ISSN:0975-9646

Automated Functional Testing Using IBM Rational Robot

A.Chakrapani, K.V.Ramesh
Department of Computer Science and Engineering, GITAM University, Visakhapatnam, India.

Abstract- **In the past, most software tests were performed using manual methods. This required a large staff of test personnel to perform expensive and time-consuming manual test procedures. Owing to the size and complexity of today's advanced software applications, manual testing is no longer a viable option for most testing situations.**

The manual testing approach has inherent risks and costs associated with it, as it relies on humans to verify the validity of functionality within an application. Automation is key to improving the speed, accuracy, and flexibility of the software testing process, enabling the company to find and fix more defects earlier. By having an automated functional testing process, owners of software applications can be assured that regression testing can be completed in a more cost effective manner with fewer errors.

Rational Robot is an automated functional regression tool. This helps to execute the tests much faster than we could manually. The key to doing this is to have a way to capture interactions with an application and make those interactions available to be repeated later. This paper demonstrates how to use IBM Rational Robot to record, edit, and execute functional tests.

I. INTRODUCTION

Software testing is an important stage in software life cycle, and it is an assurance of software quality. Software testing exists in each stage of software life cycle, and verifies the expected results are achieved or not, correct the bugs as soon as possible. In software development processing, bugs are always existed no matter what technology is adopted. Testing is applied to find bugs, and used to calculate software bugs density [1] [2].

Furthermore, the software testing is defined in [3] as "the dynamic verification of the behavior of a program on a finite set of test cases, suitably selected from the usually infinite executions domain, against the expected behavior". However, the software testing process can be assisted with software tools to make it automated.

1.1. Types of Software Testing

Although there are many types of software testing, this paper will only include the following types:

Stress Testing: Testing conducted to evaluate a system or component at or beyond the limits of its specified requirements [4].

Load Testing: A test type concerned with measuring the behavior of a component or system with increasing load [4].

Regressions Testing: It is the testing which is to be done to software that was previously working correctly and stops working as intended due to changes [5].

Functional Testing: It is the testing which is conducted on a complete and integrated system to evaluate its compliance with its specified requirements [5].

Unit Testing: It is the verification and validation technique where the programmer gains confidence that individual units of source code are fit for use [5].

Performance Testing: It is the testing which refers to the assessment of the performance of a human examinee [5].

Acceptance Testing: It is the testing which involves running a suite of tests on the completed system [5].

Security Testing: It is the testing which determines that an Information System protects data and maintains functionality as intended [5].

Open Source Testing: It is a functional and unit testing framework for open source software products [5].

Manual testing is time-consuming, error-prone and requires lot of infrastructure and manpower. All these drawbacks can be overcome if the testing process is automated. The testing tools reduce manual testing to a large extent and the testing can be done automatically [6].

II. II. LITERATURE REVIEW

Testing allows companies to discover bugs in an early stage and allows them to be corrected in due time resulting in companies preventing losses in projects. Testing can however be viewed as a boring task. This is the reason why automation testing is moving towards success as its use enable to reduce time, cost and also labor. Test Automation develops and customizes automation test suite that are implemented across projects. The fast pace of modem software development creates tremendous challenges for the test team. Online software updates and frequent release cycles turn out to be a "moving target" for test automation efforts. Test organizations that have not yet advanced automation, beyond simple record/playback, soon discover that recorded test scripts happen to be obsolete within a few project iterations of the software being tested. Many organizations acquiring a test automation tool begin the automation process without any consideration about maintenance, scalability, or effectiveness with however great expectations on immediate payback [7].

"Automated Testing" is simply the automation of the manual testing process currently in place. This requires that a formalized "manual testing process" currently exists in the organization. Automation is the use of strategies, tools and artifacts that enhance or reduce the need of manual or human participation or interaction in unskilled, repetitive or redundant tasks [8]. An automated testing tool is basically a computer-controlled tool that tests software for functionality and performance.

2.1 Why Automating Testing?

Every software development group tests its products, yet delivered software always has defects. Test engineers strive to catch them before the product is released but they always creep in and they often reappear, even with the best manual testing processes. Automated software testing is the best way to increase the effectiveness, efficiency and coverage of software testing process [9].

