Software Test Automation- Approach on evaluating test automation tools

Tarik Sheth [*], Dr. Santosh Kumar Singh [**]

[*] Research Scholar, AMET- Chennai
[**] HOD- Thakur College of commerce and Science, Kandivali East, Mumbai, India

Abstract- The aim of this paper is to evaluate and establish a mechanism to evaluate testing tools effectively, at the moment, there are many complex systems are built across the platforms and it is very complex problem to establish one common criterion to evaluate the testing tools. There are lot many tools available in the market at the moment, each to0ls have its own features to test software. For this paper, test cases are first manually tested and then as a part of the regression execution, same test cases are coded in the test scripts and being executed over the application under test. Automated testing is developed which saves time and resources, providing a good ROI. Test automation speeds up the testing process and minimizing time to market. An important contribution of this paper is the development of the metric suite that facilitates comparison and selection of a desired testing tool for automated testing.

Index Terms- Automation Testing, Software testing tools, evaluation of tools.

I. INTRODUCTION

Software testing is an area of software development where persistence and continuous efforts are critically essential. Software testing is the process of verifying software quality by using the software with applicable test cases to determine if proposed software requirements are being met properly. Testing is a fundamental aspect of software engineering, but it is a practice that too often is easily forgotten in today's fast-paced Web application development environment. Web applications are being changed both in terms of technology and functionality. Testing applications thoroughly and efficiently is necessary for deployment when wanting to retain existing customers and also fetch in new customers and clients. Testing is really important because software reliability is defined using testing and test effectiveness, approximately fifty percent of the software development budget for software projects is being spent on testing, and this is average in the industry. Software testing is expensive effort as the entire cycle needs to be carry out as frequently as the build is rolled out. therefore, there is a need to reduce testing dependency on human. Software testing is necessary and inevitable because errors are often introduced into software Inadvertently as it is designed, coded and developed. Software has become complex today, which means there are many lines of code, and as a result more cycles of testing needs to be done.

II. METHODOLOGY

The research in this paper includes:

(1) identifying a set of tools to be evaluated
(2) developing a metric suite to be used to evaluate the tools
(3) selecting the target application to be tested
(4) manually testing the applications and recording results
(5) performing a feature analysis of each tool and aggregating an ideal feature set
(6) testing the target application using each selected tool and gathering resulting data
(7) interpreting results
(8) drawing appropriate inferences and making recommendations

3.1 Selected Tools

The testing tools chosen in the comparison for stand-alone based testing were RFT, Ranorex, and Janova. Initially, many other test tools were selected but were rejected because of many reasons. For example, Quicktest and SilkTest were initially chosen but because of the cumbersome setup and initialization they were both rejected.

Also, another tool that was initially chosen but rejected was Panorama. This was rejected also because of the cumbersome installation instructions. The first tool selected is RFT, and the reason it was chosen is because it is the one most widely used. The design of this experiment uses automatic testing tools to gothrough end user steps in the application and compare features between the tools. The second tool that was evaluated was Janova, and the reason it was chosen was because there is no need to download any software or buy equipment. Since it is cloud based, all that is needed is an Internet connection. The tests, or features, are created, and they are then ready to be queued in the tool. By pressing the queue button, a broker, or sort of middleman, is notified that the feature is ready to run. The broker sends the feature to the next available worker in the cloud where it is then executed against your AUT. The test is then completed and filtered back down through the broker and back to the browser with results. The last tool that was chosen is Ranorex, and this was chosen mainly due to the fact that it is widely used with web based applications.

3.2 Evaluation Metrics

There are many reasons why we want to compare testing tools. Metrics are important because it shows us a way to compare different tools to efficiently select appropriate tools to

use based on what testing needs are at the given time. The criteria for comparison in each tool studied are as follows: features, debugging help, automated progress, support for the testing process, usability, and requirements.

The first metric that will be looked at are the features in each tool. Each feature will be highlighted and compared among the different tools. Table 1 is a description of how the features metric will be defined in this project. The features are listed and then evaluated with each tool.

