An Improved Scheduling Mechanism for Automated Testing Framework based on Spreadsheet DDT

Maykin Warasart[1]

Abstract

Software Testing is considered as one of the most crucial processes in software development. Although many tools have been used to facilitate software testing, these still lack a critical feature, such as test scheduler. Additionally, they lack flexibility in which the test results must be opened by a specific program only. In this paper, the author proposes that "Rational Functional Tester" is an automated test tool that can provide a scheduling mechanism and can self-generate result logs, which can be conveniently opened by Microsoft Excel. The proposed technique combines the test data from users and a configuration file that acts as a scheduler together. Test results will be output in the same input file from users. The users, especially for the user acceptance testing, only need to deal with their provided files.

Keywords: Software Testing, Automated Testing, Test Automation, Test Scheduling, Test Case

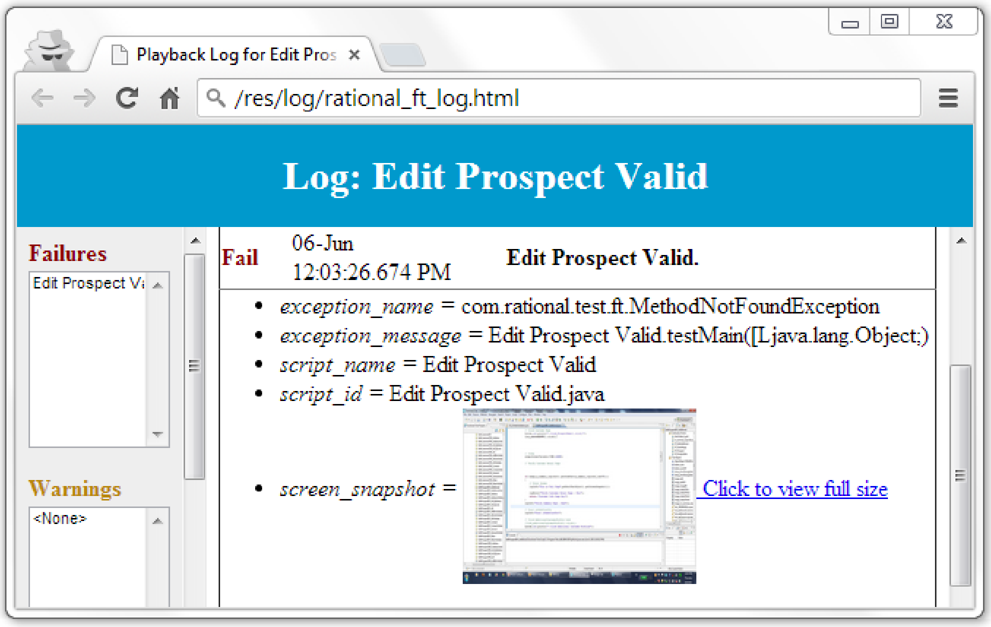
[1] *Corresponding author:* Digital Government Development Agency (Public Organization) (DGA), 17th Floor, Bangkok Thai Tower Building 108 Rangnam Rd. Phayathai, Ratchatewi, Bangkok 10400, Thailand, Email: maykin@dga.or.th, maykin@owasp.org

Introduction

Automated Software Testing (AST) or Test Automation is the utilization of computer software to replace or aid Manual Test Executions. AST offers two obvious benefits to software testing. First, it can improve the accuracy of testing. Secondly, it can reduce testing time and thus overall test duration. In the market today, there are a number of AST tools; however, two of the most prevalent ones are IBM Rational Functional Tester (RFT) (______, n.d.), and HP Quick Test Professional (QTP). These two tools can be used to create test scripts in 2 ways. First, novice testers can use their built-in recording functionality for simple test cases. However, for more advanced users, they may choose to code the test scripts using the programming language supported by the software to create more complex test scenarios (Nawalikit, & Bhattarakosol, 2009)

For both RFT and QTP, the execution of these scripts is usually triggered by a tester who manually launches the programs and tells them to run each script, one at a time. However, running each script one by one may be very time-consuming, especially in a large test project.

To make matter worse, results or logs of the test script created by AST software are typically in a format that cannot be readily used. Additionally, due to the fact that each script normally represents only one test case, the generated result is separated for each test case and contains more information than typically needed. The difficulty is that the tester needs to gather the results of the logs manually in order to produce meaningful reports. As a result, utilizing these logs is quite troublesome.