III. Functional Testing with Rational Robot

3.1 What is Rational Robot?

Rational Robot is a complete set of components for automating the testing of Microsoft Windows client/server and Internet applications. The main component of Robot starts recording tests in as few as two mouse clicks. After recording, Robot plays back the tests in a fraction of the time it would take to repeat the actions manually. Rational Robot is an **automated functional regression** testing tool.

Components of Rational Robot:

Rational Administrator: Create and manage rational projects to store testing information.

Rational Test Manager: Review and analyze test results.

Object Properties, Text, Grid, and Image Comparators: View and analyze the results of verification point playback.

Rational Site Check: Manage Internet and intranet Web sites.

3.2 Case Study

Classics Online is the case study of this research. In this paper we use a simple Visual Basic demo application called Classics Online, included with Rational Robot as Application under Test (AUT) which is shown in figure 1.

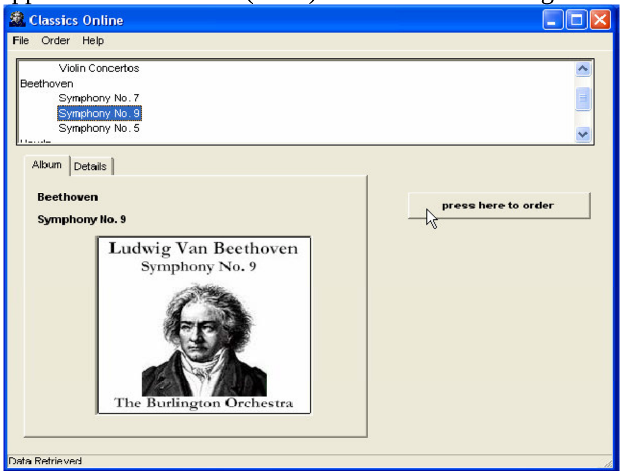


Figure 1. Main Screen of Classics Online

Classics Online is a simple Visual Basic application that simulates a classical music shopping application. The mains screen consists of a tree control where the buyer can select the CD to purchase.

3.3 Rational administrator

This component is used to create and manage projects. The project created in the Rational Administrator is shown in the following figure 2.

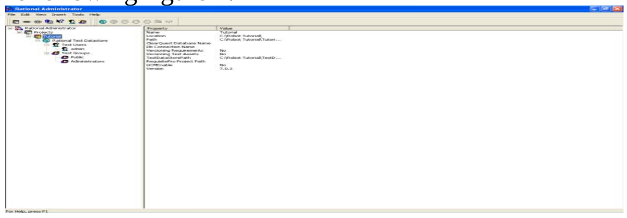


Figure 2. Rational Administrator Window

Rational projects store application testing information such as scripts, verification points, queries, and defects. Projects help to organize the testing information and resources for easy tracking.

3.4 Starting Rational Robot

Next step is to log onto a Rational Administrator Project. The project just created should be selected. The main screen of Rational Robot opens which is as shown in figure 3.

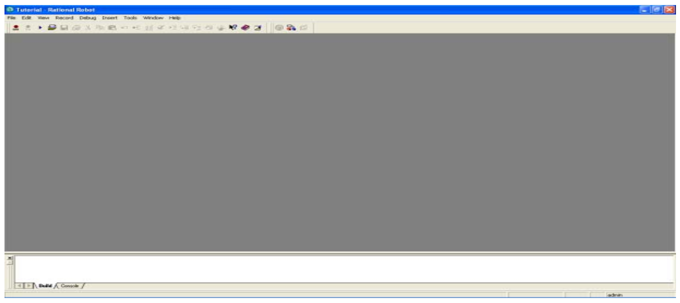
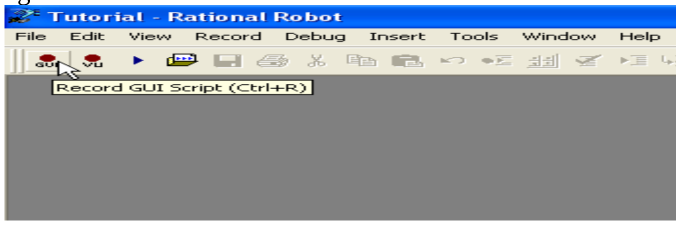


Figure 3. Main screen of Rational Robot

3.5 Recording a Test Script

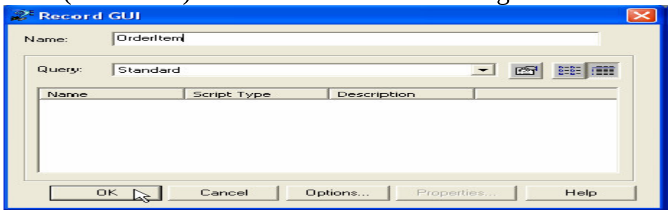
To start recording a GUI test script, click on the red button on the top-left called "Record GUI script" as shown in figure 4.



