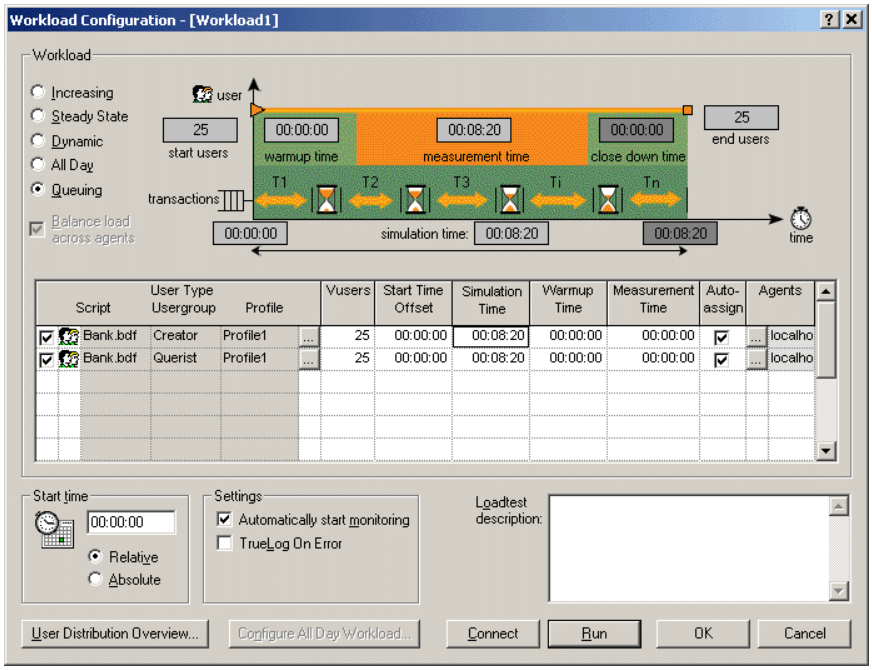
Running the load test

After you have adjust the workload, you run the full load test. The test script is run with multiple virtual users in order to test the Bank server. A large load test would need the appropriate testing environment to be set up in the local area network, including a full complement of agent computers to host the virtual users. For this tutorial, however, you use only your local computer.

Running the test

Procedure To run the load test:

1 The **Workload Configuration** dialog box is open.



2 In the **VUsers** column, change the number of Creator and Querist users to **25**.

3 For each of these user groups, change the simulation time to **300** seconds (5 minutes).

4 Disable all user groups that are declared in the Bankappn test script.

5 Click **Run**.

Silk Performer runs a load test with 50 virtual users, load testing the Bank server.

Performance Explorer opens automatically and you can watch progress of the load test and the server being tested in the **Monitor** window.

What you have learned

In Tutorial 4, you learned how to:

- Find the test baseline
- Confirm the test baseline
- Set up server monitoring for your load test
- Adjusting the workload
- Execute the load test

Tutorial 5: Viewing the results of your load test

Silk Performer incorporates **Performance Explorer**, a powerful graphing and analysis tool to help you work with your results information.

Performance Explorer provides a comprehensive array of results information, which can be displayed using advanced features for creating statistical reports and displaying performance results in real time generated graphics. Results information may vary, depending on the type of application and/or server being tested. In general it falls into the following categories:

- Response time information: This is the total time to process a given timer; it provides application performance information from the point of view of the client computer.
- Throughput information: This is the rate at which a given timer is processed on average by the server; it allows you to analyze performance from the point of view of the server.

Overview report of performed measurements

You can consult an HTML overview report that contains results of all performed measurements.

Exploring overview report

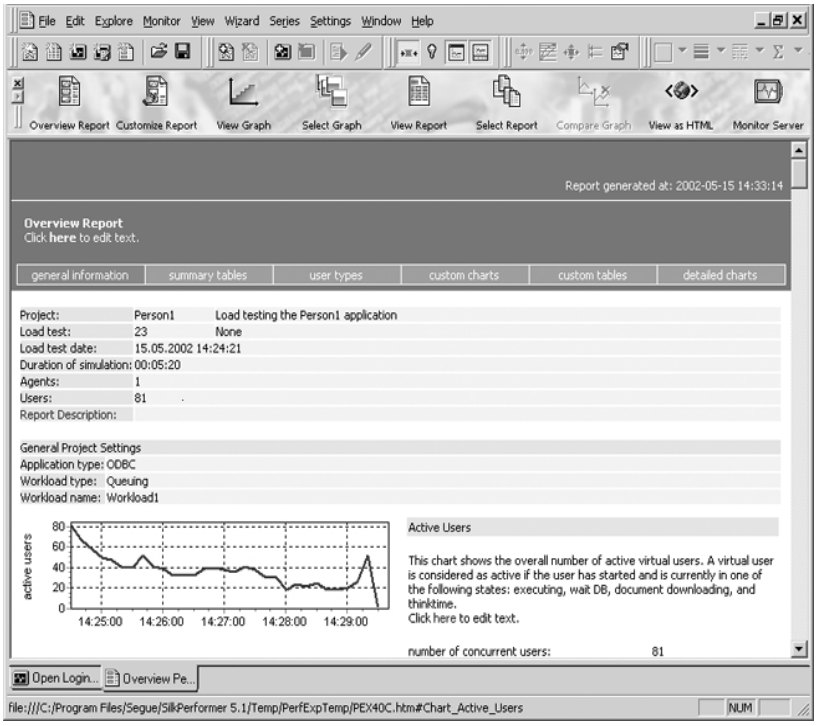
Procedure To explore overview report:

1 In the **Workflow** toolbar, click **Explore Results**.

The **Workflow - Explore Results** dialog box opens.

2 Click the **Performance Explorer** button.

Performance Explorer opens, and the HTML overview report is generated.



You may consult this report which includes short explanations of the displayed graphs. If you want you can view more detailed information.

Detailed response time measurements

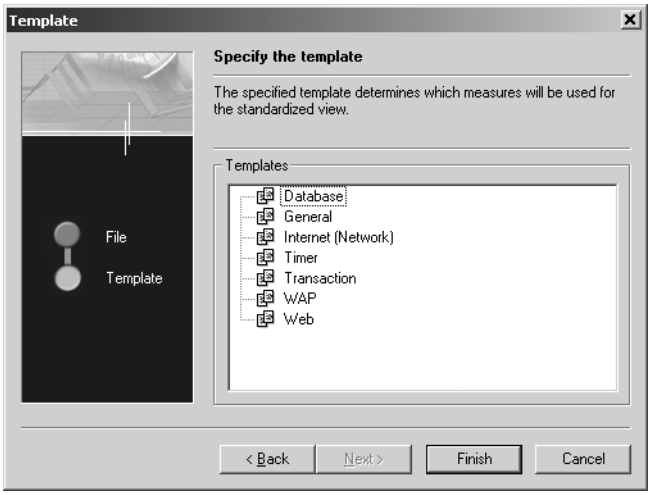
When you analyze performance of an application from the client point of view, you examine the response times for all the transactions and individual timers. This will enable you to find out how users will experience their interaction with server objects. A common goal in load testing is to find out if response times for all transactions remain below a specified, critical limit.

Response time details

Procedure To explore detailed response time measurements:

1 In the Performance Explorer **Workflow** bar, click the **Select Graph** button.

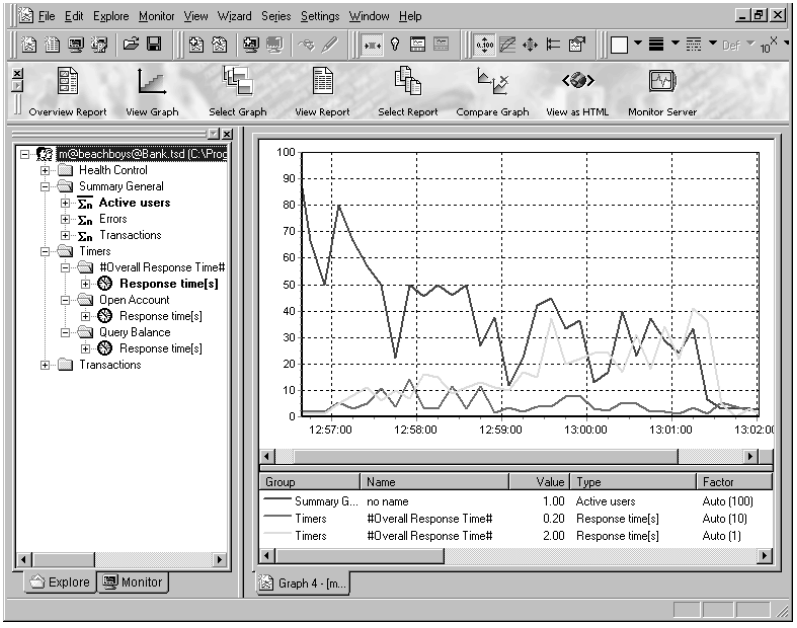
The **Template** dialog box opens.



2 In the **Templates** area, select the **Timer** option.

3 Click **Finish**.

The Performance Explorer displays a chart that looks similar to the following.



The load test that you ran in the previous tutorial lasted only a few minutes, so there are only a few measurements, which makes it difficult to draw any conclusions.

In real-life you will run load tests that last several hours, a few days, or even a week. These tests will provide results information that will allow you analyze the performance of your server accurately.

Exploring throughput measurements

When you analyze the performance of a server from the point of view of the server, you examine the throughput rate. Throughput is the work done by the server within a specific time interval, for example, the number of requests that the server processes within a second, a minute, or an hour. A common goal in load testing is to ensure that the throughput rate is above a specified, critical limit.

Throughput rate details

Procedure To explore throughput measurements:

1. From the menu bar, select **File/New/Graph**.

   The Performance Explorer opens a new, blank graph.

2. In the **Explore** tree view, expand both the **Summary General** and the **Timers** measurement groups.

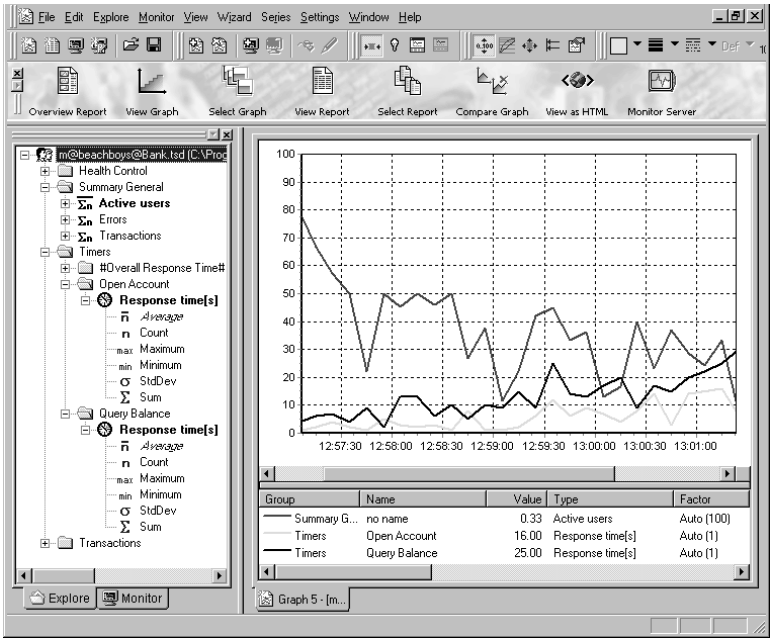
3. In the **Summary General** measurement group, drag the **Active users** measurement and drop it on the graph.

4. In the **Timers** measurement group, select **Open Account/Response time**, drag the **Count** measurement, and drop it on the graph.

5. In the **Timers** measurement group, select **Query Balance/Response time**, drag the **Count** measurement, and drop it on the graph.

   You see a graph that displays the number of service calls processed by the Bank server.

The graph should look similar to the following.



Again, since the load test that you ran lasted only five minutes, it is difficult to derive conclusions from the results.

What you have learned

In Tutorial 5, you learned how to:

- Explore response time measurements
- Explore throughput measurements

7

Load Testing a Jolt Application

Introduction

This chapter contains a series of tutorials that will help you to start load-testing Jolt applications with Silk Performer.

What you will learn

This chapter contains the following sections:

Overview

Each of the following tutorials provides step-by-step instructions for important Silk Performer tasks. You should perform the tutorials in order, since each tutorial builds on what you have learned in the previous ones.

You will learn how to do the following:

- Execute a predefined load test on your local machine.
- Create a test script by recording Jolt function calls using the Recorder.
- Customize the recorded test script.

- Run the customized script, using multiple virtual users.
- Use reporting tools to view throughput and response time information.

Sample: Bank applet

In these tutorials, you will use the sample BEA Jolt Bank applet; this applet is included on the BEA Jolt CD-ROM. Make sure your network configuration fulfills the following requirements:

- The BEA TUXEDO System server software is installed and running on a computer in your local area network.
- The BEA Jolt server software is installed and running on the computer as the TUXEDO System; this computer must also run a Web server to enable the applet to be accessed.
- The BEA Jolt client software is installed on your Silk Performer controller computer.
- Any version of the Sun Java Development Kit (JDK) is installed on your Silk Performer controller computer.

Tutorial 1: Running a sample load test

The following tutorial shows you how to create load-testing projects and execute sample scripts that accompany Silk Performer. You will run one of the sample scripts on your local machine, for a single virtual user.

For now, just to keep everything simple, you will work with a sample script provided by Silk Performer. This is the test script JoltBankApp.bdf, which is included at **<Public User Documents>\Silk Performer 16.5\Samples\Middleware\Jolt**.

Creating a load-testing project

For each application you load test with Silk Performer, you have to create a project file. A project file administers all the settings that are associated with a series of load tests, like the test type, the workload model, and a number of options that depend on the type of application being tested. In addition, project files enable you to easily switch between different load-testing projects without having to define any new settings when you switch to a different project.