Playback Log for Edit Pros
/res/log/rational_ft_log.html

Log: Edit Prospect Valid

Failures
Edit Prospect V

Warnings
<None>

Fail 06-Jun 12:03:26.674 PM Edit Prospect Valid.

- *exception_name* = com.rational.test.ft.MethodNotFoundException
- *exception_message* = Edit Prospect Valid.testMain([Ljava.lang.Object;)
- *script_name* = Edit Prospect Valid
- *script_id* = Edit Prospect Valid.java
- *screen_snapshot* = Click to view full size

Figure 1 An Example of the Result Log Generated by RFT

Related Theories

Researchers have conducted studies on the use of AST tools (Bering, & Covey, 1991; Huang, & Chen, 2006; Taipale, Smolander, & Kalviainen, 2006; Zhu, Zhou, Li, & Gao, 2008; Karhu, Repo, Taipale, & Smolander, 2009); they have mostly referred to the major benefits of the tools as providing an efficient aid in testing, and that it also has the ability to record scripts for reuse (Playback). Additionally, the researchers have suggested that the record and playback functionalities of the AST tools are flexible, as they can provide an ease of use as well as reducing costs of testing. However, most researchers have not mentioned how these tools could improve the speed, performance and efficiency of testing.

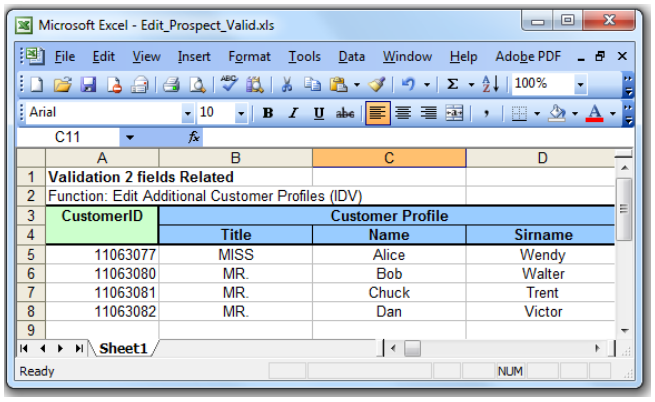
A study conducted by Nawalikit et al. recommended SBTAR (Nawalikit, & Bhattarakosol, 2009) as a tool to help improve the efficiency of generating test results. The study suggests that the format of test results generated by the AST should be changed into a file format that can be recognized by common software in the market, such as Microsoft Word. Nevertheless, the results are probably being separated according to their test scenarios, which still makes it difficult to create a meaningful summary of the overall test results.

Development Details

A. Summary of the System

IBM Rational Functional Tester (RFT) is the software used for automated testing of applications. Automated testers can use RFT to record the test steps or code Java scripts that contain the test steps (Oracle, n.d.). Thereafter, the code can be further modified to test scenarios more accurately. For example, the information (test data) required for test scenarios can be transformed into different sets of data that can be used for the other test scripts. Test scripts can also accommodate additional conditions added to existing test scenarios in the future. In other words, the test scripts are agile and adaptable.

It is worth noting that test data used for RFT test scripts, for example, the information used to fill in the forms, can conveniently be prepared in a spreadsheet using a program such as Microsoft Excel (or Excel) as shown in Figure 2.

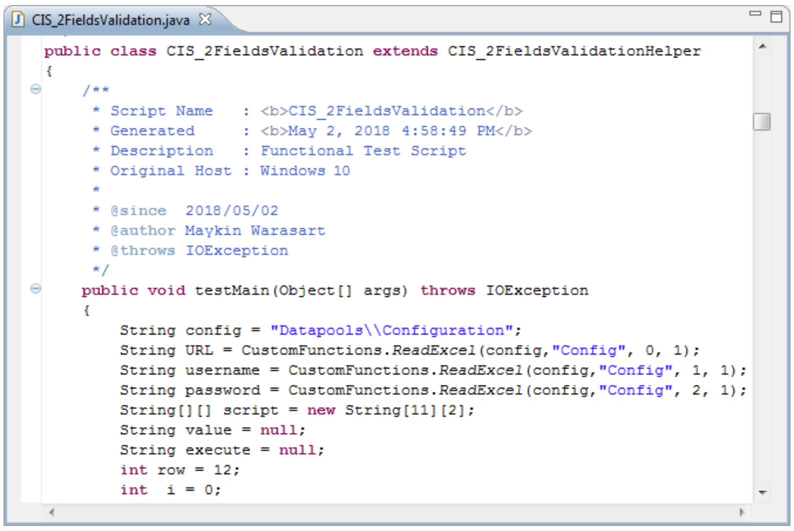

Microsoft Excel - Edit_Prospect_Valid.xls
File Edit View Insert Format Tools Data Window Help Adobe PDF
100%
Arial 10
C11
Sheet1
Ready
NUM

