Indian Journal of Science and Technology, *Vol 10(13), DOI: 10.17485/ijst/2017/v10i13/109048, April 2017*
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Comparative Study of Open Source Automated Web Testing Tools: Selenium and Sahi

Arjun Satheesh and Monisha Singh

Department of Computer Science, Christ University, Bangalore – 560029, Karnataka, India;
arjun.satheesh@mca.christuniversity.in, monisha.singh@christuniversity.in

Abstract

Objectives: Testing has become an integral part of the Software Development Life Cycle (SDLC). Through open source automated testing tools, the cost of the entire testing process, as well as the time it takes to perform the testing process, has significantly reduced. The primary objective of this research paper is to analyse the web testing tools Selenium and Sahi as to compare their features and performance so that a particular user can select a tool that is suitable in terms of usability and features required for a specific task. **Methods:** The selection of a tool should be according to the user's skill level. The usability of these tools should be taken into consideration. Using a tool which has a complicated and confusing user interface would be frustrating to the user even if it has all the features and the required performance. The tools are compared with respect to - the browsers and the operating systems that they support, the programming languages that can be used to generate the test scripts, the ease in setting up and configuring the tools, the reporting facilities that are available, their data-driven capabilities, the ability to perform parallel batch as well as distributed playback and their ability to handle AJAX. The IDE versions of the tools are used to perform the comparison on two different types of websites. **Findings:** The comparative study shows that Selenium is best suited for people with some experience and coding skill as it involves integrating various frameworks in order to implement high-level functionalities. Sahi is easy to set up and use but it does not have certain advanced features that are seen in Selenium. A major advantage of Sahi is its data-driven capabilities that it supports out of the box. No complex frameworks need to be set up for this purpose. **Improvements:** IDE of both the tools were used in the comparative study and not much manual scripting was done. Selenium requires another IDE such as Eclipse in order to write test scripts while Sahi allows one to write scripts in the IDE itself. Since this paper focuses mainly on the performance aspect of the tools, the IDE's were chosen to provide a fair method of comparison.

Keywords: Automation, Open Source, Selenium, Sahi, Testing Tools, Web Application Testing

1. Introduction

Creating high-quality software is the main aim of software development. Testing involves the evaluation of the quality and efficiency of a product by finding errors. Testing can be done manually as well as automatically, that is, any type of testing can be done manually or automated using a tool.

Manual testing is a time-consuming process and it requires the tester to possess certain qualities such as patience and observation skills. For this purpose, many applications can be used which may serve different purposes. Most of the time a tool is used without figuring out its advantages and disadvantages. For a person who is new to testing, this is the biggest mistake that can be made. They should figure out a tool that is right for their specific purpose. The user takes into consideration the ease of using a tool, for instance, a user-friendly tool would be the best fit for a person who is new to the field of testing. The features that are provided by the tool are also another important factor that should be taken into consideration. Hence selecting a tool that contains the features that satisfy your testing needs is very important. Compatibility is an issue that most people do not take seriously. This is

**Author for correspondence*

a major issue when it comes to web testing. Certain tools may not be compatible with certain browsers and this would be a grave problem when testing a web application. The same is the case with operating systems; all tools may not work with all operating systems.

All the factors show us that the selection of a proper testing tool is a vital part of the testing process. Selecting the proper tool would not only save a lot of time but would keep the costs of the entire software development process in check. The paper would be comparing the performance of the IDE of both the tools and would also be comparing the features of the IDE's of the respective tools as well of their other components.

This paper explains the various features and limitation of the web testing tools Selenium and Sahi.

The various sections of the paper are explained. The details of each tool are explained in Section II along with its various components. The related work is explained in Section III. Research methodologies that are implemented to perform the comparative study are mentioned in Section IV. Analysis of the comparative study is showcased in Section V and the conclusion of the paper is explained in Section VI.