Creating a project

Procedure To create your load-testing project:

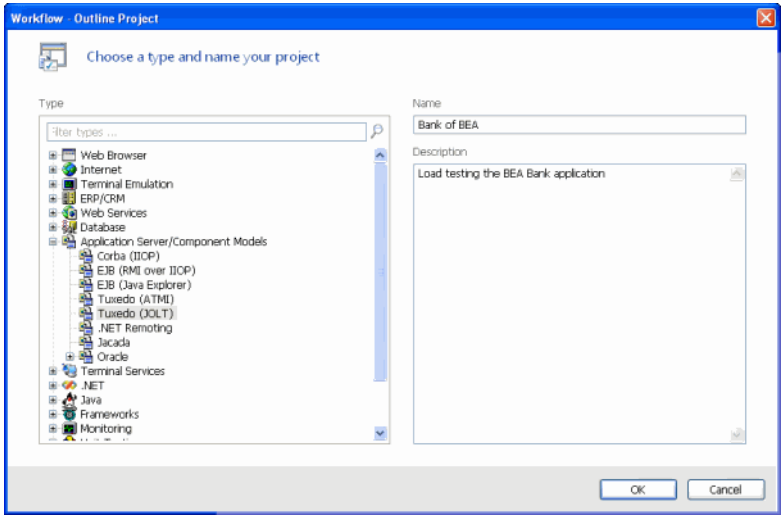
1. Open Silk Performer.

   If you did not close the project you last worked on, Silk Performer will open with that project.

2. Close any open projects.

3 In the **Workflow** bar, click **Start here**.

The **Workflow – Outline Project** dialog box opens.



4 In the **Name** text box, enter **Bank of BEA** as the name for your load-testing project.

5 Enter an optional project description in **Description**.

6 In the **Type** list box, select the **Tuxedo (JOLT)** option.

7 Click **OK**.

Silk Performer creates a new load-testing project called “Bank of Bea” with the default settings profile and your local computer as the only agent.

Adding a load-testing script

Each project contains a number of load-testing scripts. A test script defines different groups of virtual users and all the actions that these users have to perform. To be able to execute a load-testing script, you have to add it to your project.

Adding a test script

Procedure To add a predefined script to your project:

1 In the menu tree, right-click **Scripts** and select **Add Existing Script**.

The **Select Script(s)** dialog box opens.

2 Navigate to **<Public User Documents>\Silk Performer 16.5\Sample\Middleware\Jolt** and select the **JoltBankApp** script.

3 Click **Open**.

Silk Performer adds the JoltBankApp test script to the **Scripts** folder of your current project and displays the script in a new editor window.

Personalizing the sample script

The JoltBankApp script contains a number of constant declarations that specify the Java home directory, the Java classpath, and the server to which the virtual users connect. Before you execute the script, you have to change the constant values so that they correspond to your configuration.

Personalizing the test

Procedure To personalize the sample load-testing script:

1 In the **Active Script** window, select **Symbols/Constants/HOST**.

You see the following.

```
HOST := "//lab5:8000";   // TUXEDO host name
```

2 Replace the address “lab5” with the name of the server where the TUXEDO System server software is running.

3 If necessary, change the port number (8,000) to the number of the port that the server listens to.

4 In the **Active Script** window, select **Symbols/Constants/ JOLTCLASSPATH**.

You see the following.

```
JOLTCLASSPATH:="c:/jolt/lib/jolt11.zip"; // path of the Jolt package
```

5 Replace the directory name “c:/jolt/lib” with the name of the directory where the Jolt classes are located on your computer.

6 In the **Active Script** window, select **Symbols/Transactions/TInit**.

7 In the TInit transaction, locate the following two lines.

```
JavaSetOption(JAVA_VERSION, JAVA_V12); // JVM 1.2.x
JavaSetOption(JAVA_HOME, "c:/jdk1.2"); // contains folders: bin/, lib/
```

8 If any version of the Java Development Kit 1.1 is installed on your computer, replace the second parameter of the first function call with the JAVA_V11 constant. The function call would then look like the following.

```
JavaSetOption(JAVA_VERSION, JAVA_V11); // JVM 1.2.x
```

9 In the second function call, replace the directory “c:/jdk1.2” with the name of the directory where the Java Development Kit is installed on your computer.

10 Select **File/Save** from the menu bar to save your changes.

Executing the test script

In the JoltBankAPP test script, the JoltUser user group is set up. Virtual users in this group will create a new account, query the current balance of the account, deposit some money in the account, and withdraw some money from the account.

Executing the test script

Procedure To execute the JoltBankApp test script:

1. In the **Active Script** window, expand the **User Groups** folder.
2. In the **User Groups** folder, right-click **JoltUser** and select **Run User "JoltUser"**.

   Silk Performer runs a virtual user, performing all the actions defined in the test script. While the virtual user is being run, you can watch progress in the **Monitor** window, for example, the number of transactions executed, the response time of the last transaction, and the average response time.
3. Monitor the load test and wait until the virtual user has finished executing its tasks.
   a. In the top part of the **Monitor** window, select an agent computer or a user group that you want to monitor.

      In the bottom part of the **Monitor** window, Silk Performer displays overview information about all the virtual users running on the selected agent computer or belonging to the selected user group.
   b. In the bottom part of the **Monitor** window, right-click a virtual user for which you want to view detailed information and select **Show Output**.

      Silk Performer displays detailed run-time information about the selected user in the **Virtual User** window at the bottom, for example, the transactions and functions the user executes, and the data the user sends to and receives from the server. The type of information displayed depends on the options that you select at the top of the Monitor window.

Display Errors. Select this button to display an appropriate error message indicating the probable reason whenever a user-related error occurs.

Display Transactions. Select this button to display a message when a user has finished executing a transaction; the message indicates whether the transaction was successful or not.

Display Functions. Select this button to display a message for each function call a given user performs, including the function name and all its parameters.

Display Info. Select this button to display messages containing additional information about the current action a given user performs.

Display Data. Select this button to show data exchanged with the server.

Display all Errors of all Users. Select this button to display error messages for all errors of all users. Each error message will indicate user, agent and probable cause of the error.

This information will be displayed in addition to the information chosen for the selected user.

What you have learned

In Tutorial 1, you learned how to:

- Create a load-testing project
- Add a predefined script to the project
- Execute the script with one virtual user
- View progress information for the load test

Tutorial 2: Creating a script with the Recorder

The Silk Performer Recorder allows you to capture and record the Jolt traffic between the client and the server. After you record traffic, you can save it as a test script. You can then easily customize the script and replay the script to simulate a large number of virtual users. Recording, customizing, and replaying a script will be covered separately in Tutorials 2, 3, and 4. In Tutorial 5, you will analyze the results of the load test you ran.

This tutorial includes the following steps:

- Specifying Java-specific options
- Setting up an application profile
- Recording traffic between a client and a server

- Generating a test script based on the recorded traffic
- Trying out the generated test script
- Validating the generated test script

Specifying Java-specific load-testing options

Before you can capture and record Jolt traffic, you need to tell the Recorder where the Java Development Kit and the Jolt classes are located on your computer.

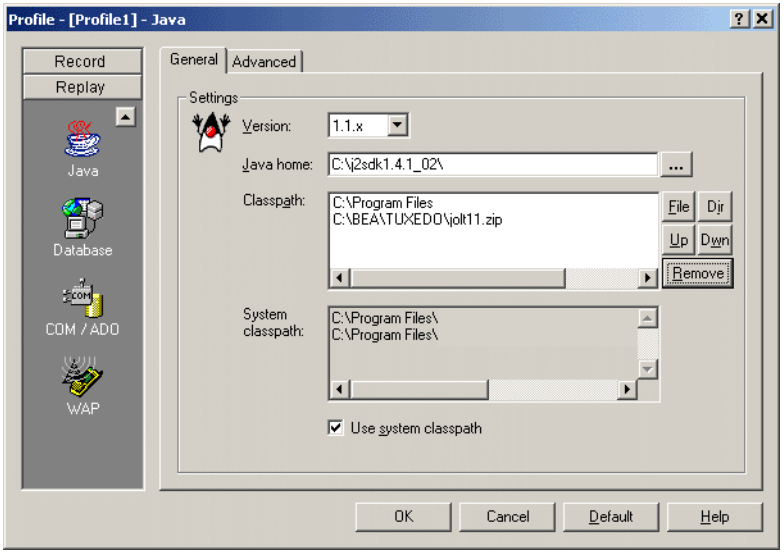
Specifying Java options

Procedure To point to the Java Development Kit:

1. In the **Project** window, expand the **Profiles** folder.
2. Right-click the **Profile1** profile and select **Edit Profile**.

   The **Profile – [Profile1] – Simulation** dialog box opens.
3. In the shortcut list on the left side, select the **Replay** category and click the **Java** icon.

   You see the **General** tab of the **Profile – [Profile1] – Java** dialog box.



4. From the Version list box, select the version number of the Java Development Kit that is installed on your computer.
5. In the Java home text box, specify the directory where the Java Development Kit is installed on your computer.

6 In the **Classpath** field, specify the archives that contain the basic Java Silk Performer framework classes (javaUserFramework.zip) and the Jolt classes (jolt.zip). Click the **File** and **Dir** buttons to add files and directories to the field, respectively. The javaUserFramework.zip file is located in the ClassFiles subfolder of the Silk Performer home directory.

   Note Make sure that *JoltMessageBundle.properties* is also added to the classpath. If this properties file is not available in the *jolt.zip* file, try the *jolt.jar* file.

7 Click **OK**.

Setting up an application profile

Whenever you want the Silk Performer Recorder to capture and record Jolt traffic that is exchanged between the client and the server, you have to set up a profile for the client. This profile specifies the type of the client and what type of traffic to record.

In this tutorial, you will use a Web browser to access the BEA Bank applet. Silk Performer automatically sets up profiles for Microsoft Internet Explorer and Netscape Communicator (if they are installed on your computer). To record Jolt traffic, however, you have to modify the application profiles that are set up by default.

Modifying the application profile

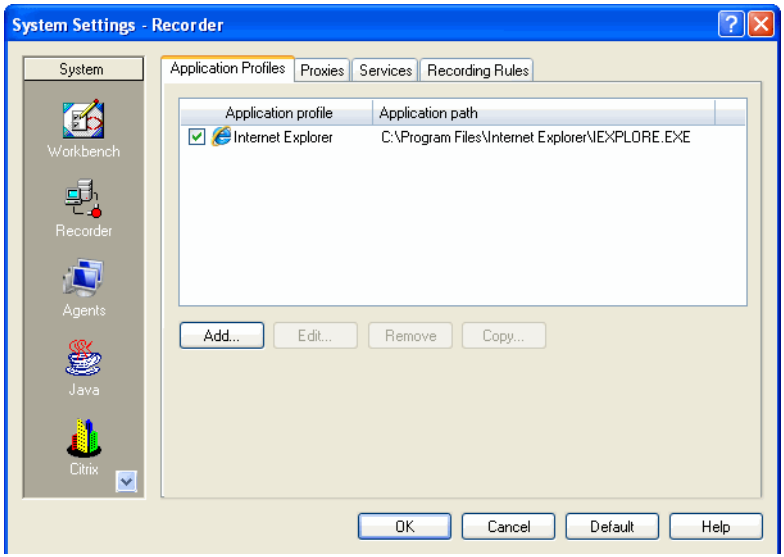
Procedure To modify the profile for the Web browser:

1 From the menu bar, select **Settings/System**.

   The **System Settings – Workbench** dialog box opens.

2 In the shortcut list on the left side, click the **Recorder** icon.

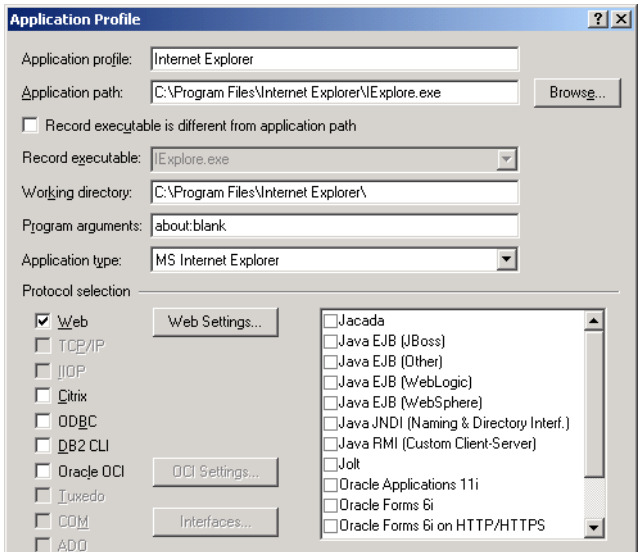
You see the following.



3 Select the application profile that corresponds to the Web browser you want to use for recording Jolt traffic. For example, if you want to use Microsoft Internet Explorer, select **Internet Explorer**.

4 Click **Edit**.

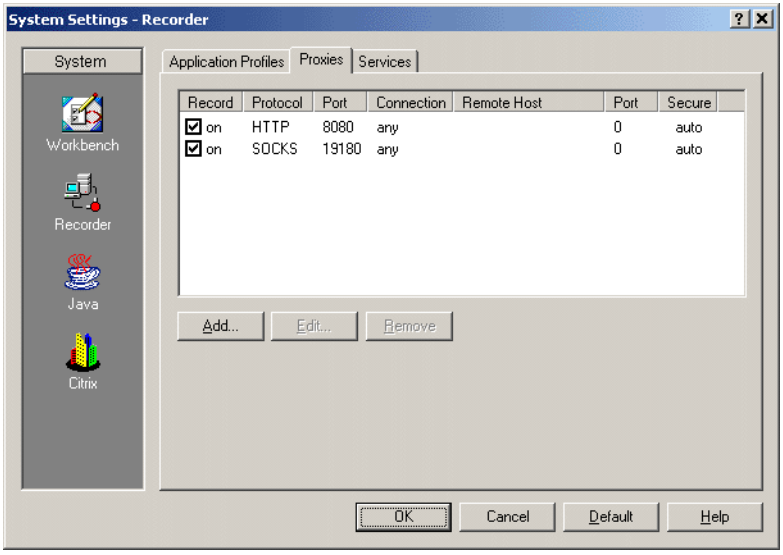
The **Application Profile** dialog box opens.