Tutorial - Rational Robot
File Edit View Record Debug Insert Tools Window Help
Record GUI Script (Ctrl+R)

Figure 4. Record GUI Script button on toolbar

On the screen that pops up, enter a name of the script (OrderItem) and click OK as shown in figure 5.



Record GUI

Name: OrderItem

Query: Standard

Name	Script Type	Description

Figure 5. Record GUI dialog box

Rational Robot records everything that can be done on computer -- all clicks, drags, and typing.

3.6 Application under Test

The first step in recording the script is to start the **Application under Test** (AUT). Click the last button on the toolbar – Display GUI Insert Toolbar as shown in figure 6.

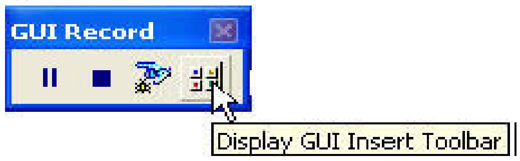


Figure 6. Display GUI Insert Toolbar

This toolbar has many options that we use while recording, but for now select the "Start Application" button as shown in figure 7.

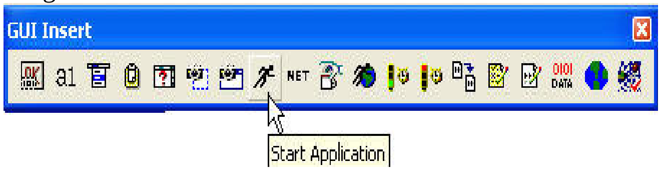
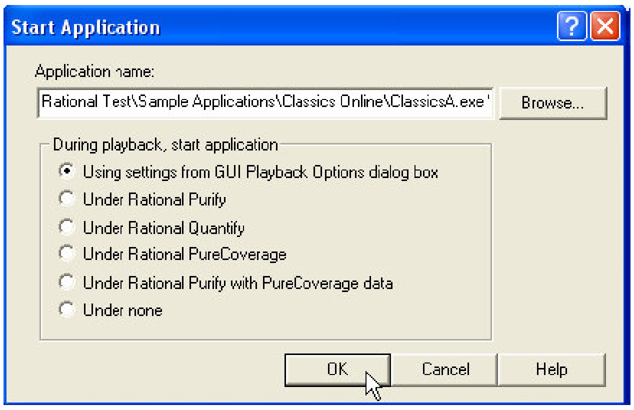


Figure 7. Start button on GUI Insert toolbar

Under Application Name enter "C:\ProgramFiles\Rational\RationalTest\SampleApplications\Classics Online\ClassicsA.exe" or browse for it and hit OK as shown in figure 8.

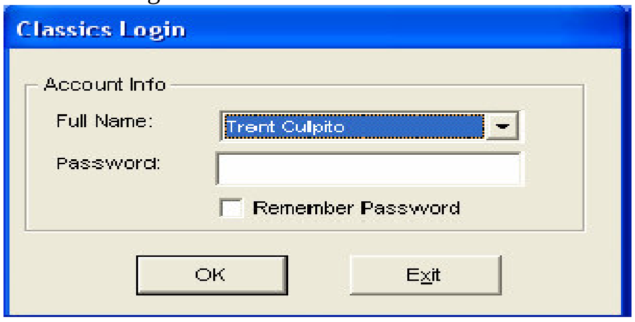

Start Application

Application name: Rational Test\Sample Applications\Classics Online\ClassicsA.exe

During playback, start application
- [✓ Using settings from GUI Playback Options dialog box]
- Under Rational Purify
- Under Rational Quantify
- Under Rational PureCoverage
- Under Rational Purify with PureCoverage data
- Under none

Figure 8. Start Application dialog box

The login window for Classic Online Appears. Click OK as shown in figure 9.


Classics Login

Account Info

Full Name: Trent Culpito

Password:

Remember Password

Figure 9. The login window for Classics Online

Once again, select "Beethoven Symphony No.9" and click press here to order as shown in figure 10.