2. Detailed Information of Tools

2.1 Selenium

It is an automated open source web testing suite that works across different platforms and browsers. It is not a single tool but a combination of many tools that can be used to satisfy an organisation's various testing requirements. The various components of Selenium are the Integrated Development Environment (IDE), WebDriver and Grid. Using the IDE tests can be done without the need for learning any scripting languages. It is a Firefox plug in that can be installed easily. The IDE facilitates recording and playback of tests and hence it is used to create simple test cases and not advanced ones. Debugging of test cases is also possible. Usually, it is used by people who are new to Selenium. Selenium has a language of its own, known as Selenese which can be used to write tests in a wide variety of popular programming languages. Test cases that are recorded using the Selenium IDE are in Selenese and these recorded scripts can be manually edited by the tester if needed. The Selenium WebDriver executes commands in Selenese or from a client application interface and sends it to a browser. Each browser has drivers of its own which are used to send the commands to the browser to get the results. Previously Selenium needed a special server to manage browsers but nowadays it directly starts an instance of the browser and manages it directly. In all possible situations WebDriver doesn't make use of browser-based JavaScript commands to control the browser, rather it uses the native operating system-level functionalities to do so.

Using Selenium Grid, it is possible to execute tests in parallel on multiple machines. It is a server using which it is possible for tests to utilise web browser instances that are being carried out on remote machines. Various browser versions and configurations can be handled centrally using Selenium Grid.

2.2 Sahi

Is an automated open source web testing tool and it comes in versions, one open source version and a paid version. The paid variant has additional features such as test distribution and reports customization. This paper would be dealing with the open source version. The open source version consists of features including record and playback on almost all browsers, HTML playback reports, JUnit style playback reports and an ability to playback tests in parallel. The script that is generated using Sahi is known as Sahi script which is an extension of JavaScript. Sahi is developed using Java and JavaScript. Since it is developed using Java it is compatible almost everywhere. Sahi executes itself as a proxy server and the proxy setting of the browser is configured such that it points towards Sahi's proxy. JavaScript event handlers are added to the web pages by Sahi which in turn allows it to perform record and play back in a browser. This usage of proxy is what makes Sahi independent of the browser used.

In[1] the two testing tools Ranorex and TestComplete are compared based on criteria's such as the difficulty in - generating the test scripts, playback of the test scripts, maintenance of the test scripts, report generation, efficiency in reusing test scripts and the cost incurred. The features and concepts of these tools were compared in order to determine their pros and cons. The paper concludes by saying that an ideal tool should be one that is easy to use and learn and which has many tutorials available on the internet. Keeping these objectives in mind, for a situation with a definite objective TestComplete is said to the better tool and in all other situations, Ranorex is preferable.

In[2] the three automated testing tools Ranorex, Rational Functional Tester (RFT) and Janova are compared to evaluate their usability and effectiveness. To do so the author has selected a web application which was first manually tested and then tested using the three automated testing tools. The focus was on the test script development and how the test scripts are affected when the application is updated. They concludes by saying that tool selected depends on the software testing goal. RFT is said to be the best when regression testing is important, Janova has a cloud-based interface and hence can be used from anywhere and finally Ranorex is the preferred tool for testing web applications. Hence it is seen that there is no one perfect tool but the tool that one should select depends on the task that is to be performed. According to the author, an ideal tool should be one that is cloud based which is easy to set up and use with many tutorials available online and which has a minimal number of bugs.

In[3] theautomated testing tools – Selenium, Quick Test Professional (QTP) and TestComplete are compared in terms of their usability and effectiveness. The paper says that the tool that should be selected depends on the type of application that needs to be tested. Selenium is an open source and free tool unlike the other two and hence it should be selected if you don't want to spend anything on the testing too. The author concludes by saying that if a person's testing requirements are getting fulfilled by TestComplete then it is pointless to use QTP for the same purpose since the latter is more expensive. Application Under Test (AUT's) is better to use QTP as it is a more accomplished tool. The QTP is the best tool among the three.

In[4] main features of different scripting techniques used in the execution phase of automated software testing are compared. The paper compares five scripting techniques are Linear, Structured, Shared, Data-Driven and Knowledge Driven. If a script is created with a short span of time, then the maintenance cost will be high and if more time is spent to create scripts then the maintenance cost will be low. The ideal situation should be the combination of both the above scenarios. The paper concludes by saying that the Data-Driven scripting technique is the recommended technique for automating the execution phase as it is the most cost effective technique.