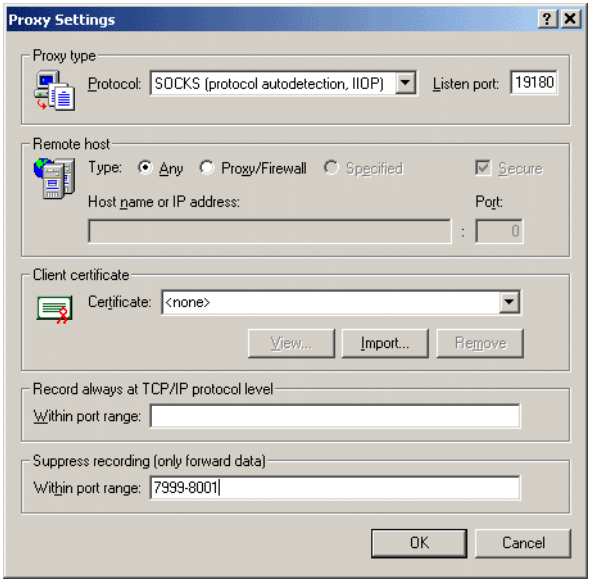
5 In the **Java API** area, select the API you want.

6 Click **OK**.

7 In the **System Settings - Recorder** dialog box, click the **Proxies** tab**.** You see the following.



8 Select the protocol type **SOCKS** and click **Edit**.
The **Proxy Settings** dialog box opens.



9 In the **Suppress recording (only forward data)** area, in the **Within port range** edit field, specify the port range on which the Jolt lisener is running. This must be specified that only the Jolt functions are recorded, and not also the TCP/IP functions.

10 Click **OK**.

11 In the **System Settings – Recorder** dialog box, click **OK** to close the dialog.

Modeling a test script

To load test the Jolt Bank applet, the Jolt traffic between the client and the server needs to be recorded and then described in a test script. The script finally enables any number of virtual users to perform actions similar to those that you performed during the recording session.

Modeling a test script

Procedure To model a load-testing script:

1 In the **Workflow** bar, click **Model Script**.

The **Workflow – Model Script** dialog box opens.

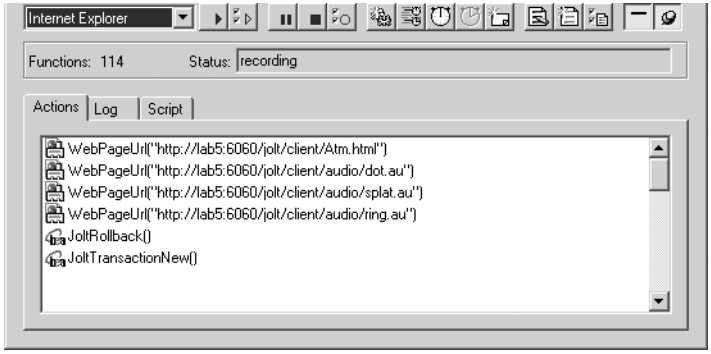
2 From the **Application Profile** list box, select the profile that corresponds to the Web browser you want to use for recording traffic.

3 In the **URL** text box, specify the URL of the BEA Bank applet.

4 Click **Start recording**.

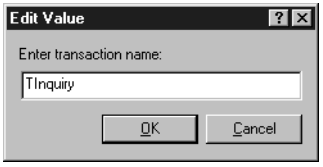
Silk Performer opens the Recorder and starts the Web browser. The Recorder automatically records the Jolt traffic that the client sends to and receives from the server.

5 To view your actions as well as the requests sent to the server and the responses received, click the **Change GUI Size** button.



6 Click the **New Transaction** button to insert a new transaction into the test script you are generating.

The following dialog box opens.



7 Create a new transaction called **TInquiry**; then click **OK**. A distinct set of time measurements will be made for each transaction you create. When recording traffic, you should create a new transaction for each piece of work that has no dependencies on others.

8 In your Web browser, enter your PIN (for example, 100, 120, or 121). You do this by clicking the digits on the number pad and clicking **OK** when you are done.

9 To the right of the number pad, click **Inquiry** and then **Checking Acct**.

10 In the Recorder, create a new transaction called **TDeposit**.

11 In your Web browser, again enter your PIN (for example, 100, 120, or 121).

12 To the right of the number pad, click **Deposit** and then **Checking Acct**.

13 Using the number pad, enter the amount of money you want to deposit in the account, for example, $100. Then click **OK**.

14 In the Recorder, click the **Stop Recording** button.

The **Save As** dialog box opens.

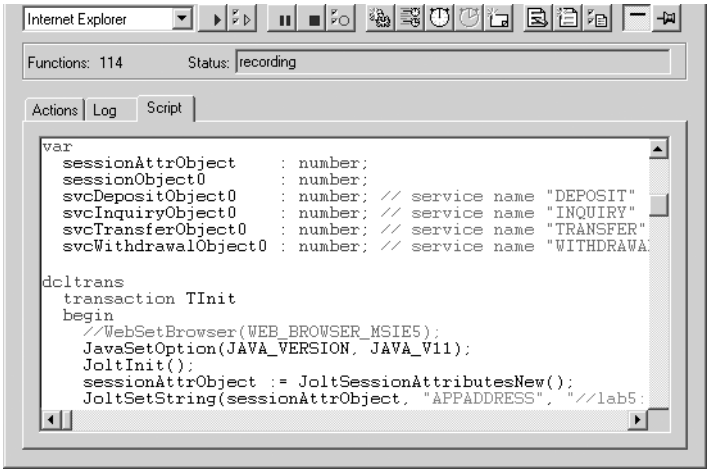
15 In the **File name** text box, enter **Bank of BEA**.

16 Click **Save**.

Silk Performer automatically adds the script to your current load-testing project.

17 You can now decide whether to close the Recorder.

If you keep it open, you can examine the script that has been generated.



18 Close the Recorder.

Trying out the generated test script

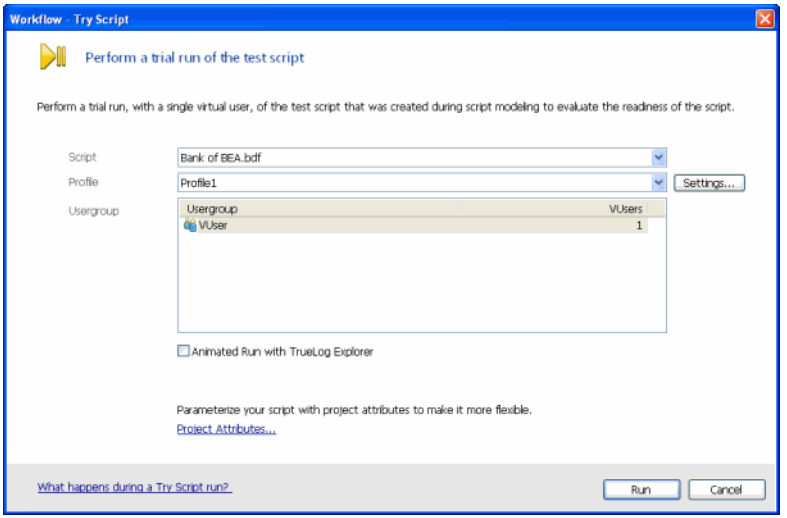
Next you should do a trial run of the test script that was created. The object of the trial run is to ensure that the script is free from error, and that it will reproduce accurately the interaction between the client and the server.

Trying out the script

Procedure To try out the generated test script:

1 In the **Workflow** bar, click **Try Script**.

The **Workflow - Try Script** dialog box opens.



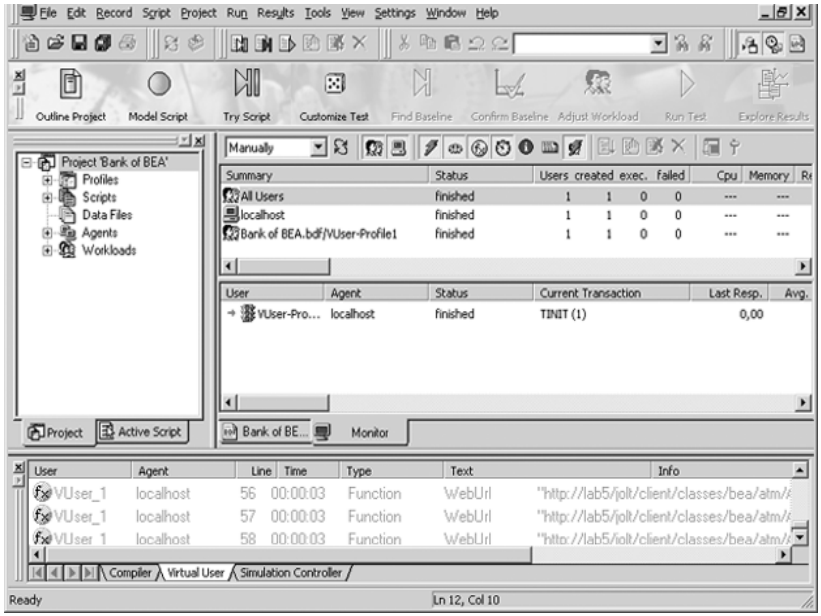
2 From the **Script** list box, select the **Bank of BEA.bdf** script.

3 Make sure the **VUser** group is selected in the **Usergroup** list.

4 Disable the **Animated Run with TrueLog Explorer** option.

5 Click **Run**.

Silk Performer runs a single virtual user, called VUser, which performs all the actions that you have performed during the previous recording session. While the single-user test is running, you can watch progress in the **Monitor** window.



What you have learned

In Tutorial 2, you learned how to:

- Set up an application profile
- Record traffic between a client application and a server
- Generate a test script based on the recorded traffic
- Try out the generated test script

Tutorial 3: Customizing the generated test script

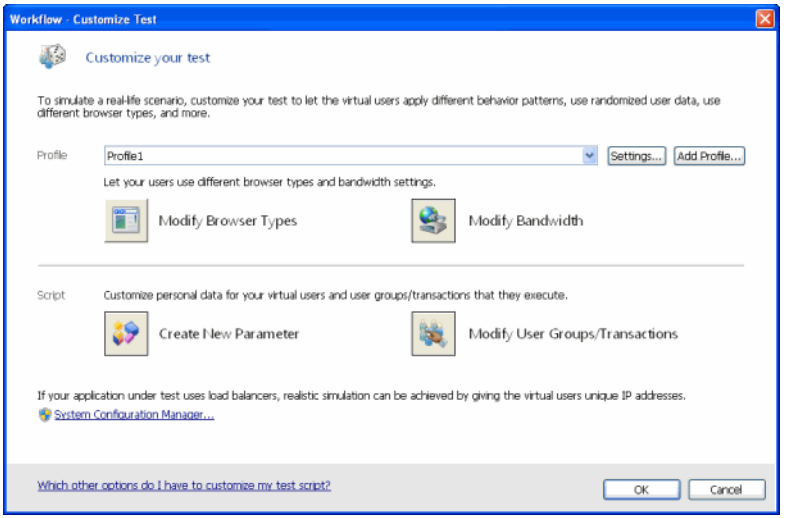
Where randomized data is necessary, random variables are employed to provide realistic user data. In this way, Silk Performer furnishes the virtual users with varied personal data – like name, address, phone number, and so on.

Replacing constant values

Procedure To replace constant values with random variables:

1 In the **Workflow** bar, click **Customize Test**.

The **Workflow – Customize Test** dialog box opens.



2 Click the **Create New Parameter** icon.
The **Parameter Wizard - Create New Parameter** dialog box opens.

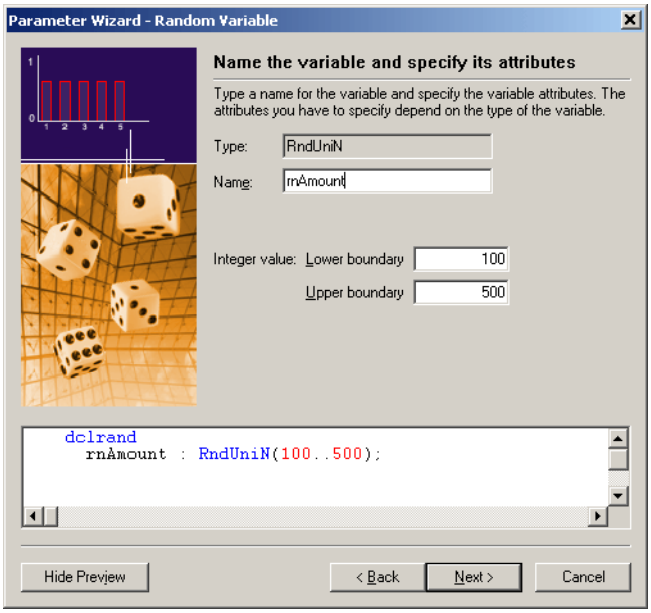
3 Select **Parameter from Random Variable** and click **Next**.
The **Parameter Wizard - Random Variable** dialog box opens.

The **Random Variable Wizard** opens.

4 From the **Random type** list box, select the **RndUniN** type.

5 Click **Next**.

You see the following dialog box.



6 In the **Name** text box, enter **rnAmount**.

This is the name for the random variable that will be inserted into your test script.

7 In the Upper boundary edit field, enter **500**.

8 In the Lower boundary edit field, enter **100**.

9 Click **Finish**.

Silk Performer inserts the following lines into your test script.

```
dclrand
  rnAmount : RndUniN(100..500);
```

10 Repeat the steps 1-8, but

- in the first dialog box, instead of selecting the **RndUniN** option, select **RndInd**.
- in the second dialog box, enter **rnAccount** in the **Name** edit field, select the **Number** option in the **Parameter type** area, and enter the following pairs in the Parameter list.

Value	Weight
10001	50
10002	50

The **dclrand** section of the test script should now look like the following.

```
dclrand
  rnAccount : RndInd(10001 = 0.500000;
                     10002 = 0.500000);
  rnAmount  : RndUniN(100..500);
```

11 Locate the TInquiry transaction.

You can do this by expanding the **Transactions** folder in the **Active Script** window and double-clicking **TInquiry**.

12 In the TInquiry transaction, locate the following function call.

```
JoltSetInt(svcInquiryObject0, "ACCOUNT_ID", 10001);
```

13 Replace the constant (10,001) with the random variable **rnAccount**.

The function should now look like the following. The code shown in bold type is what you have changed.

```
JoltSetInt(svcInquiryObject0, "ACCOUNT_ID", rnAccount);
```

14 Locate the TDeposit transaction.

You can do this by expanding the **Transactions** folder in the **Active Script** window and double-clicking **TDeposit**.

