	IBM Rational Quality Solutions for Siebel Test Automation Demo Movie Script	Rational. software

IBM Rational Quality Solutions for Siebel Test Automation v6.1

Demo Movie Script

AUTHOR	REVISION DATE	REVISION DESCRIPTION
Dennis Schultz	07/21/05	Initial Version

Table of Contents

Assumed Context

- It is assumed that the audience for this demo movie already has a basic knowledge of Rational Functional Tester including the scripting languages, recording paradigm, and key features. The audience should also be familiar with Rational Performance Tester.
- It is also assumed that the audience has recently been presented the IBM Software Quality Solutions for Siebel sales presentation.

Benefits

- Increase client satisfaction
  - Validate business transactions prior to deployment
  - Execute automated tests in a fraction of the time spent testing manually
- Maximize ROI with error-free deployments
  - One toolset to test Siebel as well as your other enterprise applications
- Ensure systems perform, function, and scale on day one
  - Simplified test creation environment for novice testers and business analysts
  - Tests can be extended with industry-standard Java language
  - Powerful IDE for advanced test development and debugging

Introduction

Slide 1

Welcome to the IBM Rational Software Quality Solutions for Siebel Test Automation Demo. This demo will show you how the power and benefits of IBM Rational software quality solutions can now be extended to testing Siebel applications through the Software Quality Solutions for Siebel Test Automation products from IBM Rational software. This demo assumes you already have at least a basic understanding of Rational Functional Tester and Rational Performance Tester.

Slide 2

The primary goals of the Software Quality Solutions for Siebel Test Automation products are to help you deliver your business-critical Siebel 7.7 applications with higher client satisfaction while helping you realize a higher return on your testing investment. We do this by providing your teams with the tools necessary to ensure your systems perform, function, and scale to the levels you need on day one.

Functional Testing

Start RFT

Let's start by examining how Rational's Software Quality Solutions for Siebel can help you ensure the functionality of your Siebel implementation. Siebel's own research indicates that the typical Siebel customer spends approximately 25% of the total project costs on testing. You may already know how Rational Functional Tester helps you automate functional testing of your Java, Web, and Microsoft .NET applications through a simple, easy to use record/playback model rich with powerful features that help make your tests extendable and robust. Now those same features are available for testing Siebel applications as well.

Rational Functional Tester supports two industry-standard scripting options. Java scripts are created using the Eclipse environment. Visual Basic .NET scripts are created using the Microsoft Visual Studio environment. Today we will be using the Java/Eclipse environment.

Start Recording

Let's jump right in and record a test. To do so, you will click the red, record button. Your test will involve adding Customer records to the ACME Anvil Corp. Account, so you will name your test AddACMEContacts.

As soon as you enter record mode, the IDE is minimized and the recording toolbar pops up. This toolbar provides several tools that you will use while recording your test. You could have your script launch the Siebel Call Center for you, but to keep things simple, I have opened the Call Center already.

Navigate to Contacts List View

First you will navigate to the Contact List view. Notice how Functional Tester confirms your interactions with the application in the record toolbar. Also notice how, with the Rational Functional Tester Extension for Siebel Test Automation, Rational Functional Tester is able to interact with custom Siebel High Interactivity or "HI" and Standard Interactivity or "SI" objects.

Enter new Contact info

To enter a new Contact, you will use the Contact Form Applet at the bottom of the page. The contact you are entering is with the ACME Corporation, so you use the Account Association Applet to find and select ACME.

Returning to the Contact Form Applet, you enter a Last name and First name for the contact. Simply automating the entry of this one contact will save you a great deal of time, but you recognize already that you will want to enter many Contacts in this same manor. Wouldn't it be nice if you could reuse this test to do that? Well, with Functional Tester's wizard for data-driven testing, you can do just that. The wizard enables you to identify all the data you may want to vary and then creates all the code necessary to iterate through the datasets you provide. You simply point the wizard at the fields you

wish to drive from your data source. You can use the wizard to begin creating your data source during recording or you can associate it to data sources that already exist in your test project.

Navigate to Account details screen

Now that you have entered a new contact record into the system, you want to verify that it has been associated to the ACME Anvil account and that this contact appears on the Account Details screen. Here, you use the Recent Records link from the Accounts Home screen to check the details.