In[5] automated testing is performed on a web application using the software testing tool Selenium. The IDE of Selenium is used in this case. The main aim of the paper is to find out the ease with which automated software testing can be done. Selenium consists of many tools such as IDE, WebDriver and Grid. The Selenium IDE records the user's actions when data is being entered and test scripts are generated from these actions. The Selenium IDE works only on the Firefox browser and this is its most major drawback. The paper concludes by saying that the generated test cases are reusable and are best suited in the regression testing environment. If one wishes to test a web application in any other browser, then the Selenium WebDriver would have to be used which supports many programming languages and supports almost all browsers that are available in the market.

In[6] certain methodologies and software architectures which when implemented decreases the fragility of software is explained. The paper supports a model-based approach to testing and a philosophy of continuous testing throughout all phases of the software lifecycle from requirement collection, the design of software, coding, testing, software release and maintenance. Various experiments are conducted and to fulfil all the objectives. The paper concludes by saying that from the experiments that were conducted methodology and tools that are used play a great deal in automated software testing. It is said that the early involvement of testers in the development process further improves the quality of the software but it makes the documentation process more time consuming as it would require earlier and more complete documentation.

In[7] Sahi and Selenium are compared with respect to their features. The paper clearly highlights the limitations of Selenium and gives a brief explanation of how Sahi works. Five limitations of Selenium are described, which are its inability to work with Ajax, the use of weak and complicated xpath, the absence of an inbuilt reporting functionality, inability to run the same script in different browsers and the option to perform record and playback only in Firefox. The paper concludes by saying that Sahi saves time for testers and developers but the selection of the tool depends on requirements and the situation that is faced by the tester.

In[8] performances of the tools, Selenium and QTP are compared. The effort required to automate the time taken to execute the test cases are compared. The results show that Selenium is the faster tool but the paper concludes by saying that Selenium is designed to test web applications only and that it is really complex as it involves the integration of many components. Even though Selenium is free, automation requires the skill set of a developer. The selec-

tion of the tools would depend on the situation and the skill set of the tester.

In[9] the differences between popular web testing tools available in the market are explained. A set of evaluation parameters are specified and the tools are compared according to these parameters. This helps to select a tool according to the needed requirements. Though the paper does not have a definite conclusion, it shows the various tools against the evaluation parameters which can enable a tester to select a tool based on his requirements.

In[10] features and the advantages of Sahi are explained. The architecture of the tools are explained and how it is set up is also explained thoroughly. The recorder is said to be very easy to use, it is browser and platform independent, no xpaths, no waits, parallel playback and inbuilt reporting. It also supports data driven testing and has an ability to connect to any database, excel or CSV file. Sahi gives testers an ability to write their test cases in excel and run them. The simplicity in doing this is also highlighted.

3. Research Methodology

Sahi and Selenium would be compared according to their features, each one's limitations and their performance according to the execution speed of their test scripts. The Table 1 lists all the evaluation parameters that are used to differentiate the two tools.

Table 1. Description of the parameters

Parameter	Meaning
Browser Support	The browsers that the tools are able to work with
OS Support	The OS's that the tools are compatible with
Support	The support that is provided by the developers of the tool
Coding Experience	Experience that is required by the user in terms of coding
Scripting Languages	Programming Languages that are used to create and edit the test scripts
Ease in Setup and Configuration	The difficulty in setting up and using the tools
IDE	The ability of the tool to record and playback the user's actions
Parallel Batch Playback	Whether the tools support parallel batch playback
Distributed Playback	Whether the tools support distributed playback
Data Driven Capability	The ability to connect to any database, excel or CSV file
AJAX Loading	The ability of the tools to handle AJAX loading
Reporting Functionality	Ability to provide reports for the executed test scripts
Locating Elements	The method used to locate the various elements in a web application

A tester would be able to understand the tools better by looking at this comparative result. To compare the performance of the IDE's of these tools, two web applications are made use of, and they are http://www.calculator.net and http://www.team-bhp.com/. The former is a calculator and in this certain scenario, the time taken to calculate the percentage of two numbers is found out. The latter is a forum and the time taken to log in and log off is calculated. The time taken by the tools for the completion of above tasks is compared to find the better performing tool. Writing and executing scripts in any language cannot be done using the recorder in the case of selenium. It requires the use of Selenium Web Driver and another IDE like Eclipse. Sahi supports writing of scripts in its IDE itself. Obviously, the execution time in the case of Selenium would be on the higher side and due to this, the comparison is not done when scripts are written manually by the tester.