	A	B	C	D
1	**Validation 2 fields Related**			
2	Function: Edit Additional Customer Profiles (IDV)			
3	**CustomerID**	**Customer Profile**		
4		**Title**	**Name**	**Sirname**
5	11063077	MISS	Alice	Wendy
6	11063080	MR.	Bob	Walter
7	11063081	MR.	Chuck	Trent
8	11063082	MR.	Dan	Victor
9				

Figure 2 Example of Test Data in Microsoft Excel Format

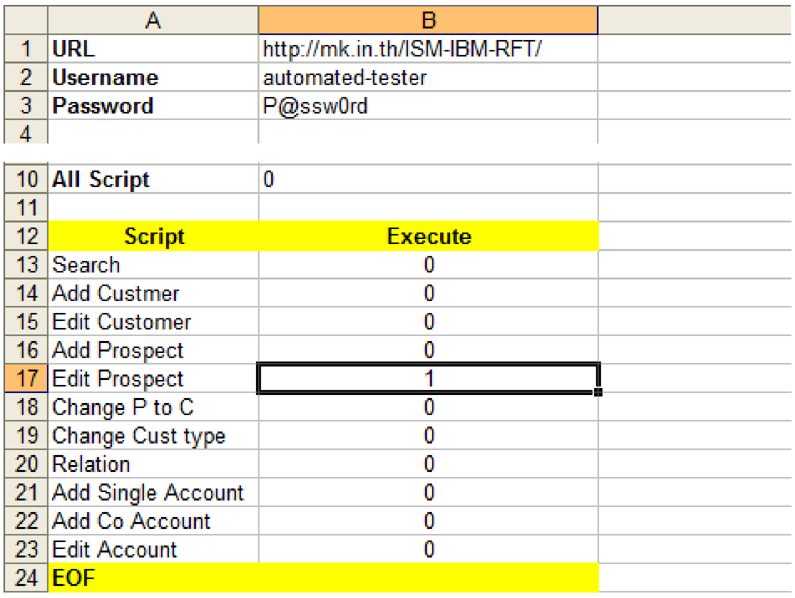
B. Design, Development and the Workflow of the System

The architectural design of the IBM Rational Functional Tester (RFT) consists of three major components including configuration files, Java libraries and the datapool. When the Main test script as in Figure 3 is executed, it will read a configuration file as shown in Figure 4. In the next script execution step, the test script requires Java libraries in order to read the datapool (the test data) because the configuration file is in the Excel format. It is important to note that the libraries for a successful script execution include the useful methods such as read/write and connection from the Java Excel API (Java Excel API, n.d.), the Jacob API (Adler, 2005), and the POI library (Apache, & Java, 2009).



```
public class CIS_2FieldsValidation extends CIS_2FieldsValidationHelper
{
    /**
     * Script Name   : <b>CIS_2FieldsValidation</b>
     * Generated     : <b>May 2, 2018 4:58:49 PM</b>
     * Description   : Functional Test Script
     * Original Host : Windows 10
     *
     * @since  2018/05/02
     * @author Maykin Warasart
     * @throws IOException
     */
    public void testMain(Object[] args) throws IOException
    {
        String config = "Datapools\\Configuration";
        String URL = CustomFunctions.ReadExcel(config,"Config", 0, 1);
        String username = CustomFunctions.ReadExcel(config,"Config", 1, 1);
        String password = CustomFunctions.ReadExcel(config,"Config", 2, 1);
        String[][] script = new String[11][2];
        String value = null;
        String execute = null;
        int row = 12;
        int  i = 0;
```

Figure 3 Example of the Main Script in RFT



	A	B
1	**URL**	http://mk.in.th/ISM-IBM-RFT/
2	**Username**	automated-tester
3	**Password**	P@ssw0rd
4		
10	**All Script**	0
11		
12	**Script**	**Execute**
13	Search	0
14	Add Custmer	0
15	Edit Customer	0
16	Add Prospect	0
17	Edit Prospect	1
18	Change P to C	0
19	Change Cust type	0
20	Relation	0
21	Add Single Account	0
22	Add Co Account	0
23	Edit Account	0
24	**EOF**	