Table 1:

Definition	
This feature will determine whether or not an install is required to use software	Install required
This feature provided by the software, e.g. install required or cloud based	Cloud Based
This feature provided by the software, e.g. can test scripts be recorded with clicks or is programming knowledge required	Knowledge of scripts required
This feature provided by the software, e.g. some tests done strictly on lines of code	Access to code required

3.3 General Testing Approach

The tools will be evaluated for their support for web-based testing. The AUT (application under test) was manually tested for a given amount of time. During the stage of manual testing, each feature in the AUT was reviewed to confirm all features were working. In the testing approach, test scripts were written based on what the AUT was written to do. In the automatic testing stage, once each test script below was written, the tests could easily be played over and over depending on how often tests on AUT needed to be run.

Table 2:

Definition of Test Scripts	
Verify All Services Appear	This script verifies that after logging into application that under the first tab , equipment and services, that all current services appear on page
Verify Login Works	This script verifies that a user can successfully login.
Verify Tabs Work	This script verifies that after logging into application that under the statements tab the current statement appears successfully.
Verify Statements are Viewable	Verify if the statements are viewable.
Verify Logout Works	This script verifies that the current user can successfully log out.
Verify Pay Online Works	This script verifies that a payment online can successfully be made.

3.4 Target Application

In the experiment the application chosen for the comparison is a customer web based application. This particular application was chosen because users need to use the site to access their services to which they currently subscribe to, view current billing statements and history, view toll calls, change account settings and pay their bill online

III. EXPERIMENTAL RESULTS

For each test that was completed, the metrics were outlined and results found for each metric. Results were first taken by reviewing the first tool, RFT.

Table 3:

	Ranorex	**Rational Functional Tester**	**Janova**
Install required	Yes	Yes	No
Cloud Based	No	No	Yes
Knowledge of scripts required	No	No	No
Access to code required	No	Yes	No
List of features	Yes	No	No

The next metric results set is for tool usability. This was found to be important when trying to determine which tool would be best to use. The ease of installation is very important if time is a factor. Table 4 shows the results found in looking at usability metric via a ranking system from 1 through 10. Table 4 shows the results found in looking at the usability metric.

Table 4: Results of Tool Usability

	Ranorex	Rational Functional Tester	Janova
Ease of installation	Very easy to install	Quite a few steps to install. In addition to the software, it is required to	No install

		also install IBM Installation Manager	
User friendly interface	Yes	Yes	Yes
Helpfulness of error messages	Error messages documented	Error messages were documented on website but when using links to get additional help on how to resolve, many links were broken and no help existed. Had to email technical support and wait average of 24 hours for response.	Error messages documented
Tutorial on How-to-use	Very easy to follow	From Help / Getting started, there is a „getting started with the product" link that is very useful	Not easy to find

Table 5: Usability of Testing Tools – Range 1 – 10 (1 Lowest to 10 Highest)

	Ranorex	Rational Functional Tester	Janova
Ease of installation	8	7	NA
User friendly interface	6	7	7
Helpfulness of error messages	7	8	3
Tutorial on How-to-use	6	5	2

IV. Conclusions

In concluding this research paper, I have learned that software testing tools are very Different and it really depends on the context. It takes time and effort and having a software testing goal to know which tool is the best suitable to use given the type of software testing needs and test requirements needs. My personal recommendations are not just restricted with the three tools that have been through the metrics in this research but to evaluate other tools which are in the market based on the parameters defined in the paper. Depending on the needs of a given AUT, the tools recommended would be different given the web based applications or stand alone and thick client appliocations. RFT is definitely the tool to be used when regression testing is important. Janova is the best tool given it is cloud based and can be accessed from any testing machine with internet access. Ranorex is the best tool for web based applications given the different test automation tools built into the software package as it provides packaged test solution.

My vision for the ideal tool is one that is cloud based with no install required and is easy to learn how to use. The ideal testing tool should be easy to navigate, ease of use/ script and also include many tutorials on how to get started in using the tool. It should also have very minimal defects since the bugs encountered during this research were so time consuming to get through it.

V. Future work

The extension of this research paper would be to evaluate the testing tools with respect to technical details such as coding, debugging and support of the technologies. There is a lot to research in this area as there are at least 40 different testing tools available in the market which one needs to explore and come up with the test metrics which can help users and companies to evaluate and select a testing tool which reduces testing cycles and time to market of the software.