4. Analysis

4.1 Comparison of Features

The tools Sahi and Selenium are compared with respect to a number of parameters that help users make a decision on the tool that is suited for their current objective is shown in Table 2.

4.2 Browser Support

Selenium supports only Mozilla Firefox, Google Chrome, Microsoft Internet Explorer, Opera and Apple's Safari. This is in the case of Selenium WebDriver and Grid.

The record and playback IDE of Selenium support only Mozilla Firefox. Sahi supports all browsers that supports JavaScript and the record and playback IDE supports almost all browsers.

Table 2. Comparison of the tools

Parameters	Tools	
	Selenium	Sahi
Browser Support	Firefox, Chrome, IE, Opera and Safari	Any browser that supports JavaScript
OS Support	Windows, Macintosh and Linux	Any OS that supports Java
Ease of Support	User andCommunity support available	Community Support is available only via forums.
Coding Experience	One should possess good technical capabilities of integrating different pieces of the framework.	Much less is required when compared to Selenium
Ease in Set Up and Configuration	Complicated	Easy but UI is confusing
Scripting Languages	C#, Java, JavaScript, Objective-C, Perl, PHP, Python and Ruby	Sahi Script (Similar to JavaScript syntax)
IDE	Supports only Firefox	Supports Firefox, IE and Chrome
Parallel Batch Playback	Supported using Selenium Grid	Supported
Distributed Playback	Supported	Not Supported in the Open Source Version
AJAX Loading	Complex and requires many lines of code which includes wait statements	Uses fewer lines of code with no wait statement
Reporting Functionality	Doesn't have inbuilt reporting functionality	Has the ability to generate its own reports.
Locating Elements	Xpath is used to locate elements	Uses its own wrappers around DOM objects to locate elements

4.3 OS Support

Sahi supports any operating system that supports Java while Selenium supports only Windows, Macintosh and Linux.

4.4 Ease of Support

The customer support for any tool is important especially when doing big projects. If an error with the tool cannot be solved quickly then it will affect the development time of the project. Even though Selenium is an open source tool, it consists of a huge user and community support where problems can be mentioned and fixes come really fast. Sahi only provides support via forums and hence fixes will be much more delayed.

4.5 Coding Experience

Sahi uses SahiScript which is very much similar to JavaScript and that is all that is needed to write scripts. The difference between JavaScript and SahiScript is that the latter has a script that is parsed by the proxy and this script is a fully valid JavaScript and the Rhino JavaScript engine executes it. Another difference is these of the$ symbol in front of all variables. Selenium uses a language called Selenese which is a set of commands to perform automation in web applications. Other languages such as Java, C#, Python, PHP, Pearl, Ruby, JavaScript and Objective – C are also supported by Selenium WebDriver and Selenium Grid. It is clear that Selenium requires the user to have a much better coding experience and knowledge.

4.6 Ease in Setting Up and Configuration

Selenium is very difficult to set up and configure as it requires the user to implement many third party frameworks. This may not be possible for a person who is not a developer. Hence proper implementation of Selenium requires developer-level knowledge. Sahi is comparatively simpler to implement but the UI of the record and playback IDE is not as intuitive as in the case of Selenium.

4.7 Record and Playback IDE

The IDE of Sahi supports almost all browsers while the IDE of Selenium supports only Mozilla Firefox. This is a major disadvantage of Selenium.

4.8 Parallel Batch Playback

Selenium supports parallel playback via the use of Selenium Grid but setting this up is not an easy task and requires some coding knowledge. Sahi, on the other hand, does not require that much knowledge but the parallel

execution, in this case, is not as advanced as Selenium Grid.

4.9 Distributed Playback

Selenium Grid is required to perform distributed playback in the case of Selenium but Sahi does not support this feature.

4.10 AJAX Loading

Selenium does not support AJAX loading. Suppose a calendar application is taken as an example. Selenium requires about 20 lines of code with explicit waits to handle this. And when the script is executed, the selected date information is not displayed. Sahi uses just 4 lines of code to perform the same function and the selected date information is displayed.

4.11 Reporting Functionality

Selenium does not have any reporting functionality and requires third party tools to generate the same. Sahi has the capability of generating reports on its own.

4.12 Locating Elements

Selenium uses xpath to locate elements. The use of xpath increases execution time by a great deal. The use of xpath leads to slow, brittle and unmaintainable tests. The time taken by xpath differs depending on the browser that is used. Sahi uses its own wrappers around DOM to locate the elements. Time consumption is much less when compared to xpath.