Figure 4 The Contents of the Configuration File

A closer look of the configuration file in Figure 4 reveals that it contains several fields which are directly inputted into the application. These fields can be information, such as a username, password, etc. The configuration file also contains a column used for specifying the names of the test scenarios (Script) while the column next to it receives a value, such as 0 or 1. The numbers 0 and 1 specify whether test scenario should be conducted when the Main test script is run with RFT (Execute); where 0 means excluded (not conducted) and 1 means included (conducted) This file basically acts as a scheduler or a driver for the test script. The "EOF" indicates that the End of File has been reached, and there are no more test scenarios to be run after this value or cell.

The presence of the configuration file allows for both cost reduction and time saving, since it is not necessary to open and run each of the individual scripts separately. This technique also allows the users to select which test scenarios they want to run at a certain instance and which ones to omit.

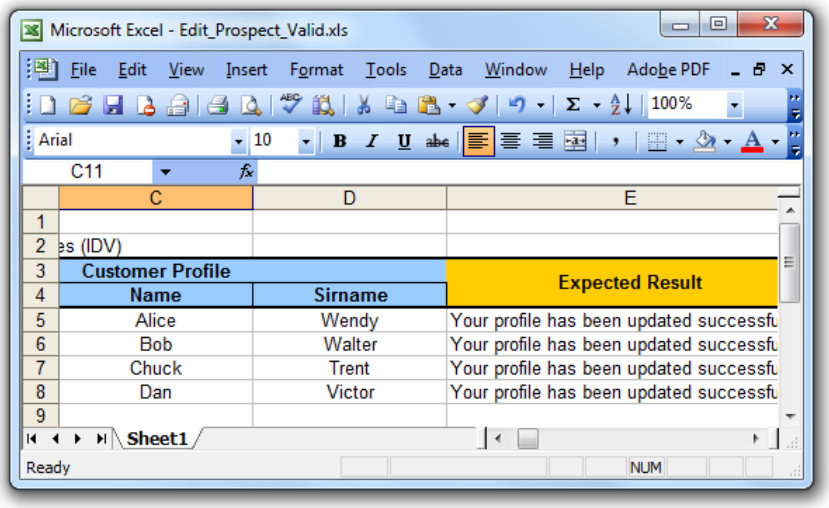
After reading the cells specifying which test scenarios should be executed, the Main script will reference and call the test scripts that will actually be executed for the selected test scenarios. These test scripts also use test data from test data files in which the file names are specified in different worksheets within the Configuration file. This method provides for convenience and compactness, as it allows the tester to select only the Configuration spreadsheet needed for the operation beforehand.

The external libraries mentioned earlier are needed for the process of reading the information, used for the test scenario, from the datapool. These libraries allow for the extraction of the information from each of the cells of the test data. They automate the filling out of the forms and fields of the program/application under test in an orderly and easy manner. The format of the test data requires that each row represents one record. Once the end of the row is reached, it signifies that the information to be filled out for that record is complete, and the marker continues onto the next row of information. However, each test scenario may have different methods of filling out information; it depends upon the objective of each test scenario. Such flexibility allows for multiple forms of test data and records that can be included in one file. The end of the test data file is denoted by an "EOF" which says that there are no more records in this file. Testing of an application, controlled by detailed information from a spreadsheet is known as Data Driven Testing (DDT).

Once the IBM Rational Functional Test (RFT) tool has completed running the test scenarios specified, it will generate a log reporting whether the test scenarios have passed or failed. A passed status implies that the functionality under test performed as expected (Expected Result). A failed status means the opposite. Additionally, one of the features of the RFT allows the ability to extract information from Graphical User Interface (GUI) objects. In this feature, text can be extracted from objects, such as, but not limited to, buttons, fields and dialog boxes.

Text that is extracted from the application under test is usually the one that notifies the status of the test condition. In other words, it produces the actual test result which is then compared to the expected test result.

The result of the test scenario is stored in the same file as the input (the test data) for the test scenario. Specifically, it is stored in the column next to the expected result. This allows for a swift and convenient means of comparison of the actual result of the test scenario against the expected result.