References

[1] T. K. Abdel-Hamid, The Economics of Software Quality Assurance: A Simulation-Based Case Study, (Management Information Systems Research Center, University of Minnesota), URL: http://www. jstor. org/pss/249206, 1988.

[2] J. Meek, N. Debnath, I. Lee, H. Lee. Algorithmic Design and Implementation of an Automated Testing tool, URL: http://www. computer. org. uncclc. coast. uncwil.edu/portal/web/csdl/abs/proceedings/itng/2011/4367/00/4367a054toc. htm. pp. 54-59, Eighth International Conference on Information Technology: New Generations,2011.

[3] N. Koochakzadeh, V. Garousi, A Tester-Assisted Methodology for Test Redundancy Detection, Advances in Software Engineering, Hindawi Publishing Corporation, V 2010, Article ID 932686, 2009, URL: http://www.hindawi.com/journals/ase/2010/932686/

[4] Sara Sprenkle, Holly Esquivel, Barbara Hazelwood, Lori Pollock, WebVizor:A Visualization Tool for Applying Automated Oracles and Analyzing Test Results of Web Applications, IEEE Computer Society, August 2008.

[5] Macario Polo, Sergio Tendero, Mario Piattini, Integrating techniques and tools for testing automation, EBSCO host database, ISSN# 09600833, Wiley Publishers, March 2007.

[6] Darryl Taft, IBM Readies Rational Revamp, EBSCO host database, ISSN#

15306283, Academic Search Complete, June 2006.

[7] David Crowther, Peter Clarke, Examining Software Testing Tools, Dr. Dobb"s Journal:Software Tools for the Professional Programmer, ISSN# 1044789X, Academic Search Premier, June 2005, Vol. 30, Issue 6.

[8] IBM.com, IBM Rational Functional Tester, IBM Corporation, December 2008, URL: http://public.dhe.ibm.com/common/ssi/ecm/en/rad14072usen/RAD14072USEN.PDF.

[9] Stuart Feldman, Quality Assurance: Much More than Testing, ACM Digital Library, Queue – Quality Assurance, February 2005, Vol 3, Issue 1.

[10] Keith Stobie, Too Darned Big To Test, ACM Digital Library, Queue – Quality Assurance, February 2005, Vol 3, Issue 1.52

[11] William White, Sifting through the Software Sandbox SCM meets QA, ACM Digital Library, Queue – Quality Assurance, February 2005, Vol 3, Issue 1.

[12] Antti Kervinen, Pablo Virolainen, Heurisitics for Faster Error Detection with Automated Black Box Testing, Science Direct, Electronic Notes in Theoretical Computer Science, Vol 111, January 2005, URL: http://www.sciencedirect.com/science/article/pii/S1571066104052326

[13] Siegfried Goeschl, The JUnit ++ Testing Tool, Dr. Dobb"s Journal: Software Tools for the Professional Programmer, February 2001, Academic Search Premier, February 2001, Vol. 26, Issue 2.

[14] Sylvia Ilieva, Valentin Pavlov, IlinaManova, A Composable Framework for Test Automation of Service-based applications.

[15] Yuetang Deng, Phyllis Frankl, Jiong Wang, Testing Web Based Applications, ACM, September 2004 SIGSOFT Software Engineering Notes, Volume 29 Issue 5.

[16] Cem Kaner, Jack Falk, Hung Nguyen, Common Software Errors, Testing Computer Software, Second Edition, 1993.

[17] Hamzeh Al Shaar, Ramzi Haraty, Modeling and Automated Blackbox Regression Testing of Web Applications, 2008, Journal of Theoretical & Applied Technology, 4 (12), 1182 – 1198. EBSCO host database, ISSN# 19928645.

[18] Edward Hieatt, Robert Mee, Going Faster: Testing the Web Application, IEEE Computer Society, August 2002, Vol 19, Issue 2, Pg (60 – 65), URL: http://www.csse. monash. edu. au/courseware/cse4431/testingWebApps-IEEE-2002. pdf

AUTHORS

First Author – Tarik Sheth, Research Scholar, AMET- Chennai tariksheth@gmail.com

Second Author – Dr. Santosh Kumar Singh, HOD- Thakur College of commerce and Science, Kandivali east, Mumbai, India, sksingh14@gmail.com