4.13 Script Writing

In Sahi, the script can be run in different browsers without any modification to the script. It's JavaScript and proxy support is guaranteed to work with any browser and its API's are normalised to work across browsers. Another reason is that it does not use xpath to locate elements. Selenium can work efficiently only with Mozilla Firefox and Google Chrome. In other browsers, issues come up from time to time.

4.14 Data Driven Framework

The use of data-drivenframework reduces the efforts needed to create and maintain test cases. It proposes better ways to organise test cases. The test data is stored in a separate file instead to taking this from the test case. This enables the storage of test data and the expected output in a single file. For instance, the login details of a web application being tested can be stored in an excel file and not together with the test case. In such cases, the maintenance cost is lower than the recording of the test cases from the beginning. Hence test will not have to be re-recorded but needs to be maintained. Sahi supports Excel framework out of the box and is very easy to use. Selenium, on the other hand, requires integration of third party API's such as Apache POI to work with excel, CSV and other formats. The use of such frameworks also enables non-programmers to write test cases which can, in turn, speed up the entire development process of the web application.

5. Recording and Playback

The web applications 'Simple Calculator' and a forum 'Team-BHP' are tested by first recording the user's movements using the recorder. Then the recorded script is played back using the playback feature.

5.1 Selenium

5.1.1 Simple Calculator

From Figure 1 we see that the IDE has generated the commands required to navigate to the link and to enter the values in Selenese. Selenium IDE does not show the execution time of a test case and hence the script is modified to accommodate the same.

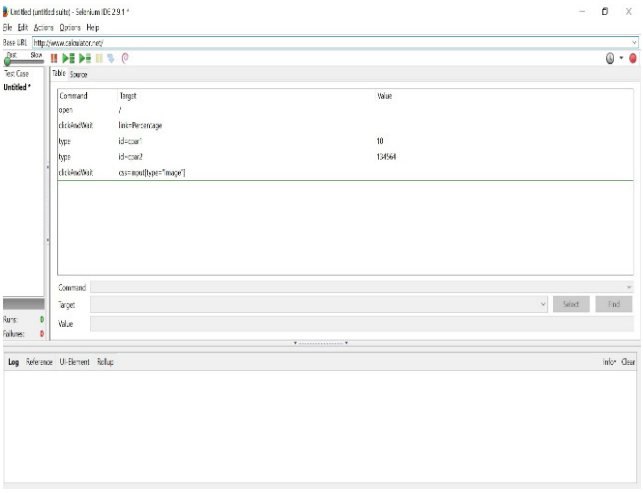

Figure 1. The code generated for the calculator application by Selenium.

A few commands are added in Selenese for this purpose which we see in Figure 2.

Command	Target	Value
storeEval	new Date().getTime()	TeststartTime
open	/	
clickAndWait	link=Percentage	
type	id=cpar1	10
type	id=cpar2	134564
clickAndWait	css=input[type="image"]	
storeEval	new Date().getTime()	TestendTime
storeEval	(${TestendTime} - ${TeststartTime}) / 1000	scriptExecutionTime

Figure 2. Code for computing execution time is added.

The end result is seen in Figure 3.

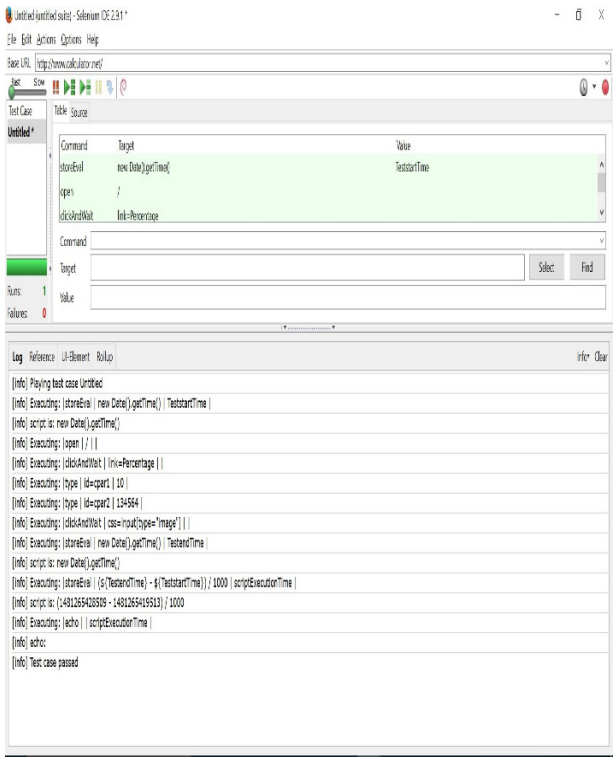

Figure 3. The log file after test case execution.