Microsoft Excel - Edit_Prospect_Valid.xls

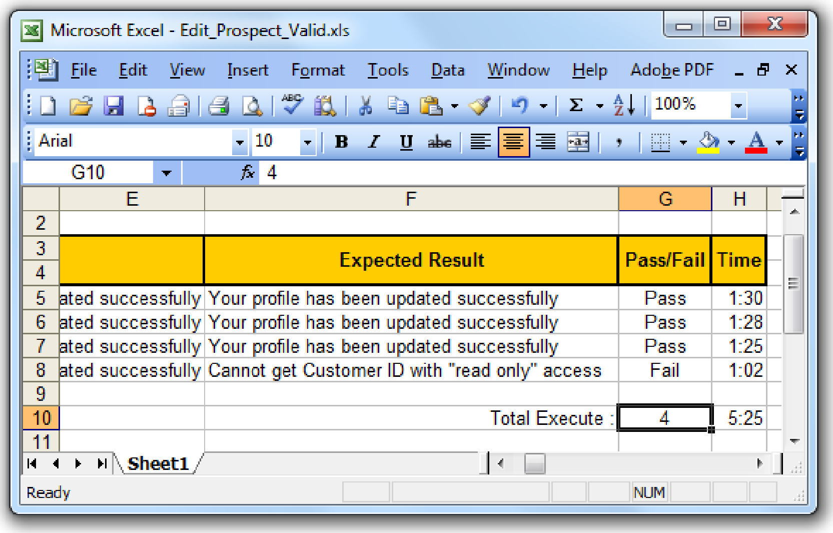
C11

	C	D	E
1			
2	es (IDV)		
3	Customer Profile		Expected Result
4	Name	Sirname	
5	Alice	Wendy	Your profile has been updated successfu
6	Bob	Walter	Your profile has been updated successfu
7	Chuck	Trent	Your profile has been updated successfu
8	Dan	Victor	Your profile has been updated successfu
9			

Sheet1

Ready

Figure 5 Input Test Data File


Microsoft Excel - Edit_Prospect_Valid.xls

G10 4

	E	F	G	H
2				
3		Expected Result	Pass/Fail	Time
4				
5	ated successfully	Your profile has been updated successfully	Pass	1:30
6	ated successfully	Your profile has been updated successfully	Pass	1:28
7	ated successfully	Your profile has been updated successfully	Pass	1:25
8	ated successfully	Cannot get Customer ID with "read only" access	Fail	1:02
9				
10		Total Execute :	4	5:25
11				

Sheet1

Ready

Figure 6 Output Test Data File

Figure 5, as illustrated below, shows the expected result of the input test data file while Figure 6 shows the contents of the output test data file generated from the input test data file. The output test data file contains the actual result of the test scenario in the column next to the expected result, along with a notification of whether the test scenario is a pass or fail. Additionally, the execution time of each test scenario and the total number of executed test scenarios are printed on the output test data file. If we need to execute the test more than one cycle with the same scenario, we are able to modify, to script and to write the result with different column with the timestamp in case we need the keep the execution history.

C. Limitations and Constraints of the System

Due to the fact that this study focuses on the test scenario development by using RFT and Excel, the project is constrained by the limitations of RFT and Excel requirements. As such, the Java Runtime Environment (JRE) and libraries specified in development details, B. are required to operate in RFT. It must also be noted that the recommended solution requires relatively a lot of time in order to create the scripts. However, this is expected in any automated testing since a great amount of time is typically required to create test scripts that are expandable, accurate and cover all required test conditions

Running the System

The test scripts developed by the RFT in this study were used to test one of IBM's internal web applications. This web application was chosen for this study because it contains a large variety of GUI components, allowing for a wide variety of test scenarios.

A. Environment of the System under Test

The system under test includes the following:

- Microsoft Windows 10
- Microsoft Office 2016
- Java Runtime Environment v8
- IBM Rational Functional Tester v8.6

B. Results and Analysis of the Test

According to the results of testing of the chosen test scenarios, it can be concluded that the IBM Rational Functional Tester (RFT) can reduce the workload of the tester conducting regression testing (Wikipedia, 2019). Table 1 demonstrates how effort (man hours) can be reduced after implementing the proposed solution.