You see the following.

```
transaction TDeposit
var
begin
  JoltSetInt(svcDepositObject0, "ACCOUNT_ID", 10001);
  JoltSetString(svcDepositObject0, "SAMOUNT", 100);
```

15 In the first function call, again replace the constant (10,001) with the random variable **rnAccount**.

16 In the second function call, replace the constant (100) with the expression **string(rnAmount)**.

The beginning of the TDeposit transaction should now look like the following. The code shown in bold type is what you have changed.

```
transaction TDeposit
var
begin
  JoltSetInt(svcDepositObject0, "ACCOUNT_ID", rnAccount);
  JoltSetString(svcDepositObject0, "SAMOUNT", string(rnAmount));
```

17 When you are done making changes to the script file, select **File/Save** from the menu bar to save your changes.

Specifying which tasks each user will perform

Usually a load test simulates a number of different types of users. In the test script, you have to specify which action each type of virtual user will perform.

Assigning tasks to users

Procedure To specify which tasks each user will perform:

1. Navigate to the **dcluser** section of the Bank.bdf script file. You can do this by selecting **User Groups/VUser** in the **Active Script** window.

   You see the following.

   ```
   dcluser
     user
       VUser
     transactions
       TInit           : begin;
       TMain           : 1;
       TInquiry        : 1;
       TDeposit        : 1;
   ```

   In this section, a single user group is defined, called VUser. By default, this user group will perform the transactions that you created, Tinit, TMain, TInquiry, and TDeposit once each.

2. Edit the **dcluser** section of the script so it looks like the following.

   ```
   dcluser
     user
       Inquirer
     transactions
       TInit           : begin;
       TMain           : 1;
       TInquiry        : 3;

     user
       Depositor
     transactions
       TInit           : begin;
       TMain           : 1;
       TDeposit        : 2;
   ```

   In the edited version of the script, each user of the Inquirer group performs the TInit transaction at the beginning and then the TMain transaction once and the TInquiry transaction three times. Each user of the Depositor group performs the TInit transaction at the beginning and then the TMain transaction once and the TDeposit transaction twice.

3. Select **File/Save** from the menu bar to save your changes.
4. In the **Workflow** toolbar, click **Customize Test**.

   The **Workflow – Customize Test** dialog box opens.

5. Click **OK** to confirm that you are done customizing the test script.

What you have learned

In Tutorial 3, you learned how to:

- Replace constant values with random variables
- Specify which tasks each virtual user will perform

Tutorial 4: Replaying the customized script

After you have created a test script by recording traffic and then customizing the script, you can load test your server by replaying the script. To run a load test, you must perform the following steps:

- Find the test baseline
- Confirm the test baseline
- Set up server monitoring for your load test
- Execute the load test

Finding the test baseline

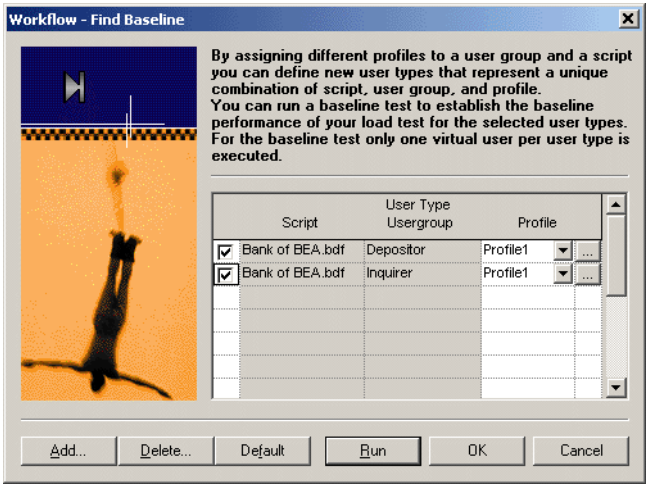
After you have customized the load-testing script, you should ascertain the baseline performance – that is, the basic, ideal performance – of the server objects under test. The fully customized test script is run with just one user per user group, and the results from this unstressed performance from the server objects form the basis for comparison with the results of the full load test later.

Finding the baseline

Procedure To find the load test baseline:

1 In the **Workflow** toolbar, click the **Find Baseline** button.

The **Workflow – Find Baseline** dialog box opens.



This dialog box lists all the user groups you have set up in your test script.

2 Click **Run**.

Silk Performer runs one virtual user from each user group that you have declared in the test script. While the test is running, you can watch progress in the **Monitor** window.

Confirming the test baseline

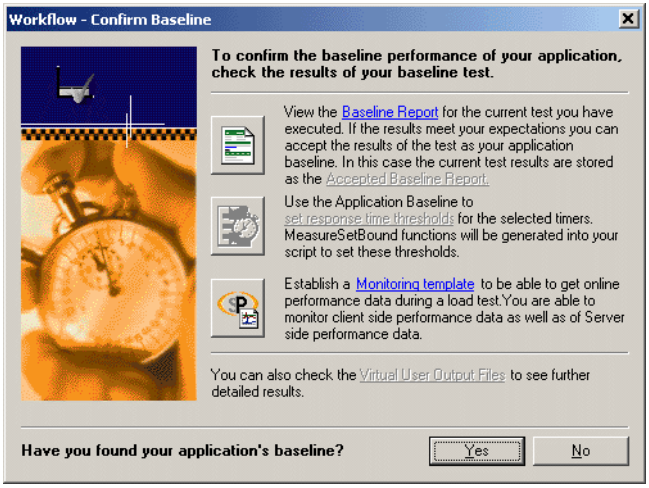
After this test run, you have to confirm that the test baseline does actually reflect the desired performance by the application under test. This is done by inspecting the results from that test. If the results are satisfactory, the baseline established will form the basis for comparison with the results from the full load test later.

Confirming the baseline

Procedure To confirm the load test baseline:

1. In the **Workflow** toolbar, click the **Confirm Baseline** button.

   The **Workflow – Confirm Baseline** dialog box opens.



2. Click the **Baseline Report** button.

   The Baseline Report opens.

3. Check the report carefully. In particular, make sure that no errors have occurred during your baseline test.

4. Click **Accept Baseline** button in the Baseline Report.

5. Click **Yes** and **OK** by the Message boxes that come up.

6. If your baseline report indicates no errors, click **Yes** in the **Workflow – Confirm Baseline** dialog box to confirm that you have found the baseline.

Setting up server monitoring for your load test

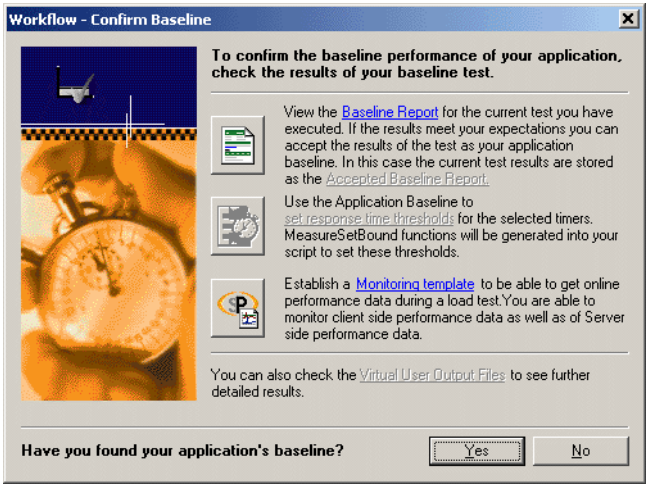
Monitoring the server under test helps you to find out whether there are bottlenecks on the server objects, and, if so, where exactly they are.

Setting up server monitoring

Procedure To set up server monitoring for your load test:

1 In the **Workflow** toolbar, once again click the **Confirm Baseline** button.

The **Workflow – Confirm Baseline** dialog box opens.



2 Click the **Monitoring template** button.

The **Profile - [Profile1] - Results** dialog box opens on the **Monitoring** tab.

The option **Automatically start monitoring** is activated by default. It means, that when you run a load test, the Performance Explorer will open automatically using a default template that invokes a display of relevant client information.

3 If you want to customize monitoring, enable the **Use custom monitoring template**, and click the **Create Customer Monitor Template** button.

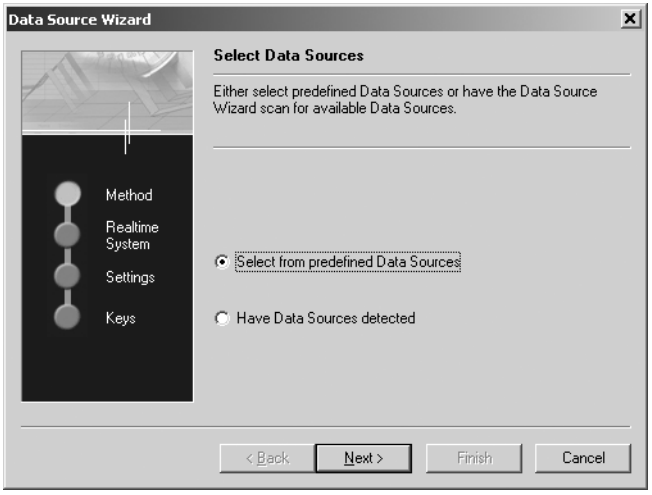
The Performance Explorer will create and open a template with the name of your project, which is equal to the default template.

4 Click **Edit Customer Monitor Template**, to customize the new template and change the display of information. You can close or maintain the default monitoring windows and you can add one or more windows to provide additional information.

To set up Performance Explorer to display additional information, proceed as follows:

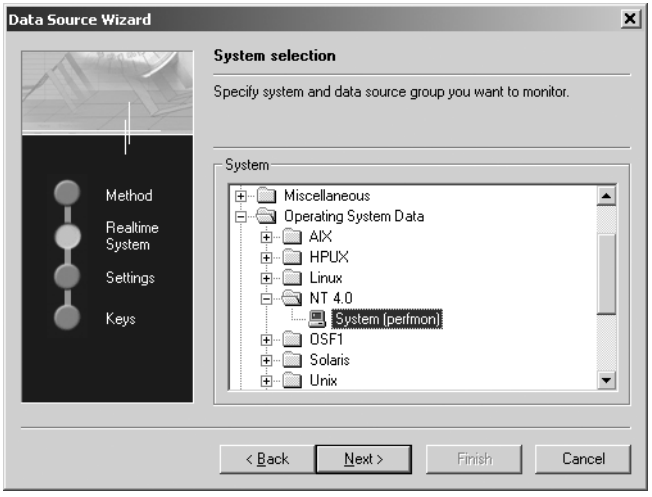
a Click the **Monitor Server** button in the Performance Explorer workflow bar.

The **Data Source Wizard** opens.



b Click **Next**.

The **Data Source Wizard - System selection** dialog box opens.



c In the tree view, expand the folder that corresponds to the operating system on which the Bank server is running. Within the folder, select **System**.

d Click **Next**.

e In the next dialog box, enter connection parameters, like the host name or IP address of the server, the connection port, the user name, and the password.

The data you have to enter here depends on the operating system that is running on the computer you are monitoring.

f Click **Next** when you are done entering connection parameters.

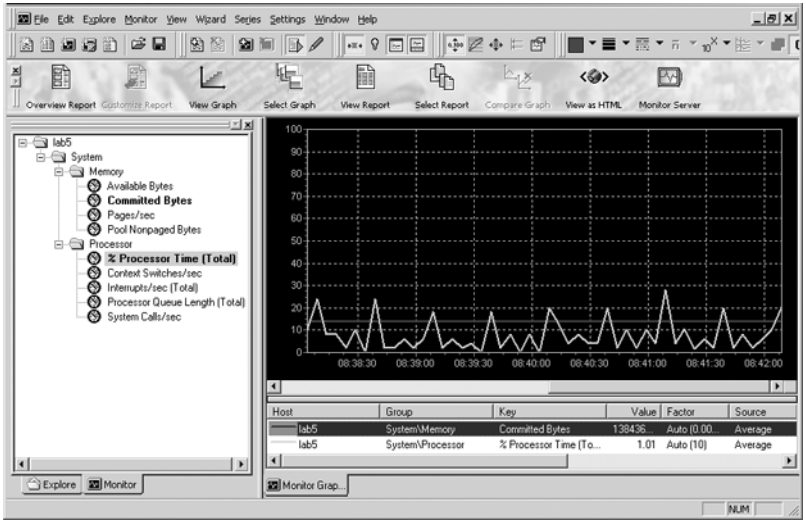
g In the next dialog box, select the performance counters you want to monitor.

Of particular interest are the processor and memory utilization of the server being tested.

h Click **Finish**.

The Performance Explorer opens a new window and displays real-time data for the performance counters you have selected.

Later, when you are running the load test, the monitor view will look similar to the following.



Adjusting the workload

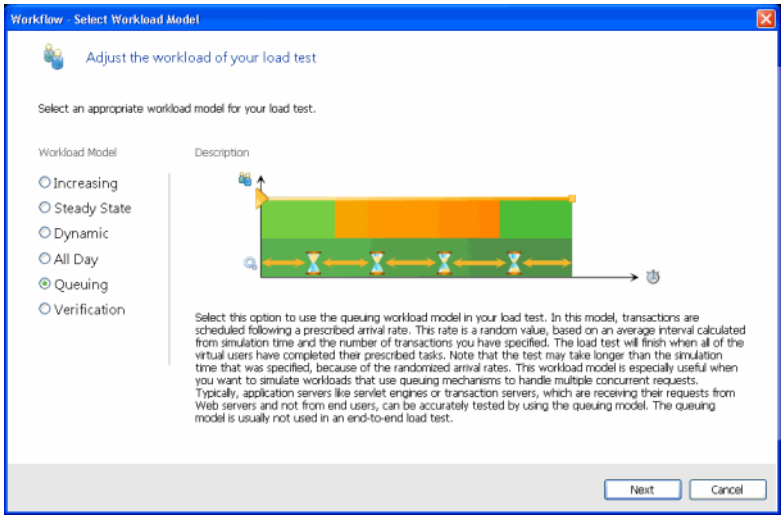
After you have confirmed your baseline, you now select your workload model and prepare the workload to run a full load test.

Adjusting the workload

Procedure To adjust the workload:

1 In the **Workflow** toolbar, click **Adjust Workload**.

The **Workflow - Select Workload Model** dialog box opens.



This dialog box shows all possible workload models, which can be selected.

2 Select **Queuing** workload model.

3 Click **Next**.

4 On the **Workflow Assign Agents** dialog box, select an option and click **OK**.

Silk Performer now opens the Workload Configuration dialog box.