When we calculate the execution time from the log file in Figure 3, it is 1.004 seconds.

5.1.2 Team-BHP

The code generated for the forum Team-BHP is seen in Figure 4.

Command	Target	Value
storeEval	new Date().getTime()	TeststartTime
open	/	
clickAndWait	css=a[title="Forum"] > span	
type	id=navbar_username	arjunsatheesh
type	id=navbar_password	
clickAndWait	css=input.button	
waitForElementPres...	link=Log Out	
click	link=Log Out	
assertConfirmation	Are you sure you want to log out?	
storeEval	new Date().getTime()	TestendTime
storeEval	(${TestendTime}-${TeststartTime})/1000	scriptExecutionTime

Figure 4. The code generated by Selenium for the forum web application.

When the execution time is calculated from the log file in Figure 5 we see that it is 9.934 seconds.

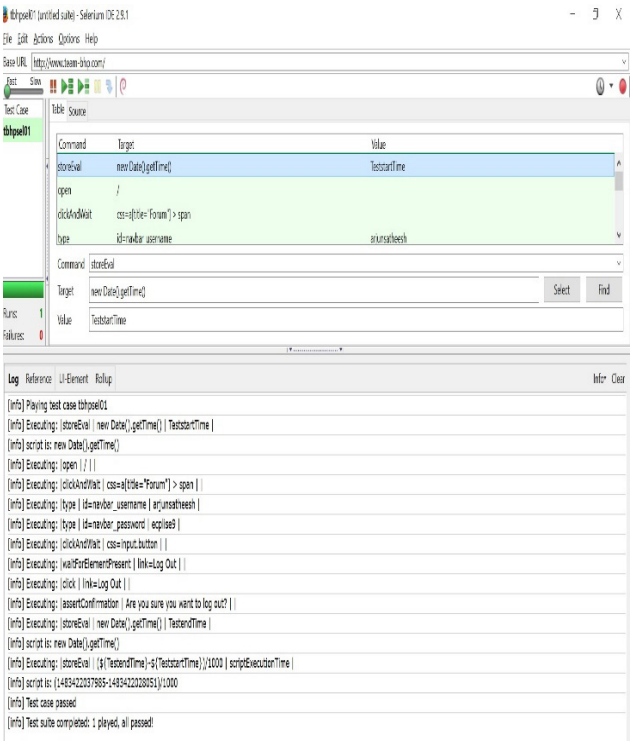

Figure 5. The log file for the forum web application.

5.2 Sahi

After Recording the User's Actions:

5.2.1 Simple Calculator

1 _click (_link (“Percentage”));
2 _setValue (_textbox (“cpar1”), “10”);
3 _setValue (_textbox (“cpar2”), “134564”);
4 _click (_image Submit Button (“Calculate”));

The script is generated in Sahi Script.

The report that is generated after playing the test case is seen in Figure 6.

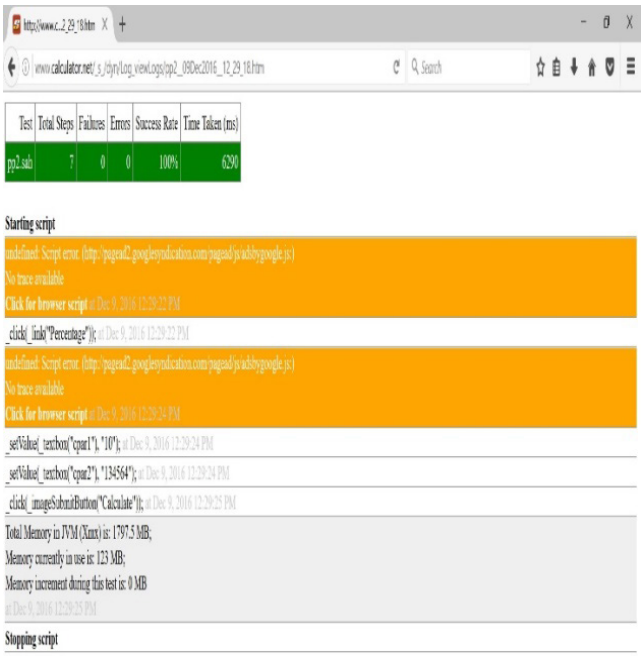

Figure 6. Log file generated by Sahi for the calculator application.