Insert Data verification point on Contacts list

The Rational Functional Tester Extension for Siebel Test Automation not only enables you to navigate through Siebel custom controls, it also provides the ability to see inside those objects as well. From the toolbar, you can simply drag and drop the Verification Point wizard onto the object you want to test, in this case, the contacts Siebel List object.

Functional Tester provides multiple types of verification points. The two most common types are Data and Properties tests. With Properties verification points, you can verify a number of common properties including Siebel-specific properties such as RepositoryName and UIName.

With a Data Verification Point, you can test the data contained in the object. Data VP's are very flexible allowing you to test the entire table's data, specific rows or even individual cells.

Remember that since you are using a datapool for the contact details entered earlier, the data in this table will be dependent on the datapool data as well. Right from the GUI, you can configure your data verification point to be dynamic and base its expected results off datapool values.

Stop Recording

This completes your test scenario, so you stop recording by clicking the stop button on the toolbar.

The script and RFT UI

The Functional Tester script for a Siebel application looks much like a test script for any other supported environment. The script is captured in industry-standard Java or Visual Basic .NET. Instructions are in an object-dot-action notation.

Functional Tester stores the definitions of all objects it encounters in a Test Object Map. This map centralizes the information on how an object will be identified at playback. If the properties of an object were to be changed in a subsequent build, Functional Tester's ScriptAssure technology, through its advanced fuzzy logic, would enable the script to playback even where most other automation tools would fail to find the correct object on the screen. If the definition of the object does need to be updated, the Test Object Map provides a single point of maintenance to do that.

The development environments are also "Siebel-aware". For example, code-savvy testers may want to write tests by hand instead of using the recorder as you just did. Functional Tester's Code Assist provides context-sensitive completions for Siebel test objects, properties, and methods. The API interface specifications are even integrated into the Test Object Map.

Notice the test datapool editor. This is the list of data that will be used for script playback. Data can be added directly or imported from external sources such as CSV files. The data you entered while recording this test has been populated into this datapool. You have already created a more complete set of data, so you will associate that datapool with your new test.

Script Playback

Now you are ready to play back the test. First you specify a log file to store the results. Next, you indicate that you wish this test to be repeated for each row in the datapool associated with this test.

As script playback proceeds, you can see your interactions being repeated, this time using data from your datapool.

Once the first iteration has completed, the second iteration will begin.

Skipping ahead, you can see the third and final contact has been added.

Results Log Review

Upon completion, Functional Tester displays the test results log. You have chosen to generate your results in HTML form, but you could also log results into simple text files or Rational TestManager test logs as well.

Here you see that the test has passed. Using the quick-links on the left side, you can easily jump to the most interesting parts of the test log. Each verification point event enables you to drill down to the results. Here you can see how extracting data from the datapool enabled you to create a verification point tied to the same data input by the script for that iteration.

Performance Testing

Let's now turn our attention to ensuring the performance and scalability of your Siebel deployments. With the Rational Performance Tester Extension for Siebel Test Automation, you are empowered to automate the scalability and performance testing of your entire Siebel architecture, enabling you to isolate performance bottlenecks and better perform capacity planning.

Rational Performance Tester

Rational Performance Tester is an advanced performance testing solution designed for day-one productivity. Tests are not captured in some proprietary script code. Performance Tester provides a simplified, graphical test interface allowing the novice tester to be productive on day-one, yet

providing all the power that an advanced performance tester might want. Performance Tester is just another perspective in the same test environment used by Functional Tester, java-scripting.

Performance Tester follows a similar recording model to that of Functional Tester. Recording a test scenario of adding a contact would produce a performance test like you see here. With one exception, each top-level node in the test represents a Siebel screen encountered during the recording. The node names are extracted from the Siebel information communicated with the server.

Expanding each of the screen nodes exposes the detail behind it. This detail is often not required by the novice tester, but is readily available for the advanced tester. Each screen or page is typically made up of one primary request and secondary requests. The protocol viewer enables the tester to see a visual representation of the item selected as well as the detail of the request and response header and content.

Message Bar

The one node that does not represent a screen is the first node – named Message Bar. Siebel clients periodically request information from the server to display in the message bar at the bottom of the Siebel web interface. The Rational Performance Tester Extension for Siebel automatically includes these periodic requests into your test with no extra effort on your part. The rate at which the message bar data is requested is configurable. Upon playback, Performance Tester will accurately represent the system load generated by the message bar requests.