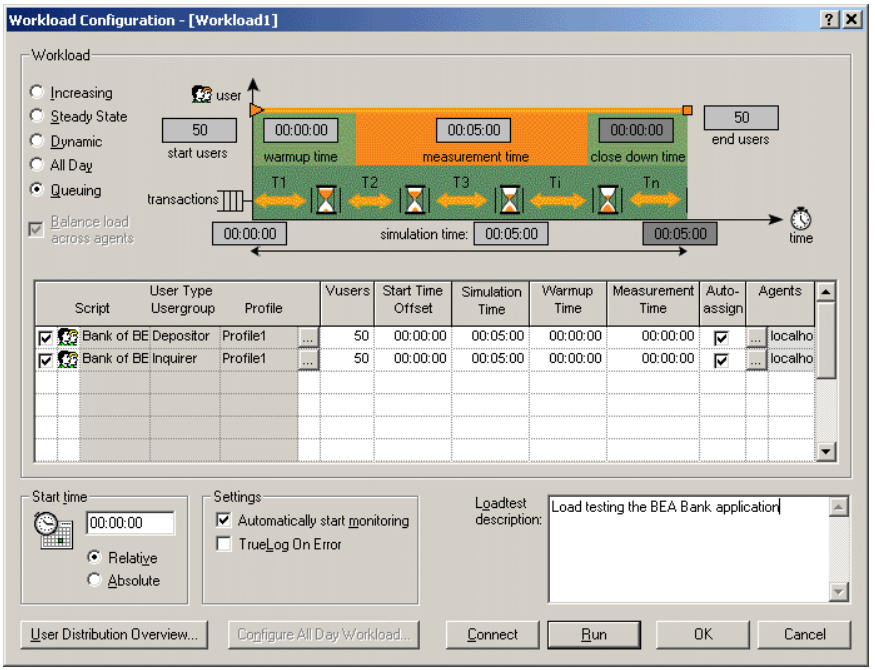
Running the load test

After you have adjust the workload, you run the full load test. The test script is run with multiple virtual users in order to test the BEA Bank applet. A large load test would need the appropriate testing environment to be set up in the local area network, including a full complement of agent computers to host the virtual users. For this tutorial, however, you use only your local computer.

Running the test

Procedure To run the load test:

1 The **Workload Configuration** dialogbox is open.



2 In the **VUsers** column, change the number of Depositor and Inquirer users to **50**.

3 For the Depositor and Inquirer user groups, change the simulation time to **300** seconds (5 minutes).

4 Disable all user groups that are declared in the JoltBankApp test script.

5 Click **Run**.

Silk Performer runs a load test with 100 virtual users, load testing the Bank applet. While the test is running, you can watch progress in the **Monitor** window and monitor the server being tested in the Performance Explorer.

What you have learned

In Tutorial 4, you learned how to:

- Find the test baseline
- Confirm the test baseline
- Set up server monitoring for your load test#
- Adjusting the workload
- Execute the load test

Tutorial 5: Viewing the results of your load test

Silk Performer incorporates **Performance Explorer**, a powerful graphing and analysis tool to help you work with your results information.

Performance Explorer provides a comprehensive array of results information, which can be displayed using advanced features for creating statistical reports and displaying performance results in real time generated graphics. Results information may vary, depending on the type of application and/or server being tested. In general it falls into the following categories:

- Response time information: This is the total time to process a given timer; it provides application performance information from the point of view of the client.
- Throughput information: This is the rate at which a given timer is processed on average by the server; it allows you to analyze performance from the point of view of the server.

Overview report of performed measurements

You can consult an HTML overview report that contains results of all performed measurements.

Exploring overview report

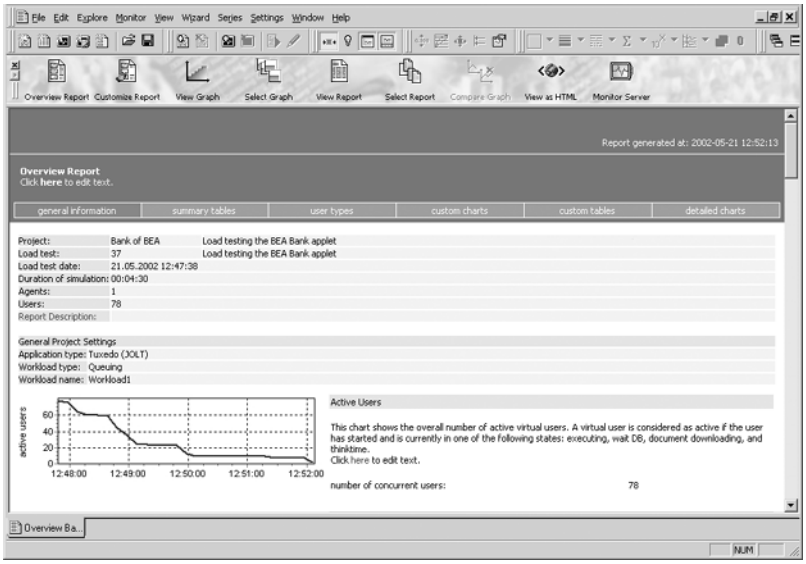
Procedure To explore overview report:

1 In the **Workflow** toolbar, click **Explore Results**.

The **Workflow - Explore Results** dialog box opens.

2 Click the **Performance Explorer** button.

Performance Explorer opens, and the HTML overview report is generated.



You may consult this report which includes short explanations of the displayed graphs. If you want you can view more detailed information.

Detailed response time measurements

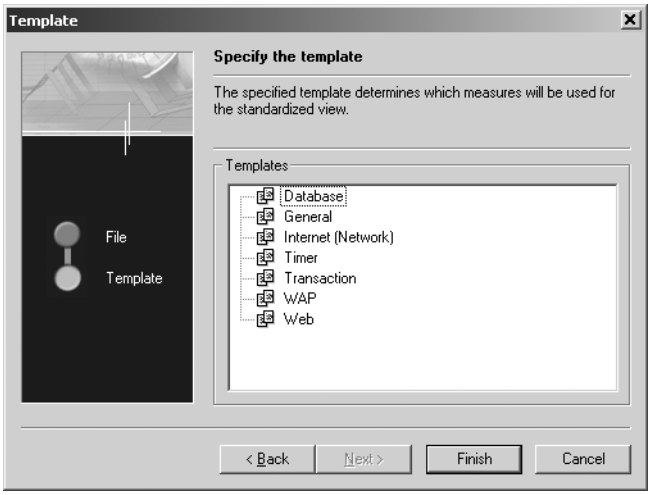
When you analyze performance of an application from the client point of view, you examine the response times for all the transactions and individual timers. This will enable you to find out how users will experience their interaction with server objects. A common goal in load testing is to find out if response times for all transactions remain below a specified, critical limit.

Response time details

Procedure To explore detailed response time measurements:

1 In the Performance Explorer **Workflow** bar, click the **Select Graph** button.

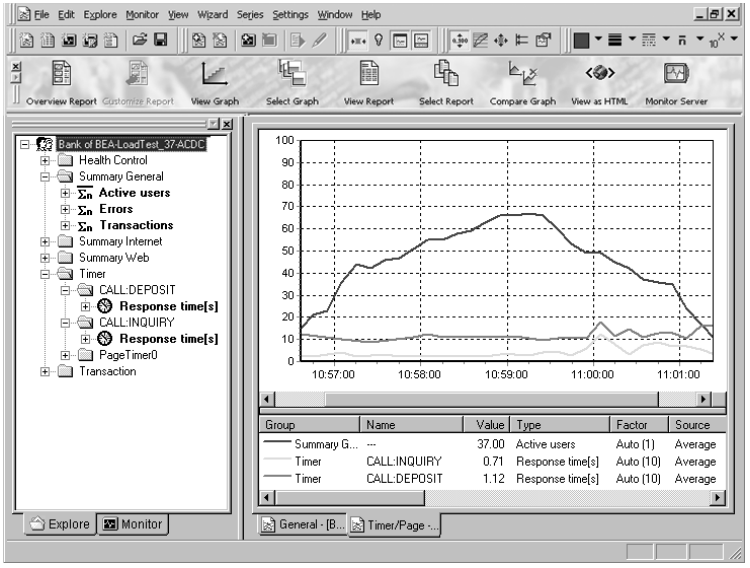
The **Template** dialog box opens.



2 In the **Templates** area, select the **Timer** option.

3 Click **Finish**.

The Performance Explorer opens a new chart with the number of active virtual users and the response time for the Jolt service calls displayed.



The load test that you ran in the previous tutorial lasted only a few minutes, so there are only a few measurements, which makes it difficult to draw any conclusions.

In real-life you will run load tests that last several hours, a few days, or even a week. These tests will provide results information that will allow you to analyze the performance of your server accurately.

Exploring throughput measurements

When you analyze the performance of an application from the point of view of the server, you examine the throughput rate. Throughput is the work done by the server within a specific time interval, for example, the number of requests that the server objects process within a second, a minute, or an hour. A common goal in load testing is to ensure that the throughput rate is above a specified, critical limit.

Throughput rate details

Procedure To explore throughput measurements:

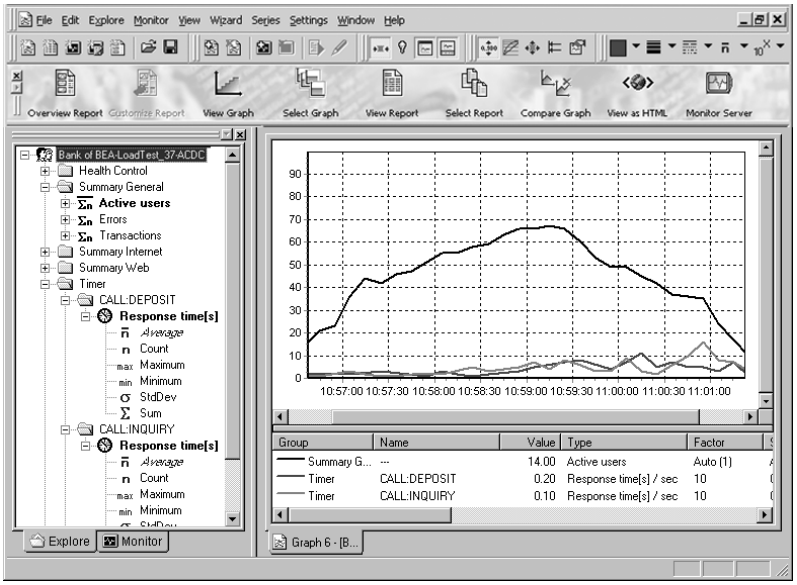
1. From the menu bar, select **File/New/Graph**.

   The Performance Explorer opens a new, blank graph.

2. In the **Explore** tree view, expand both the **Summary General** and the **Timer** measurement groups.
3. In the **Summary General** measurement group, drag the **Active users** measurement on to the graph.
4. In the **Timer** measurement group, select **CALL:DEPOSIT/Response time**, drag the **Count** measurement, and drop it on the graph.
5. In the **Timer** measurement group, select **CALL:INQUIRY/Response time**, drag the **Count** measurement, and drop it on the graph.

   You see a graph that displays the number of services performed by the server.

The graph should look similar to the following.



Again, since the load test that you ran lasted only five minutes, it is difficult to derive conclusions from the results.

What you have learned

In Tutorial 5, you learned how to:

- Explore response time measurements
- Explore throughput measurements

8

Load Testing IBM Mainframe Applications

Introduction

This chapter contains a series of tutorials that will help you begin load-testing IBM TN3270/5250 protocol-based applications with Silk Performer.

What you will learn

This chapter contains the following sections:

Overview

Each of the following tutorials provides step-by-step instructions for important Silk Performer tasks. You should perform these tutorials in order as each tutorial builds on the lessons learned in the previous tutorial.

The procedure for load testing IBM TN3270 and 5250 applications with Silk Performer is almost identical. While this tutorial focuses on testing a 3270 application, it can be applied to a 5250 application too. Differences between testing the two protocols will be noted within the tutorial where applicable.

You will learn how to do the following:

- Generate a test script by recording a mainframe system.
- Customize the recorded test script.
- Run the customized script in a TryScript run.

Prerequisites

Installing a terminal emulator

Before you can begin recording a mainframe application, you need to install a terminal emulator. Silk Performer records traffic between terminal emulation software and mainframe servers.

While Silk Performer supports most terminal emulators, it is recommended that you use Micro Focus RUMBA®, available from http://www.microfocus.com/products/RUMBA/.

Sample mainframe application

For this tutorial, you will need access to a mainframe application that offers telnet access. There are few public domain telnet servers in service currently, so you will need to create a connection to your own terminal emulation site.

Code page settings

If you will be testing an application that displays special characters, note that the host screens in TrueLog Explorer will not display special characters correctly. To correct this problem, you may have to adjust the terminal type and code page settings. To do this, select the *Active Profile* command from the Workbench's *Settings* menu. Then select *Terminal Client* under the *Record* tab in the left pane. In the *Host code page* drop-down listbox, select the appropriate code page for the characters that are to displayed (*IBM EBCDIC <country>* for 3270/5250 applications).

Tutorial 1: Creating a Script with the Recorder

The Silk Performer Recorder allows you to capture function calls that an IBM mainframe (TN3270/3270e/5250) terminal emulator performs. After you record function calls, you can save them as a test script. You can then easily customize and replay the script to simulate a large number of virtual users. Customizing and replaying a script is covered in Tutorials 2 and 3.