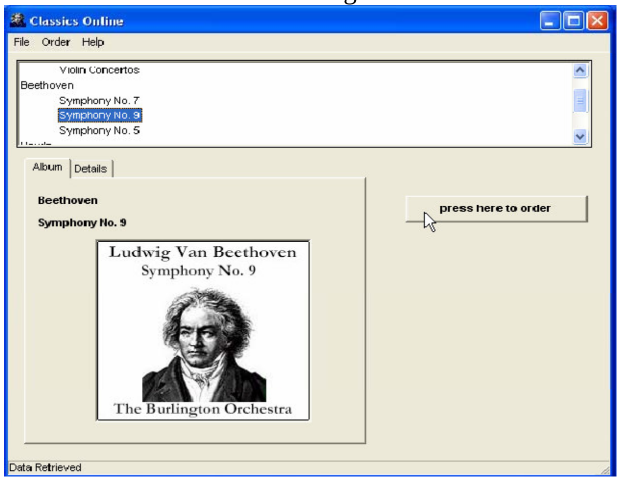
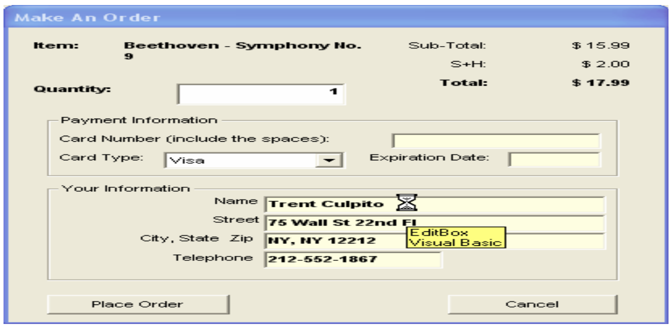


Figure 10. Classics Online window

3.7 Verification Points

So far we've only had Robot record the navigation through the application, but have not actually tested anything. To test something, we set "Verification points" (VPs) on items we want to check against. Now we must designate the object to be tested. We can do this either by dragging the "Object Finder Tool" over the object, or by browsing for it. Drag and drop the tool over the Customer Name text field as shown in figure 11.


Make An Order

Item: Beethoven - Symphony No. 9 — Sub-Total: $ 15.99 — S+H: $ 2.00 — Total: $ 17.99

Quantity: 1

Payment Information

Card Number (include the spaces):

Card Type: Visa — Expiration Date:

Your Information

Name: Trent Culpito

Street: 75 Wall St 22nd Fl

City, State Zip: NY, NY 12212

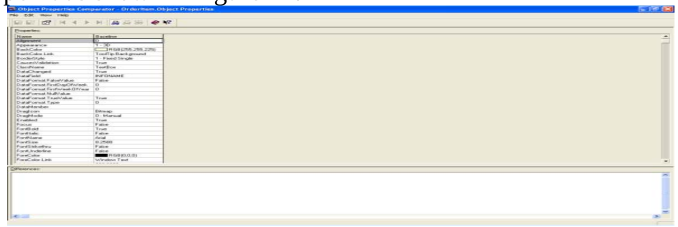
Telephone: 212-552-1867

EditBox
Visual Basic

Figure 11. Make an Order window

Notice that the tool-top provides info about the object as Robot sees it.

Remember we are inserting an object properties verification point. The window that appears next shows us all the properties that we could verify with this verification point as shown in figure 12.


Object Properties Comparator - OrderItem.Object Properties

Properties:

Name	Baseline
Alignment	0
Appearance	1 - 3D
BackColor	RGB(255,255,225)
BackColor Link	ToolTip Background
BorderStyle	1 - Fixed Single
CausesValidation	True
ClassName	TextBox
DataChanged	True
DataField	INFONAME
DataFormat.FalseValue	False
DataFormat.FirstDayOfWeek	0
DataFormat.FirstWeekOfYear	0
DataFormat.NullValue	
DataFormat.TrueValue	True
DataFormat.Type	0
DataMember	
DragIcon	Bitmap
DragMode	0 - Manual
Enabled	True
Focus	False
FontBold	True
FontItalic	False
FontName	Arial
FontSize	8.2500
FontStrikethru	False
FontUnderline	False
ForeColor	RGB(0,0,0)
ForeColor Link	Window Text