WriteRecord

There is a great deal of information transferred between client and server in a Siebel environment. The Performance Tester Extension for Siebel identifies and manages all but the data you provided in forms. By unmasking the Siebel references, you can see Performance Tester has done a great deal of correlation without any work on your part. This allows you to focus on managing the data you want to use to accurately represent your system workload. Here, for example, you see only the four fields you completed when creating the new Contact. These are values you may want to extract from a datapool, much like you did during functional testing. Again, this is all possible through the Graphical Test Editor with no script coding required.

AddContact Test

While conducting performance tests, you may want to verify the responses to your requests are as expected by introducing Verification Points into your test. There are several types of verification points in Performance Tester: Page Title, Response Code, and Response Size VPs verify the basic attributes of the response. Content VPs allow you to examine the data returned in the response message. The Rational Performance Tester Extension for Siebel has predefined a number of the most common Siebel error and status messages. You can simply choose the messages you want to detect and indicate that the verification point should fail if any one of them is found.

You can even define your own messages. The text you enter can be case sensitive and can even include regular expressions for pattern matching. ACME has implemented custom error codes

consisting of 12 digit numbers. This pattern will detect if a custom error message containing one of those codes is received.

Content Verification Points can be inserted at the test, screen, or individual response levels.

EndOfMonthLoadTest Schedule

Performance test playback is based around a Test Schedule. A Test Schedule enables you to define the workload mix of your system under test. User Groups are used to classify the users of your system. Here, you have defined three groups, Sales Managers, Sales Reps, and Shipping. Sales Reps make up 50% of the total users, Sales Managers 20% and Shipping 30%. The total number of users can be changed when the Schedule is invoked. In fact, additional users can be added at anytime during a test run.

Each User Group will perform different functions. For example, the Shipping department is only concerned with searching for urgent orders that need to be shipped. Each shipping clerk will run this query three times.

Sales Reps sometimes add a contact, other times add an account. They are about three times as likely to add a contact as an account. The random selector with the weighted blocks takes care of implementing that randomness for you.

At the end of the month, the only thing Sales Managers are concerned with is researching opportunities that may close.

Playback of Performance Test

When a test schedule is invoked for playback, Performance Tester computes the number of emulated users for each group based on the percentages and the total number of users selected. Multiple agent machines can be employed for large tests that require additional driver power.

Overview Report

The default performance report starts up immediately. The colored blocks across the top of the Overview tab indicate the state of the test operation. As the test gets underway, the Overview bar-graphs give you a quick indication how the test is going.

The performance reports are "live" and are updated constantly as the test proceeds. This means you get instant feedback on your performance test, potentially saving you hours of time wasted on bad runs.

These reports are highly customizable, although the default reports give the average user most everything needed.

Page Throughput

Page Throughput shows number of users and delays between requests and responses. On the right, this report shows the number of concurrently active users at a given time. On the left, you see the page attempt and hit rates. On this graph you want to see the two in sync. Large gaps indicate delays between requests and responses. This is a very high level overview of overall server response.

Response vs. Time Summary

The Response vs. Time Summary graphs summarize the average response times for all pages and elements distributed across the duration of the entire test run. This is slightly more detailed than the Page Throughput.

Page Performance

The Page Performance graph shows the average page response time for each page over the course of the test run. From this graph, you can quickly see that the Login page is taking considerably longer to load than all others – on the order of 28 seconds on average.

Response vs. Time Details

The Response vs. Time Details enables you to drill down to see how individual pages response time varies over the duration of the test run. This graph quickly alerts you to an unhealthy trend. Not only was your Login page slower than you would like, but it appears that each successive user that attempts to log in experiences poorer performance than the last. You have quickly identified a performance bottleneck with the Login page that should be investigated further.

Summary

Thanks for taking the time to see what IBM Rational has to offer in the Software Quality Solutions for Siebel Test Automation. Rational wants to help you deliver your business-critical Siebel 7.7 applications with higher client satisfaction while helping you realize a higher return on your testing investment by providing you with the tools necessary to ensure your systems perform, function, and scale to the levels you need on day one.