This tutorial includes the following steps:

- "Outlining a Project"
- "Setting up an Application Profile"
- "Modeling a Test Script"
- "Trying Out the Generated Test Script"

Outlining a Project

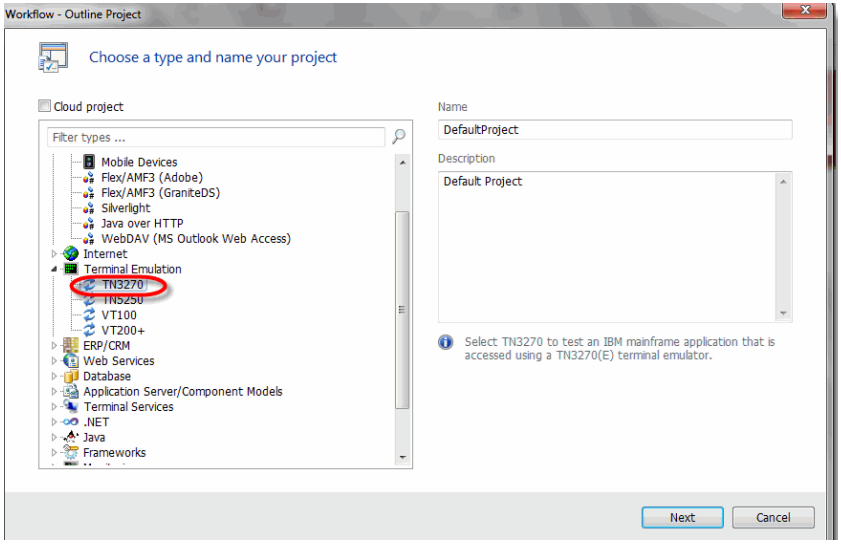
Before you can begin recording, you must outline a new project with the correct application type.

Outlining a project

Procedure To create a new project:

1. From Silk Performer's menu bar, select *File/New Project*.
   The *Workflow – Outline Project* dialog box opens.
2. Enter a project name, for example *TN3270_Tutorial*, in *Name*.
3. Enter an optional project description in *Description*.
4. Select *Terminal Emulation/TN3270* as the application *Type*.

   Alternative For recording a TN5250 application, you would select *Terminal Emulation/TN5250* as application type.
5. Click *OK* to confirm your settings.



Setting up an Application Profile

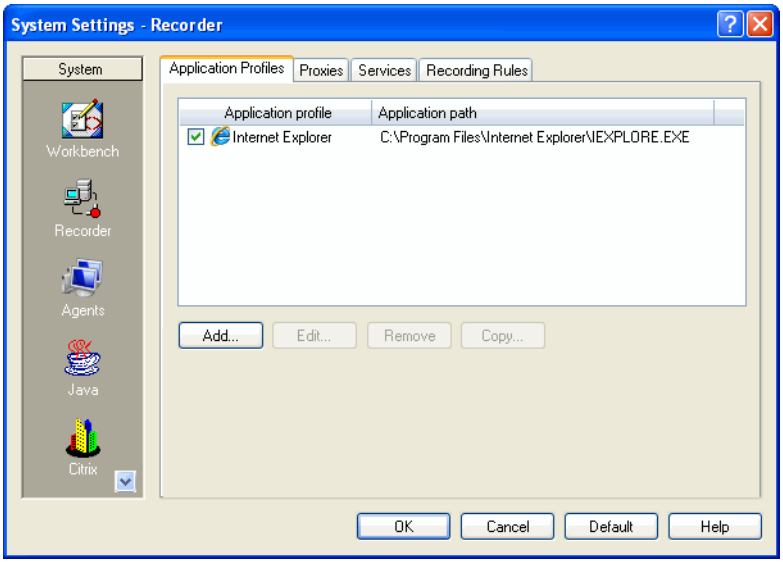
When you want the Silk Performer Recorder to capture and record the function calls that a client application performs, you must set up a profile for the client application. The profile specifies the type of client application and the type of function calls that are to be recorded. To record an IBM TN3270/5250 protocol-based application, the client application that needs to be recorded is a terminal emulator.

Setting up an application profile

Procedure To set up a profile for the client application:

1. From Silk Performer's menu bar, select *Settings/System*.
   The *System Settings – Workbench* dialog box opens.

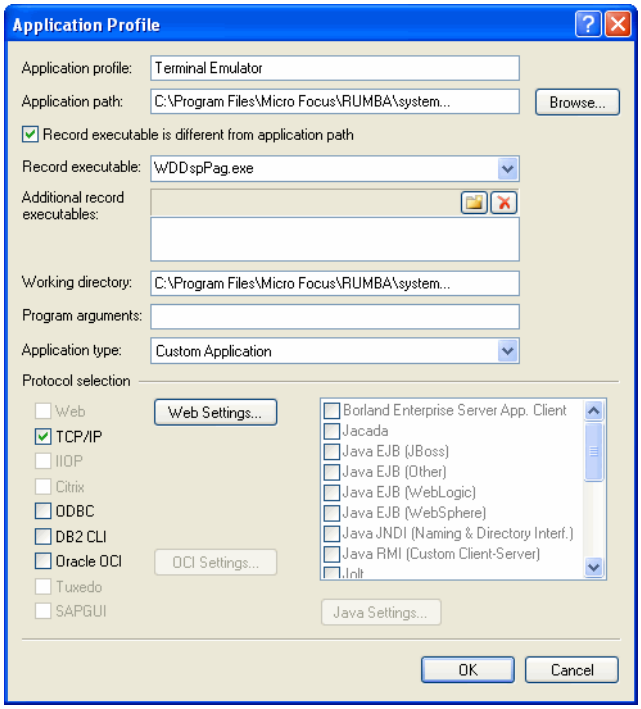
2 In the shortcut list on the left side, click the *Recorder* icon.
You see the following.



The list contains all the application profiles that are currently set up on your computer.

3 Click *Add*.

The *Application Profile* dialog box opens.



4. In the *Application profile* text box, enter a name for the application profile, for example *Terminal Emulator.*
5. Click *Browse* and locate the emulator in your RUMBA installation directory.
6. If you're using RUMBA as terminal emulator, check the *Record executable is different from application path* check box, then enter *WDDspPag.exe* in the *Record executable* field.
7. From the *Application type* list box, select the *Custom Application* option.
8. In the *Protocol selection* area, select the *TCP/IP* option.
9. Click *Web Settings*. On the following dialog box, select *ws2_32.dll* in the *WinSock* list box and click *OK* to confirm.
10. Click *OK* on the *Application Profile* dialog box.

    Silk Performer adds the new application profile to the profile list in the *System Settings – Recorder* dialog box.
11. Click *OK* to close the dialog box.

Modeling a Test Script

To test a mainframe application, the function calls that the client application performs need to be recorded and then described in a test script. The script ultimately enables any number of virtual users to perform actions similar to those performed during the recording session.

Alternative For recording an IBM TN5250 application, you'll have to connect to a 5250 application instead.

Note If you encounter any problems with the screen display (characters not displaying as they should), make sure that you've defined the correct code page settings. While the default settings will work in most cases, you may need country-specific settings for your application recording. Code page settings can be changed in Silk Performer's *Settings/Active Profile* menu, under the *Record/ Terminal Client* option in the left pane. Here you'll find the *Host code page* list (*IBM EBCDIC <country>* for recording TN3270/ 5250 applications).

Modeling a test script

Procedure To model a load-testing script:

1. In the *Workflow* bar, click *Model Script*.
   The *Workflow – Model Script* dialog box opens.
2. From the *Application Profile* list box, select *Terminal Emulator* (the profile you defined previously).
3. Click *OK*.

   Silk Performer opens the Recorder and starts the emulator.
4. Configure your emulator to connect to a mainframe application via a Telnet session.
5. Once connected, perform a few simple operations on the server. For example, login/log out, and execute a search query.
6. Close your terminal emulator.
7. Stop the Recorder, save the BDF file, and close the Recorder.
   Silk Performer automatically adds the script to your current load-testing project.

Trying Out the Generated Test Script

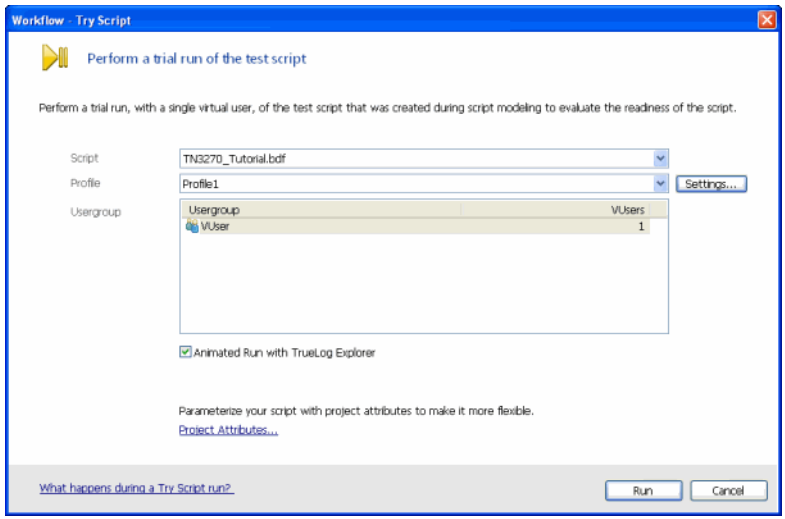
Now you will perform a trial run of the test script that you've created. The object of the trial run is to ensure that the script is error free, and that it will accurately

reproduce the interaction between the client application and the mainframe application.

Trying out the script

Procedure To try out the generated test script:

1. On the *Workflow* bar, click *Try Script*.
   The *Workflow – Try Script* dialog box opens.



2. From the *Script* list box, select the *TN3270_Tutorial* script (if it is not pre-selected).
3. Make sure that the *VUser* group is selected in the *Usergroup* list.
4. Enable the *Animated Run with TrueLog Explorer* option.
5. Click *Run*.

Silk Performer runs a single virtual user, called *VUser*, which performs all the actions that you performed during the recording session. You can watch the progress while the single-user test runs.

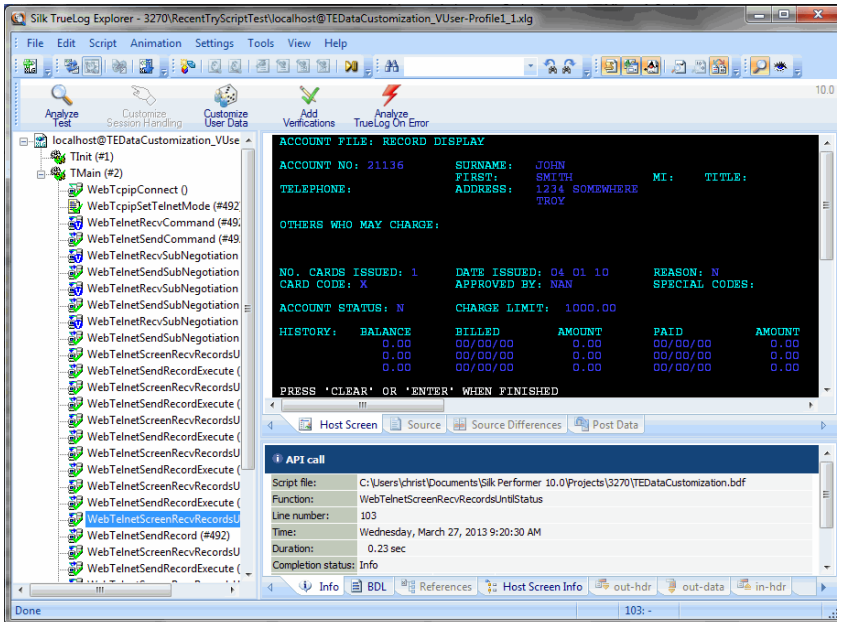
TrueLog Explorer supports the visualization of terminal emulation requests and responses in the same way it supports the visualization of HTTP client requests and HTTP/HTML server responses.

The three windows that are displayed with terminal emulation TrueLogs are:

- **Tree list** (left-hand pane) - Lists all terminal emulation API calls that were included in the test run.
- **Source window** (upper right-hand pane) - Displays the state of the terminal emulation application at each API node.

  Note TrueLog screengrabs are captured only during TryScript runs, not load tests.

- **Information window** (lower right-hand pane) - Displays data related to the most recent test run. The view tabs in this pane that are active and applicable to terminal emulation TrueLogs are *Info*, *BDL*, *in-data*, *out-data*, and *Host Screen Info*.



What you have learned

In Tutorial 1, you learned how to:

- Set up an application profile
- Record traffic between a terminal emulator and a server
- Generate a test script based on the recorded traffic
- Try out the generated test script

Tutorial 2: Customizing the Generated Test Script

Customizing the test script you have generated allows you to insert time measurements, verifications, screen synchronizations and other customizations. In this tutorial you will focus on inserting a timer, verifying screen contents, and looking briefly at some functions in the script.

> **Note** You must use Benchmark Description Language (BDL), Silk Performer's high-level scripting language, to customize some parts of the generated test script. Although knowledge of BDL is not required for this tutorial, it may be helpful for you to browse the introductory *BDL Reference* topics in Silk Performer Help for more information about BDL before you begin.

Adding a Timer to the Script

Now you will add a timer to our script so that you can measure the response time of a certain transaction during a test. This can be done visually in TrueLog Explorer.

Procedure To add a timer to the script:

1. Within TrueLog Explorer, Right-click the node within the API node tree menu that displays the search screen where you executed the sample search query.
2. Select *Start new Timer.*
3. In the *Start Timer* dialog box, enter the name *Search*, then click *OK*.
4. One or two nodes down, the results screen of your search will be displayed. Right-click this node and select *Stop Timer.*
5. On the *Select timer* dialog box, select the *Search* timer and click *OK*. Click *OK* again to confirm the script modification.

Adding Content Verification to the Script

Now that you've added a timer to measure how long a specified search transaction takes during replay, you need to verify that the results of the search are correct during replay. This can also be done visually in TrueLog Explorer, directly on the displayed host screen.

Procedure To add content verification to the script:

1. With TrueLog Explorer still open, select the same node to which you added the *Stop Timer*.
2. Look at the host screen displayed in the upper right-hand pane. For your content verification you may want to verify that, later during a test, the search query produces the same number of items as were identified during script recording. To do this, highlight text on the host screen that indicates the number of search items that were returned (for example, “ITEMS 1-4 OF 101”).
3. Right-click the highlighted text and select *Verify Selected Text*.
4. On the *Insert Verification Function* dialog box, leave the settings as they are and confirm insertion of the verification function by clicking *OK*.

For detailed information about verification and parsing functions, refer to the *TrueLog Explorer Help*.