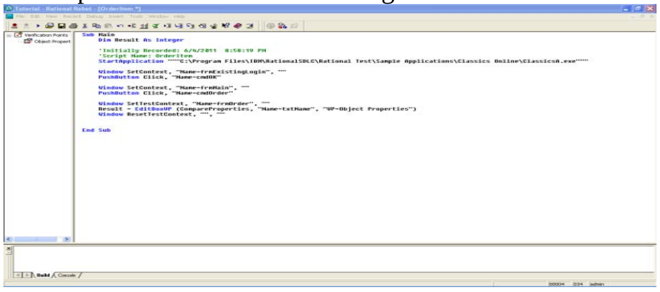
Differences:

Figure 12. Object Properties Verification Point window

Notice how Rational Robot can see many properties of the object.

3.8 The Test Script

Now that we have finished recording, Rational Robot restores itself. Maximize Rational Robot and maximize the script window within Rational Robot. The largest pane is the script itself which is shown in figure 13.


Tutorial - Rational Robot - [OrderItem *]

```
Sub Main
    Dim Result As Integer

    'Initially Recorded: 6/4/2011  8:58:19 PM
    'Script Name: OrderItem
    StartApplication """C:\Program Files\IBM\RationalSDLC\Rational Test\Sample Applications\Classics Online\ClassicsA.exe"""

    Window SetContext, "Name=frmExistingLogin", ""
    PushButton Click, "Name=cmdOK"

    Window SetContext, "Name=frmMain", ""
    PushButton Click, "Name=cmdOrder"

    Window SetTestContext, "Name=frmOrder", ""
    Result = EditBoxVP (CompareProperties, "Name=txtName", "VP=Object Properties")
    Window ResetTestContext, "", ""


End Sub
```

Figure 13. Script Window

The commands we see are in a language called SQABasic. SQABasic is an extension of the Visual Basic programming language. As we can see, it is easy to read and understand what the script will do. Commands like Start Application and Pushbutton Click are pretty self-explanatory.

3.9 Running the Test Scrip

Click on playback script button on the toolbar as shown in figure 14.

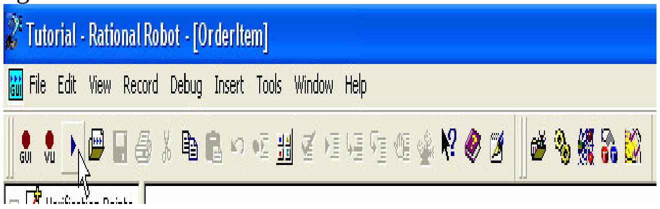


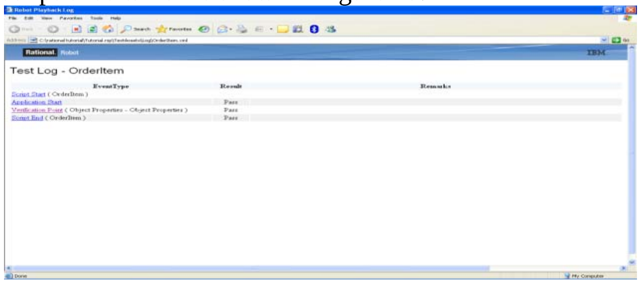
Figure 14. Playback Script button on the toolbar

Rational Robot plays the script much faster than we could manually. Besides the repeatability of an automated testing approach, the computer can perform tests much faster than we ever could manually [10].

IV. DISCUSSION OF RESULTS

4.1 Analyzing results in the Test Log

The Test Log captures all the significant events during playback such as application starts, verification points, and script ends which is shown in figure 15.



Robot Playback Log

Test Log - OrderItem

EventType	Result	Remarks
Script Start (OrderItem)		
Application Start	Pass	
Verification Point (Object Properties - Object Properties)	Pass	
Script End (OrderItem)	Pass	

Figure 15. Test Log

4.2 Analyzing results in the Object Properties Comparators

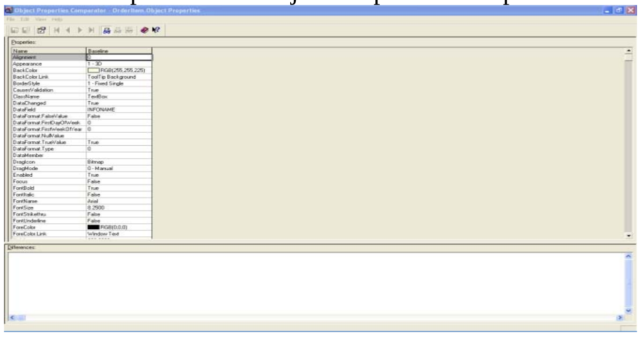
4.2.1 Editing a Verification Point

When we record a verification point in a script, the verification point is stored in the project, along with any associated files.