Table 1 Comparison of Man Hours of Activities before and after Implementation of Solution

Activity	**Days Spent**	
	Before	**After**
Creation of the scripts	25	27
Preparation of test data	2	2
1st cycle of using scripts	20	10
2nd cycle of using scripts	19	10
Total	66	49

The test results of the software can also be presented in a simple manner because the results and test conditions are organized and kept within the same file. This makes it easy for testers to analyze the results of the test scenarios and to see which test case is passed or failed, along with the reasons of the failed test cases. In addition to providing the users with the ability to generate test results in a simple format, as demonstrated in this study, the tool also provides built-in test results/logs (refer to Figure 1) that offer further detailed information of the test results.

Conclusion

In order to improve the quality of software, it is vital to have effective testing to validate that the software operates in accordance with the requirements. However, the process of testing can be very time consuming and costly. Therefore, the authors propose that automated software testing (AST) tools should be implemented to reduce the test effort, duration and the cost. However, many Automated Software Testing (AST) tools, in the market today, lack the scheduling feature that allows for multiple test scripts execution to cover multiple test scenarios. They are also incapable of producing and displaying the test results in an easy-to-read format.

The conclusion of this study suggests a solution to solve these issues through the use of the IBM Rational Functional Tester (RFT). This tool utilizes a configuration file that acts as a scheduler for multiple runs of various test scripts. These test scripts, in turn, can execute many test scenarios. Each of the test scenarios retrieves the test data from an input test data file. After the scheduler is run, an output test data file is generated, displaying the actual result and comparing it to the expected result, along with the status (pass/fail) of the test scenario. The RFT reduces both the time and cost spent in running the test scripts; it essentially offers a simpler-to-use solution to software testing than its competitors.

References

Adler, D. (2005). *The jacob project: A java-com bridge*. 2009· 01-06)[2009-03-14]. http://danadler. com/jacob.

Apache, P. O. I., & Java, A. P. I. (2009). *To Access Microsoft Format Files*. 2007-03-15]. http://poi. apache. org.

Bering, C. A., & Covey, J. H. (1991, May). Software testing-concepts and approach. *In Proceedings of the IEEE 1991 National Aerospace and Electronics Conference NAECON 1991* (pp. 750-756). IEEE.

Fewster, M. (2001). Common mistakes in test automation. In Proceedings of Fall Test Automation Conference.

Huang, C. H., & Chen, H. Y. (2006). A tool to support automated testing for web application scenario. *In 2006 IEEE International Conference on Systems, Man and Cybernetics* (Vol. 3, pp. 2179-2184). IEEE.

Java Excel API. (n.d.). *A Java API to read, write, and modify Excel spreadsheets*. Accessed from http://jexcelapi.sourceforge.net/

Karhu, K., Repo, T., Taipale, O., & Smolander, K. (2009). Empirical observations on software testing automation. *In 2009 International Conference on Software Testing Verification and Validation* (pp. 201-209). IEEE.

Michael, J. B., Bossuyt, B. J., & Snyder, B. B. (2002). Metrics for measuring the effectiveness of software-testing tools*. In 13th International Symposium on Software Reliability Engineering, 2002.* Proceedings. (pp. 117-128). IEEE.

MSDN Community Center.(n.d.) accessed from http://msdn.microsoft.com/en-us/aa497440

Nawalikit, N., & Bhattarakosol, P. (2009). SBTAR: An Enhancing Method for Automate Test Tools. World Academy of Science, Engineering and Technology*, International Journal of Mathematical, Computational, Physical, Electrical and Computer Engineering, 3*(8), 591-595.

OpenOffice. (n.d.). *Apache OpenOffice 4.1.6.* Accessed from http://www.openoffice.org/

Oracle.(n.d.). *Java Scripting Programmer's Guide*. Accessed from http://download.oracle.com/javase/6/docs/technotes/guides/scripting/programmer_guide/

Softonic.(2019). *IBM Lotus Symphony*. Accessed from http://symphony.lotus.com/

Taipale, O., Smolander, K., & Kalviainen, H. (2006, April). Cost reduction and quality improvement in software testing. *In Software Quality Management-International Conference-* (Vol. 14, p. 63).

Wikipedia. (2019). *Regression Testing*. Accessed from http://en.wikipedia.org/wiki/Regression_testing

Zhu, X., Zhou, B., Li, J., & Gao, Q. (2008). A test automation solution on gui functional test. *In 2008 6th IEEE International Conference on Industrial Informatics* (pp. 1413-1418). IEEE.

_______. (n.d.). *Rational Functional Tester*. Accessed from http://www-01.ibm.com/software/awdtools/tester/functional/

_______.(n.d.). *Java Runtime Environment*. Accessed from http://java.com/en/download/index.jsp