Analyzing the Script

At this point you should close TrueLog Explorer and take a look at the modified script in Silk Performer. Some of the more interesting functions to consider are:

- *WebTelnetScreenRecvRecordsUntilStatus*:
  This function receives data (packets terminated by IAC EOR) from a given connection until the rendered screen contains a specified status.
- *WebTelnetSendRecord*:
  This function sends a Telnet record sequence to a terminal service server, which is a shortcut to <BinaryData> IAC EOR. In combination with *StrToHostCP*, which defines a user input command, the script sends user input to the terminal service server.
- *WebTelnetScreenVerifyText*:
  This function was inserted into the script via the content verification you created in TrueLog Explorer. It verifies rendered screen content in an active Telnet connection.
- *WebTelnetRecvRecordsUntilClose*:
  This function receives data (packets terminated by IAC EOR) from a given connection until the Telnet connection is closed. Silk Performer automatically inserts this function at the end of a transaction to synchronize script replay with the actual termination of the connection.

What you have learned In Tutorial 2, you learned how to:

- Insert a timer to measure response times

- Verify host screen content during replay
- Work with some common Telnet BDL functions

Tutorial 3: Replaying the Customized Script

Once you have created a test script by recording function calls that the client application performs and then customized the script, you can load test your server by replaying the script. To run a test, you must perform the following steps:

- Find the test baseline
- Confirm the test baseline
- Define the workload model
- Execute the test

Synchronization issues

In some cases, synchronization between replay and a host application may fail. Refer to Silk Performer's *Online Help* for detailed information about scripting manual synchronization functions.

9

Load Testing VT 100+ Applications

Introduction

This chapter contains a series of tutorials that will help you to start load-testing VT 100, and later, protocol-based applications with Silk Performer.

What you will learn

This chapter contains the following sections:

Overview

Each of the following tutorials provides step-by-step instructions for important Silk Performer tasks. You should perform these tutorials in order as each tutorial builds on the lessons learned in the previous tutorial.

The procedure for load testing VT 100 and VT 200+ applications with Silk Performer is almost identical. The only difference lies in the application type you have to select when outlining the project. Differences between testing the two protocols will be noted within the tutorial where applicable.

You will learn how to do the following:

- Generate a test script by recording a VT 100 application.
- Customize the recorded test script.
- Run the customized script in a TryScript run.

Prerequisites

Installing a terminal emulator

Before you can begin recording a VT 100+ application, you need to install a terminal emulator. Silk Performer records traffic between terminal emulation software and VT 100+ servers.

While Silk Performer supports most terminal emulators, it is recommended that you use Micro Focus RUMBA®, available from http://www.microfocus.com/products/RUMBA/.

Sample mainframe application

For this tutorial, you will need access to a mainframe application that offers telnet access. There are few public domain telnet servers in service currently, so you will need to create a connection to your own terminal emulation site.

Code page settings

If you are testing an application that displays special characters however, the host screens in TrueLog Explorer won't display the special characters correctly. To correct this problem, you may have to adjust the terminal type and code page settings. To do so, select the *Active Profile* command from Silk Performer Workbench's *Settings* menu. Then select *Terminal Client* under the *Record* tab in the left pane. In the *Host code page* list box, select the appropriate code page for the characters that will be displayed (any codepage except *EBCDIC* for VT 100+ applications).

Tutorial 1: Creating a Script with the Recorder

The Silk Performer Recorder allows you to capture function calls that a VT 100+ terminal emulator performs. After you record function calls, you can save them as a test script. You can then easily customize and replay the script to simulate a large number of virtual users. Customizing and replaying a script is covered in Tutorials 2 and 3.

This tutorial includes the following steps:

Outlining a Project

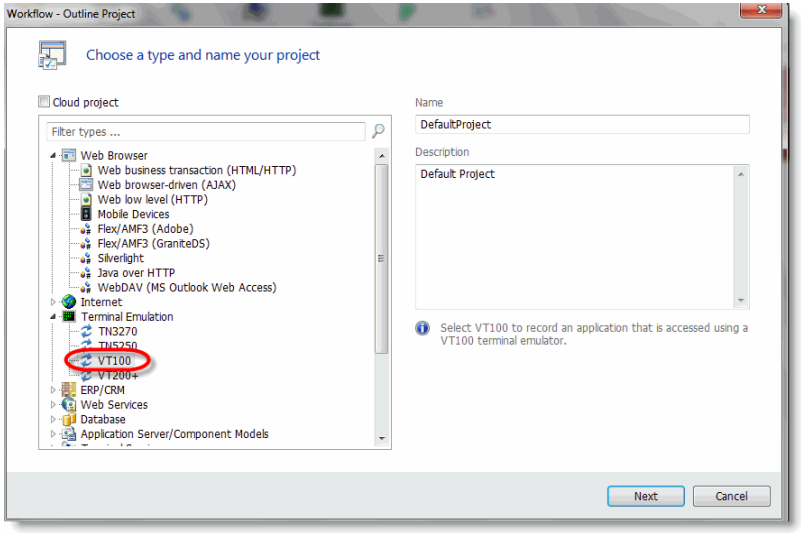
Before you can begin recording, you must outline a new project with the correct application type.

Outlining a project

Procedure To create a new project:

1. From Silk Performer's menu bar, select *File/New Project*.
   The *Workflow – Outline Project* dialog box opens.
2. Enter a project name, for example *VT100_Tutorial*, in *Name*.
3. Enter an optional project description in *Description*.
4. Select *Terminal Emulation/VT100* as the application *Type*.

   Alternative For recording a VT 200+ application, you would select *Terminal Emulation/VT 200+* as application type.
5. Click *Next* to confirm your settings.



Setting up an Application Profile

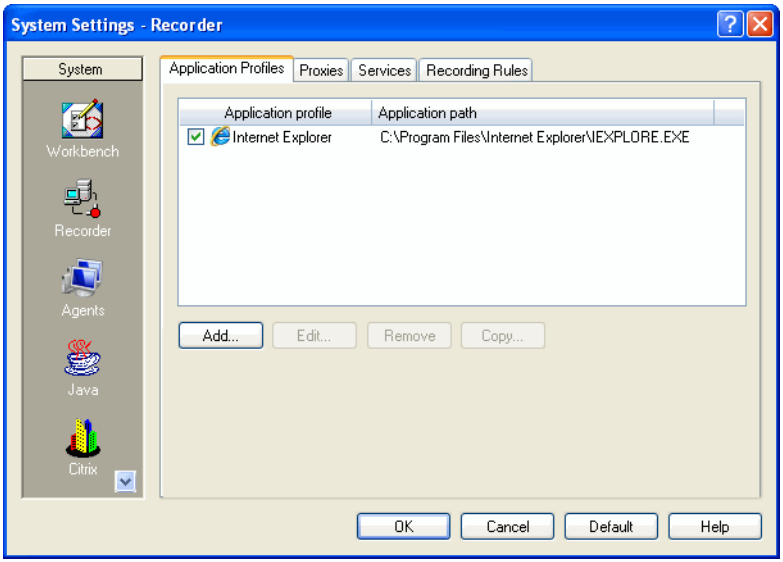
When you want the Silk Performer Recorder to capture and record the function calls that a client application performs, you must set up a profile for the client application. The profile specifies the type of client application and the type of function calls that are to be recorded. To record an VT 100+ protocol-based application, the client application that needs to be recorded is a terminal emulator.

Setting up an application profile

Procedure To set up a profile for the client application:

1. From Silk Performer's menu bar, select *Settings/System*.
   The *System Settings – Workbench* dialog box opens.

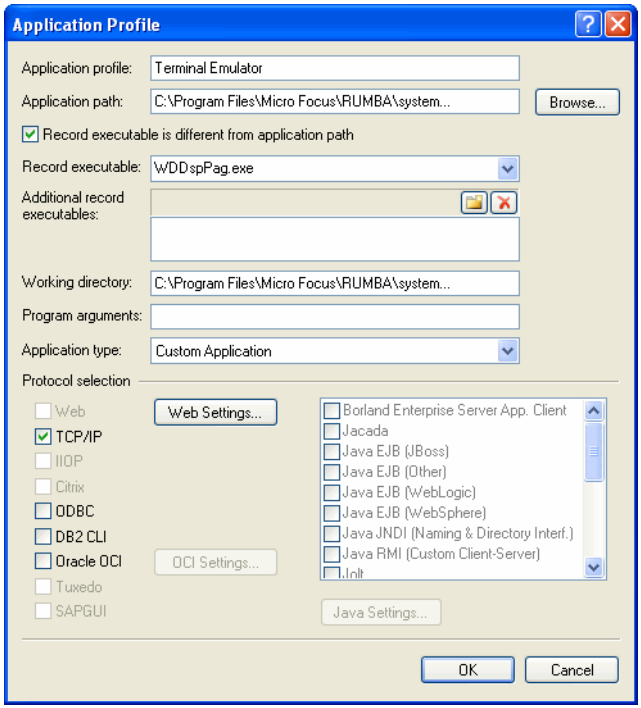
2 In the shortcut list on the left side, click the *Recorder* icon.
You see the following.



The list contains all the application profiles that are currently set up on your computer.

3 Click *Add*.

The *Application Profile* dialog box opens.



4 In the *Application profile* text box, enter a name for the application profile, for example *Terminal Emulator.*

5 Click *Browse* and locate the emulator in your RUMBA installation directory.

6 If you're using RUMBA as terminal emulator, check the *Record executable is different from application path* check box, then enter *WDDspPag.exe* in the *Record executable* field.

7 From the *Application type* list box, select the *Custom Application* option.

8 In the *Protocol selection* area, select the *TCP/IP* option.

9 Click *Web Settings*. On the following dialog box, select *ws2_32.dll* in the *WinSock* list box and click *OK* to confirm.

10 Click *OK* on the *Application Profile* dialog box.

Silk Performer adds the new application profile to the profile list in the *System Settings – Recorder* d

ialog box.

11 Click *OK* to close the dialog box.

Modeling a Test Script

To load test a VT 100+ application, the function calls that the client application performs need to be recorded and then described in a test script. The script ultimately enables any number of virtual users to perform actions similar to those performed during the recording session.

> **Note** If you encounter any problems with the screen display (characters not displaying as they should), make sure that you've defined the correct code page settings. While the default settings will work in most cases, you may need country-specific settings for your application recording. Code page settings can be changed in Silk Performer's *Settings/Active Profile* menu, under the *Record/ Terminal Client* option in the left pane. Here you'll find the *Host code page* list (any code page but *EBCDIC* for recording VT 100+ applications can be selected).

Modeling a test script

Procedure To model a load-testing script:

1. In the *Workflow* bar, click *Model Script*.
   The *Workflow – Model Script* dialog box opens.
2. From the *Application Profile* list box, select *Terminal Emulator* (the profile you defined previously).
3. Click *Start recording*.

   Silk Performer opens the Recorder and starts the emulator.
4. Configure your emulator to connect to a mainframe application via a Telnet session.
5. Once connected, perform a few simple operations on the server. For example, login/log out, and execute a search query.
6. Close your terminal emulator.
7. Stop the Recorder, save the BDF file, and close the Recorder.
   Silk Performer automatically adds the script to your current load-testing project.

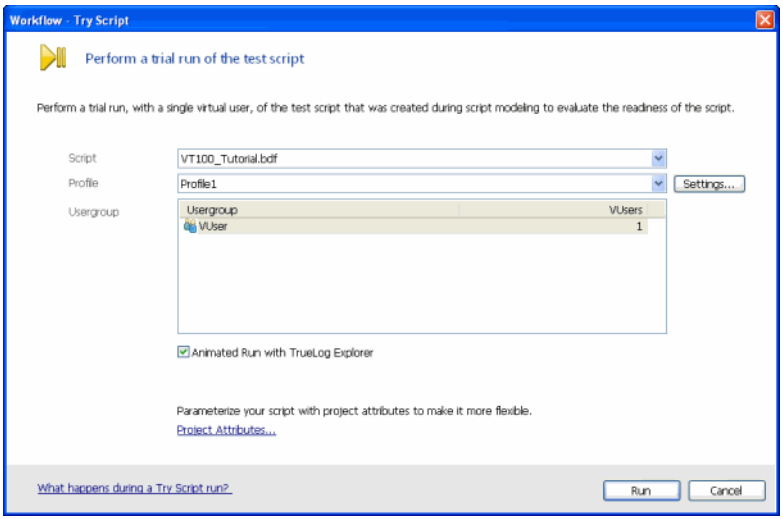
Trying Out the Generated Test Script

Now you will perform a trial run of the test script that you've created. The object of the trial run is to ensure that the script is error free, and that it will accurately reproduce the interaction between the client application and the Telnet application.

Trying out the script

Procedure To try out the generated test script:

1. On the *Workflow* bar, click *Try Script*.
   The *Workflow – Try Script* dialog box opens.



2. From the *Script* list box, select the *VT100_Tutorial* script (if it is not pre-selected).
3. Make sure that the *VUser* group is selected in the *Usergroup* list.
4. Enable the *Animated Run with TrueLog Explorer* option.
5. Click *Run*.

Silk Performer runs a single virtual user, called *VUser*, which performs all the actions that you performed during the recording session. You can watch the progress while the single-user test runs.

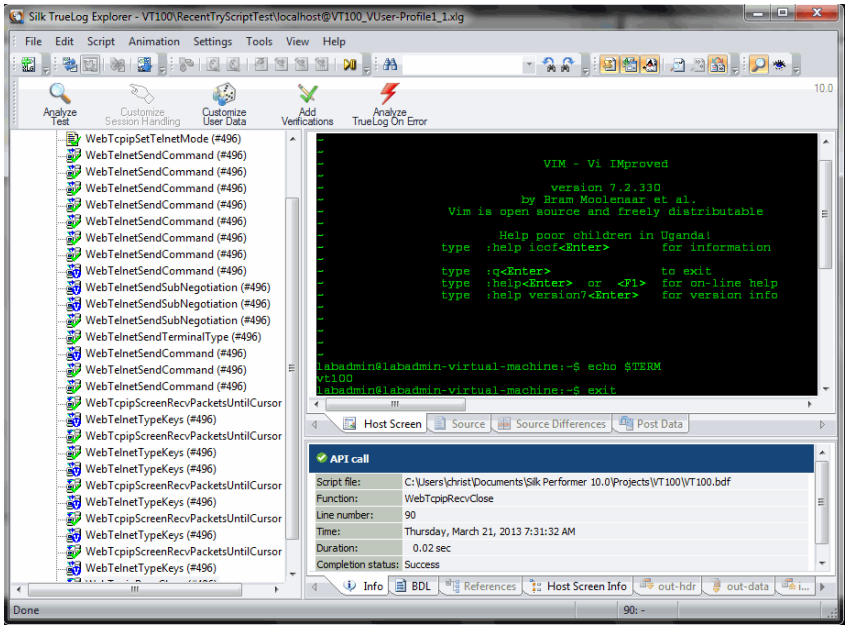
TrueLog Explorer supports the visualization of terminal emulation requests and responses in the same way it supports the visualization of HTTP client requests and HTTP/HTML server responses.