The verification point name appears in the Asset pane of the Robot Script window. We can view and edit the baseline file of a verification point.

4.2.2 Viewing a Baseline File

In the Asset (left) pane in Robot, right-click the verification point name and click View Baseline, or double-click the name. As the following figure 16. shows, Robot opens the baseline file of an Object Properties verification point in the Object Properties Comparator.

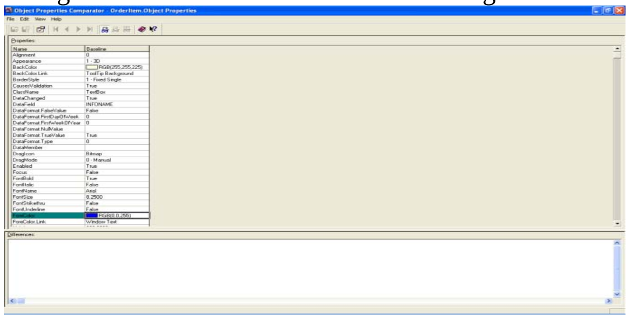


Object Properties Comparator - OrderItem.Object Properties

Name	Baseline
Alignment	0
Appearance	1 - 3D
BackColor	RGB(255,255,225)
BackColor.Link	ToolTip Background
BorderStyle	1 - Fixed Single
CausesValidation	True
ClassName	TextBox
DataChanged	True
DataField	INFONAME
DataFormat.FalseValue	False
DataFormat.FirstDayOfWeek	0
DataFormat.FirstWeekOfYear	0
DataFormat.NullValue	
DataFormat.TrueValue	True
DataFormat.Type	0
DataMember	
DragIcon	Bitmap
DragMode	0 - Manual
Enabled	True
Focus	False
FontBold	True
FontItalic	False
FontName	Arial
FontSize	8.2500
FontStrikethru	False
FontUnderline	False
ForeColor	RGB(0,0,0)
ForeColor.Link	Window Text

Figure 16. Object Properties Comparator

Change the fore color to violet as shown in figure 17.

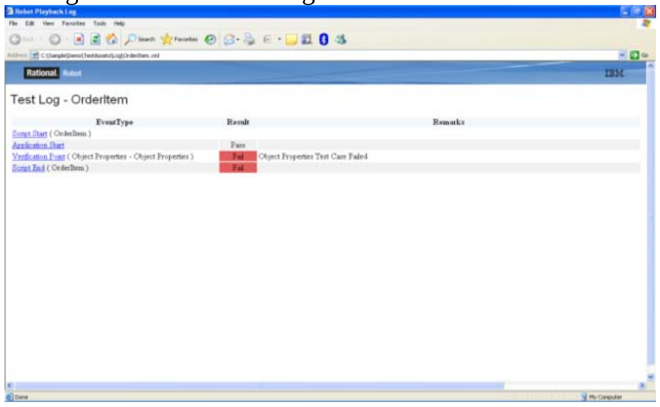


Object Properties Comparator - OrderItem.Object Properties

Name	Baseline
Alignment	0
Appearance	1 - 3D
BackColor	RGB(255,255,225)
BackColor.Link	ToolTip Background
BorderStyle	1 - Fixed Single
CausesValidation	True
ClassName	TextBox
DataChanged	True
DataField	INFONAME
DataFormat.FalseValue	False
DataFormat.FirstDayOfWeek	0
DataFormat.FirstWeekOfYear	0
DataFormat.NullValue	
DataFormat.TrueValue	True
DataFormat.Type	0
DataMember	
DragIcon	Bitmap
DragMode	0 - Manual
Enabled	True
Focus	False
FontBold	True
FontItalic	False
FontName	Arial
FontSize	8.2500
FontStrikethru	False
FontUnderline	False
ForeColor	RGB(0,0,255)
ForeColor.Link	Window Text

Figure17. Object Protperties Comparator OrderItem-Object Properties

Click on playback button. Now the object properties test case failed because initially the fore color was black, but we have changed to violet. The results can be viewed in test log which is shown in figure 18.