We see from Figure 6 that the execution time is 6.290 seconds. This is mainly due to some error that came up during playback. IDE’s generate an error in certain instances when elements such as advertisements crop up on the page.

5.2.2 Team-BHP

_click (_span (“Forum”));
_setValue (_textbox (“vb_login_username”), “arjunsatheesh”);
_setValue (_password (“vb_login_password”), “*****”);
_click (_submit (“Log in”));
_click (_link (“Log Out”));
_expectConfirm (“Are you sure you want to log out?”, true);

The report that is generated for the web application ‘Team-BHP’ is seen in Figure 7.

Figure 7. The log file that is generated for the forum web application.

From Figure 7 we see that the execution time is 13.043 seconds.

Selenium Script Execution Time:

Simple Calculator: 1.004 seconds.
Team-BHP Forum: 9.934 seconds.

Sahi Script Execution Time:

Simple Calculator:6.290 seconds.
Team-BHP Forum: 13.043 seconds

In the case of both the web applications, we see that Selenium executes the test case faster but to calculate the execution time, the Selenese script had to be modified while Sahi produced a full report about the executed test case. Hence for a tester with not much code knowledge the use of selenium won’t be that easy. Sahi, on the other hand, is very easy to use and doesn’t require any coding knowledge. The drawbacks are that it is slow in executing the test case and the presence of advertisements may lead to errors in testing.

6. Conclusion

The selection of a testing tool depends upon the tester’s requirements. There is no single testing tool that performs everything. Certain tools are best suited for certain purposes. Selenium is best suited for people with coding experience and who knows how to integrate different

frameworks as most of the advanced functionalities of Selenium requires the use of various frameworks. Sahi is for the novice user with very less to almost no coding experience. Even though Sahi does not have the advanced features that Selenium possesses, it is very easy to set up and operate. It also has certain features that are essential for a testing tool which is absent in Selenium. These include the ability to generate reports and the inbuilt support for excel framework.

7. References

1. Neha D, Shiwani S. Studying and comparing automated testing tools: Ranorex and testcomplete. International Journal of Engineering and Computer Science (IJECS). 2014 May; 3(5):5916–23.
2. Bordelon N, Simmonds D, Reinicke B, Patterson L, Noble J. A comparison of automated software testing tools. Annals of the Master of Science in Computer Science and Information Systems at UNC Wilmington. 2012; 6(1):1–77.
3. Kauar H, Gupta G. Comparative study of automated testing tools: Selenium, quick test professional and testcomplete. International Journal of Engineering Research and Applications. 2013; 3(5):1739–43.
4. Hanna M, El-Haggar N, Sami M. A review of scripting techniques used in automated software testing. International Journal of Advanced Computer Science and Applications (IJACSA). 2014; 5(1):209–17. Crossref
5. Sharma S, Jyoti V. Study and analysis of automation testing techniques. Journal of Global Research in Computer Science. 2012 Dec; 3(12):36–43.
6. Gronau I, Hartman A, Kirshin A, Nagin K, Olvovsky S. A methodology and architecture for automated software testing. IBM Research Laboratory in Haifa, Technical Report; 2000. p. 1–17.
7. Kurapati M, Sukhavasi V. Sahi – web automation and testing tool: a white paper. ZenQ Testing Summit; 2015. p. 1–11.
8. Rattan R, Shallu. Performance evaluation and comparison of software testing tool. International Journal of Information and Computer Technology. 2013; 3(7):711–16.
9. Mohamed M, El-mahdy MM. Evaluation of automated web testing tools. International Journal of Computer Applications Technology and Research. 2015; 4(5):405–8.
10. Shalu, Garg N. Web application testing with Sahi tool. Graphic Era University; 2013.