The three windows that are displayed with terminal emulation TrueLogs are:

- **Tree list** (left-hand pane) - Lists all terminal emulation API calls that were included in the test run.
- **Source window** (upper right-hand pane) - Displays the state of the terminal emulation application at each API node.

  Note TrueLog screengrabs are captured only during TryScript runs, not load tests.

- **Information window** (lower right-hand pane) - Displays data related to the most recent test run. The view tabs in this pane that are active and applicable to terminal emulation TrueLogs are *Info*, *BDL*, *in-data*, *out-data*, and *Host Screen Info*.



What you have learned In Tutorial 1, you learned how to:

- Set up an application profile
- Record traffic between a terminal emulator and a server
- Generate a test script based on the recorded traffic
- Try out the generated test script

Tutorial 2: Customizing the Generated Test Script

Customizing the test script you have generated allows you to insert time measurements, verifications, screen synchronizations and other customizations. In this tutorial you will focus on inserting a timer, verifying screen contents, and looking briefly at some functions in the script.

> **Note** You must use Benchmark Description Language (BDL), Silk Performer's high-level scripting language, to customize some parts of the generated test script. Although knowledge of BDL is not

required for this tutorial, it may be helpful for you to browse the introductory *BDL Reference* topics in Silk Performer Help for more information about BDL before you begin.

Adding a Timer to the Script

Now you will add a timer to our script so that you can measure the response time of a certain transaction during a load test. This can be done visually in TrueLog Explorer.

Procedure To add a timer to the script:

1. Within TrueLog Explorer, Right-click the node within the API node tree menu that displays the search screen where you executed the sample search query.
2. Select *Start new Timer.*
3. In the *Start Timer* dialog box, enter the name *Search*, then click *OK*.
4. One or two nodes down, the results screen of your search will be displayed. Right-click this node and select *Stop Timer.*
5. On the *Select timer* dialog box, select the *Search* timer and click *OK*. Click *OK* again to confirm the script modification.

Adding Content Verification to the Script

Now that you've added a timer to measure how long the specified search transaction takes during replay, you need to verify that the results of the search are correct during replay. This can also be done visually in TrueLog Explorer, directly on the displayed host screen.

Procedure To add content verification to the script:

1. With TrueLog Explorer still open, select the same node to which you added the *Stop Timer.*
2. Look at the host screen displayed in the upper right-hand pane. For your content verification you may want to verify that, later during a test, the search query produces the same number of items as were identified during script recording. To do this, highlight text on the host screen that indicates the number of search items that were returned (for example, "ITEMS 1-4 OF 101").
3. Right-click the highlighted text and select *Verify Selected Text*.

4 On the *Insert Verification Function* dialog box, leave the settings as they are and and click *OK*. Click OK on the following dialog box to confirm insertion of the verification function.

For detailed information about verification and parsing functions, refer to the *TrueLog Explorer Help*.

Analyzing the Script

At this point you should close TrueLog Explorer and take a look at the modified script in Silk Performer. Some of the more interesting functions to consider are:

- *WebTcpipScreenRecvPacketsUntilCursor*:
  This function receives packets from a Telnet server until cursor is detected at a specified position. This function is used for synchronizing replay and the actual Telnet server response.
- *WebTcpipSend*:
  This function sends a string to the remote host.
- *WebTelnetTypeKeys*:
  This function simulates a user input to a Telnet server. This function sends alphanumeric character data that is specified at a specific rate and optionally verifies data echoed by the server.
- *WebTelnetScreenVerifyText*:
  This function was inserted into the script via the content verification you created in TrueLog Explorer. It verifies rendered screen content in an active Telnet connection.
- *WebTcpipRecvClose*:
  This function retrieves data from the remote host until the connection is closed by the server. Silk Performer automatically inserts this function at the end of a transaction to synchronize script replay with the actual termination of the connection.

What you have learned

In Tutorial 2, you learned how to:

- Insert a timer to measure response times
- Verify host screen content during replay
- Work with some common Telnet BDL functions

Tutorial 3: Replaying the Customized Script

Once you have created a test script by recording function calls that the client application performs and then customized the script, you can load test your

server by replaying the script. To run a load test, you must perform the following steps:

- Find the test baseline
- Confirm the test baseline
- Define the workload model
- Execute the test

Synchronization issues

In some cases, synchronization between replay and a host application may fail. Refer to Silk Performer's *Online Help* for detailed information about scripting manual synchronization functions.

A

Setup and Configuration

Introduction

This appendix comprises information and step-by-step instructions on how to set up and configure the sample applications that are used in the tutorials.

What you will learn

This chapter contains the following sections:

For “Load Testing a CORBA Application”, the Bank application is one of the automatically installed samples that accompany VisiBroker. To set up VisiBroker, follow the setup instructions given in the VisiBroker documentation.

For “Load Testing a TUXEDO Application”, the Bank application is one of the automatically installed samples that accompany BEA TUXEDO. To set up the BEA TUXEDO System follow the setup instructions given in the TUXEDO documentation.

For “Load Testing a Jolt Application”, follow the setup instructions given in the TUXEDO and Jolt documentation.

Setting up the Person1 application

This section provides step-by-step guidance through the configuration of the database sample application, Person1. You need this application to perform the steps described in the tutorials in "Load Testing a Database through ODBC".

As your database you will use the Microsoft Data Engine (MSDE). This is a client/server data engine that provides local data storage on a smaller computer system (such as a single-user computer or small workgroup server) and is compatible with Microsoft SQL Server 2000.

Installing a database

You can download MSDE from http://www.microsoft.com/downloads/. After completing the download, follow the instructions of the setup wizard.

After you have installed your database server, you need to set up an ODBC data source on your Silk Performer controller computer. The data source stores information about the database server and makes this information available to applications.

Setting up a data source

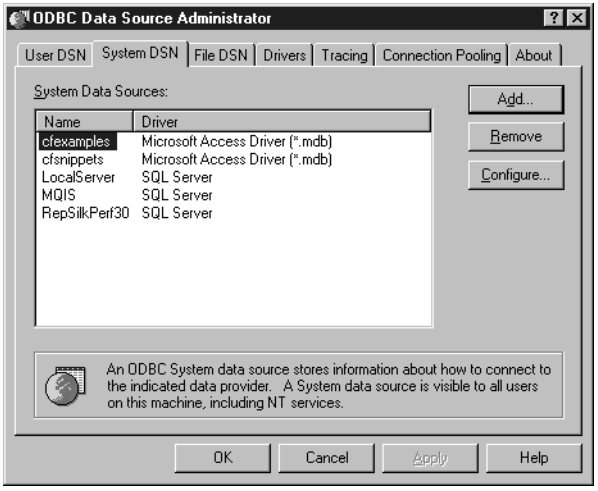
Procedure To set up an ODBC data source:

1. On your Silk Performer controller computer, select **Settings/Control Panel/ODBC Data Sources** from the Windows **Start** menu.

   The **ODBC Data Source Administrator** dialog box opens at the **User DSN** tab.

2. Click the **System DSN** tab.

   You see the following dialog box.



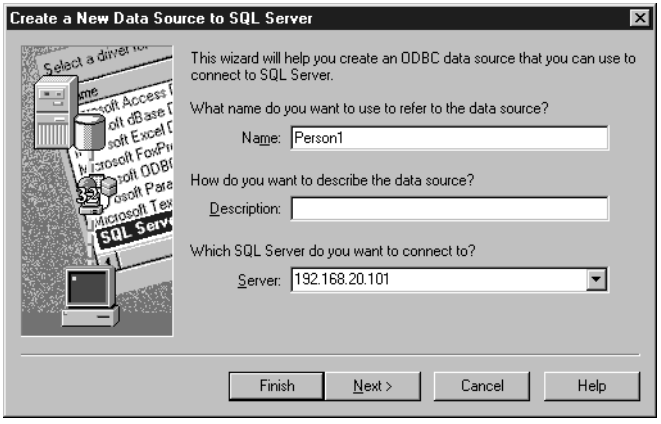
3. Click **Add**.

The **Create New Data Source** dialog box opens.

4 In the list, select the **SQL Server** option.

5 Click **Finish**.

The **Create a New Data Source to SQL Server** dialog box opens.



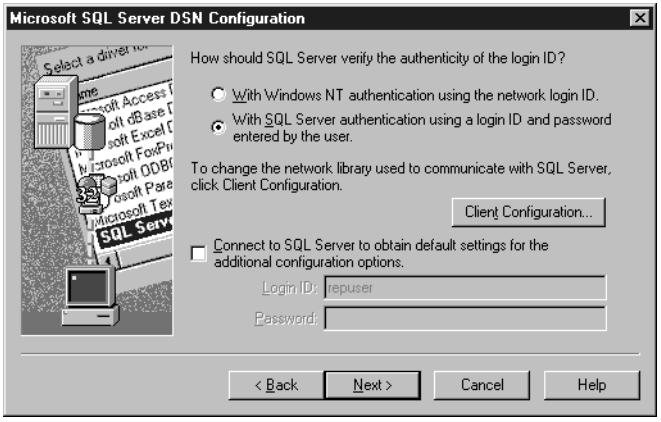
6 In the **Name** text box, enter **Person1**.

This is the name by which the ODBC data source will be identified.

7 In the **Server** text box, enter the host name or IP address of the computer on which you have installed MSDE.

8 Click **Next**.

You see the following dialog box.



9 Select the **With SQL Server authentication using a login ID and password entered by the user** option.

10 Click **Next**.

11 In all remaining dialog boxes, just retain the default settings and click the appropriate button to proceed (**OK**, **Next**, or **Finish**).

The new data source is added to the list in the **System DSN** tab of the **ODBC Data Source Administrator** dialog box.

12 In the **ODBC Data Source Administrator** dialog box, click **OK**.

The Person1 application is now configured properly. When you start the application, enter the following data in the Login dialog box.

Field	Value
Data source	Person1
user name	repuser
password	repuser

Setting up the PersonPB application

This section provides step-by-step guidance through the configuration of the sample Oracle database application called PersonPB. You need this application to perform the steps described in the tutorials in "Load Testing an Oracle OCI7 Database".

Installing Oracle

Procedure To install the Oracle software:

1 On the computer that you want to use as your sever, install the Oracle server software.

Refer to the Oracle documentation for a detailed description of the setup process.

2 On your Silk Performer controller computer, install the Oracle client software.

Refer to the Oracle documentation for a detailed description of the setup process.

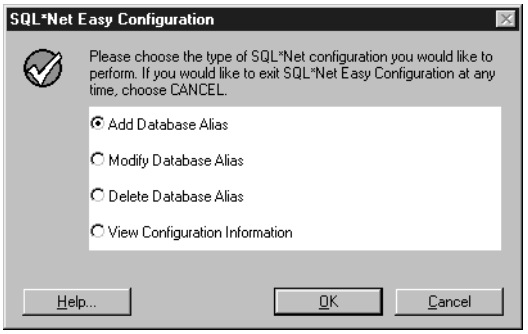
After you have installed the Oracle server and client software, you need to set up a database alias on your Silk Performer controller computer. The alias stores information about the database server and makes this information available to applications.

Setting up an alias

Procedure To set up a database alias:

1 On your Silk Performer controller computer, select **Programs/Oracle/ SQL Net Easy Configuration** from the Windows **Start** menu.

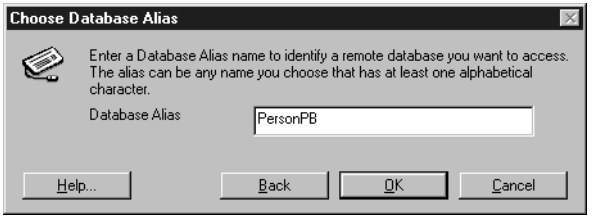
The **SQL *Net Easy Configuration** dialog box opens.



2 Select the **Add Database Alias** option.

3 Click **OK**.

The **Choose Database Alias** dialog box opens.

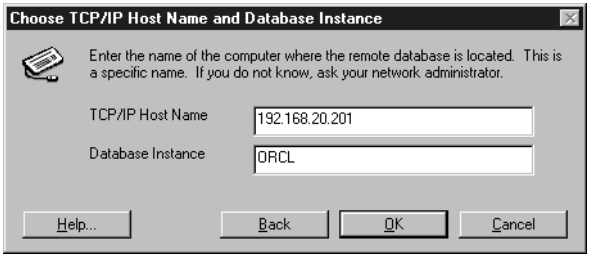


4 In the **Database Alias** text box, enter **PersonPB**.

This is the name by which the alias will be identified.

5 Click **OK**.

The **Choose TCP/IP Host Name and Database Instance** dialog box opens.

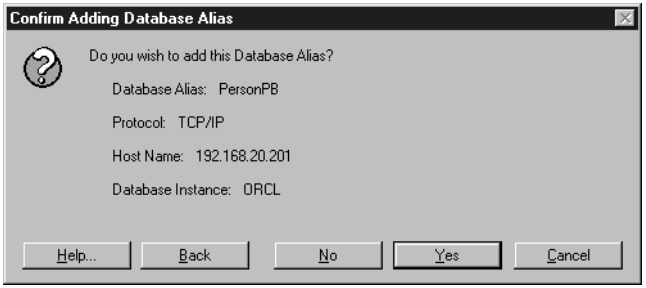


6 In the **TCP/IP Host Name edit** text box, enter the host name or IP address of the computer on which you have installed the Oracle server software.

7 In the **Database Instance** text box, retain the default entry **ORCL**.

8 Click **OK**.

The **Confirm Adding Database Alias** dialog box opens.



9 Double-check the information you have entered.

If you want to change any of the settings, click **Back**.

10 Click **Yes**.

11 In the **SQL *Net Easy Configuration** dialog box, click **Cancel**.

The PersonPB application is now configured properly. When you start the application, enter the following data in the Login dialog box.

Field	Value
Alias	PersonPB
user name	scott
password	tiger

The user “scott” with the password “tiger” is set up by default by the Oracle software.