Robot Playback Log

Test Log - OrderItem

EventType	Result	Remarks
Script Start (OrderItem)		
Application Start	Pass	
Verification Point (Object Properties - Object Properties)	Fail	Object Properties Test Case Failed
Script End (OrderItem)	Fail	

Figure 18. Viewing Results in the Test Log

When the baseline and actual are same, the test case can be considered as passed otherwise it has to be considered as failed. At present the fore color was violet, that's why the test case has to be considered as failed.

Now again we changed the fore color to black which is shown in figure 19.

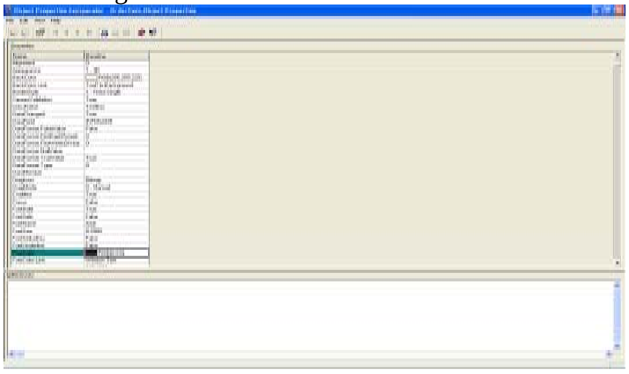
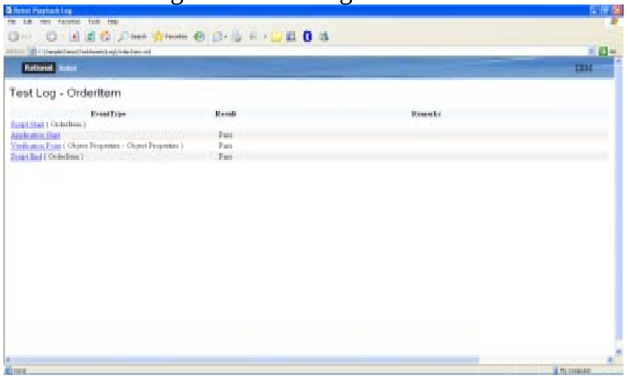


Figure 19. Object Properties Comparator window

In this case the test case is passed because both the baseline and actual data are same. The results can be viewed in test log as shown in figure 20.



Robot Playback Log

Test Log - OrderItem

EventType	Result	Remarks
Script Start (OrderItem)		
Application Start	Pass	
Verification Point (Object Properties - Object Properties)	Pass	
Script End (OrderItem)	Pass	

Figure 20. Test Log- Viewing Results

When the baseline and actual data do not match the verification point fails. As mentioned in the above case the fore color was black, but we changed to violet. Here in this case the object properties test case failed because both the baseline and actual data are not equal. After that we changed the fore color to black. In this case the test case passed (the baseline and actual data are equal).

III. CONCLUSION AND FUTURE WORK

This paper explains some of the basic functionalities of Robot and shows how to record a script with Rational Robot, set up verification point in the script, playback the script, analyze the results of the playback using Test Log, creates functional test and automate the testing process.

In this paper the functionality of Classics Online application by using Object Properties verification point is discussed. Our future work is to test the application by using different kinds of verification points provided by Rational Robot.

REFERENCES

[1] Paul C.Jorgensen. Software Testing: A Craftsman Approach [M].CRC Press, 2002.

[2] Duan Nian, Software performance testing and practical [M]. Tsinghua university Press, 2006.

[3] ISO, ISO/IEC TR 19759: Guide to the Software Engineering Body of Knowledge (SWEBOK), International Organization for Standardization, Geneva, Switzerland, 2005.

[5] G. J. Myers, T. Badgett, T. M. Thomas, and C Sandler, *the Art of Software Testing*, Wiley, USA, 2004.

[6] Dr. K.V.K.K. Prasad, *Software Testing Tools*, Dreamtech Press, 2008.

[7] Doug Hoffman, *A Course on Software Test Automation Design*, Winter 2003.

[8] L Nagowah, P Roopnah, “A simple automated system testing tool”, 2010 *IEEE International Conference on Computer Science and Information Technology*, pp. 301-306.

[9] SmartBear Software, “Why Automated Testing”, 2011.

[10] Dennis Schultz, IBM Corporation, “Functional testing with Rational Robot”, 2007.