Compare of Open Source Tool using in Web Application Testing

Saurabh Dwivedi[1] Ms. Garima Gupta[2]
[1]M.Tech Student [2]Assistant Professor
[1,2]Department of Computer Science & Engineering
[1,2]School of Computing Science and Engineering Galgotias University, Greater Noida, U.P

Abstract— Web testing is the name given to software testing that focuses on web applications. Complete testing of a web based system before going live can help address several issues. Manual testing is a time consuming practice and is difficult to repeat but can't be overlooked. Each time a software does not perform according to specifications; the program will record and report the exact command that is the root cause of problem. Once the problem is identified and the bug is fixed, one can execute the very same set of commands to verify the success. There are a number of commercial and open source tools available for assisting with the development of test automation. In this paper, automated test tools named SAHI and SELINUM are proposed to support the automated test scenario for web based applications.

Keywords: Sahitool, Selinumtool, Selinumide, Rc, Webdriver, Grid, Selinum Commands

I. INTRODUCTION

In software testing, test automation is the use of special software (separate from the software being tested) to control the execution of tests and the comparison of actual outcomes to predicted outcomes. Test automation can automate some repetitive but necessary tasks in a formalized testing process already in place[1], or add additional testing that would be difficult to perform manually. With the development of internet technology, web application are becoming more complex and the scale of it is changing rapidly. The quality and reliability are getting much more attentions. The web application programs testing, especially the regression testing is much more difficult than the traditional[2]. Consequently research is required to help focus using on Browser compatibility for effectively debugging and testing web applications[3]. Therefore it becomes important to carryout research by providing tools and mechanisms that would help towards browser based testing[4]. This paper intends to contribute in this direction by proposing browser based testing tools named SAHI and Selenium [5].

II. RELATED WORK

A. Sahi Tool:

SAHI is an automation and testing tool for web application. It is open source tool. SAHI open source is written in java script .SAHI pro is currently in version 5.1.2 used to industry[6]. SAHI was born as an open source product in 2005 with specific focus on automation of emerging web 2.0 technologies but as a tool geared towards testers. SAHI Pro is truly a world class web test automation product that supports technologies such as flex applets load testing etc.

B. Selinum Tool:

Selenium is a robust set of tools that supports rapid development of test automation for web-based applications[7]. Selenium is a web testing tool which uses simple scripts to run tests directly within a browser. It uses JavaScript and I frames to embed the test automation engine into the browser[8]. This allows the same test scripts to be used to test multiple browsers on multiple platforms.

III. COMAPRE OF OPEN SOURCE TOOL

In this section, we describe COMPARE OF OPEN SOURCE TOOL like as a record id programming language support etc.

A. Recorder:

Selenium	SAHI
Works only on Firefox	Works on all browsers (IE, FF, Chrome, Safari, Opera)
Has trouble recording I Frames, Frames and popup windows	Can record on I Frames, Frames and popup windows
For Frames and I Frames, need to explicitly select Frame	Implicit smart identification of elements even across Frames and I Frames
Uses X Path for identification of elements if id or name not present	Uses various algorithms to uniquely identify elements in a simple human recognizable way

Table 1:

B. Programming Language Support:

Selenium	SAHI
Java, Ruby, Perl, Python, C# (and may be more).	SAHI Script, Java, Ruby SAHI Script has the syntax of java script but can interoperate with any Java code directly from script. The Java/Ruby drivers are available since SAHI V3
Needs language bridges for each new feature. For example, needs java bridge to invoke Flash via External API.	SAHI Script can directly invoke anything exposed by java script.

Table 2:

C. Ease of use:

Selenium	SAHI
Easy to start with because of Selenium IDE which is a firefox plugin. Estimated start time less than 5 minutes	More difficult than Selenium to start because it needs installation of SAHI, Estimated start time 10-30 minutes, depending on Java installation etc.
Deep learning curve when the need is felt to move from Selenium IDE to Selenium RC.	There is only one mode of operation for SAHI Extremely simple to learn and use for testers
Knowledge of programming	Can achieve most

language required	automation with just functions and variables. SAHI has inbuilt APIs for most other complex tasks like data driven testing
Needs J Unit style of programming	Can choose your own style
Uses X Path based identification for elements in complex html structures or those with dynamic ids. css selectors and java script may also be used.	Has nearness APIs like _in and _near which can help show nearness of elements. Eg. _image("delete.gif", _near("username 4"))
Needs waits for making AJAX work	No waits needed in 90% cases
Supports parallel execution	Inbuilt parallel execution. Needs only one parameter change

Table 3:

D. Stability of Scripts and Ease of Maintenance:

	Selenium	SAHI
Smart DOM Relations resilient to UI changes	No Dependent on X Path Difficult for testers to understand and debug	Yes Does not use X Paths. Uses _near and _in
Implicit waits for page load and AJAX: 1) Saves time 2) Keeps scripts simple 3) Reduces random failures	No Explicit waits needed.	Yes
Ease of adoption by a team of testers	Needs testers to know TestNG/Junit, XPaths, HTML structures, Frames IFrame knowledge, Javascript for AJAX conditional waits	Sahi abstracts out all these for the tester.

Table 4:

E. Dependency on Other Tools:

Selenium – Java (Others need something similar)	SAHI
Needs J Unit (and optionally eclipse) to run tests	No additional tools required. Tests run from the SAHI Controller/command line/ant
Non persistent reporting. Needs Testing or something similar for that	Persistent HTML reporting which can be shared via
	URL or file

Table 5:

F. Stability of Product and Number of Releases:

Selenium	SAHI
Started 2004(?) in Thought Works	Started 2005 in Thought Works
Version 1 took 5 years, Version 2 planned mid-2010. Moving away from original architecture to Web Driver based architecture	Current release: Version 3 Number of stable releases in 2009: 7

Table 6:

G. Footprint:

Selenium	SAHI
RC: 10.5 MB, Grid 15 MB	less than 2 .5 MB with source
Not sure	Runtime ~ 50MB for 3 parallel threads

Table 7:

H. Reporting:

Selenium	SAHI
Needs external tools to create readable reports	Inbuilt HTML reports with click through to relevant portion of script

I. Others:

	Selenium	SAHI
Build tool integration (ant, batch files)	Yes	Yes
Multiple OS support	Yes	Yes
Version Controllable Scripts/Code	Yes	Yes
HTTPS support/redirects	Not sure	Yes
401 Authentication, Windows/NTLM Authentication dialogs	Not sure	Yes
External proxy tunneling	Yes	Yes
In built APIs for data driven testing	No	Yes
Works only with browsers	Yes	Yes
Needs privileged modes on browsers for operation. (Privileged is bad)	Yes	No
Extensible on future browsers	Depends on finding a way to use privileged mode on that browser	Yes. Very little dependency on type of browser.
Editor support	Has good	Editor support

	editors in various languages	for javascript is not as good as for Java.

J. Support Available:

	Selenium	**SAHI**
Free support via Forums	Yes	Yes
Paid support available	Yes	Yes
Authoritative training available	-	Yes

Table 9:

IV. CONCLUSION AND FUTURE DIRECTION

When developing web software, the ultimate goal of the tester or developer is to ensure that the application is tested often and thoroughly. More often than not, creating automated test scripts is the best way to be sure that this goal is accomplished. In particular, the developer wants to be sure to create maintainable test scripts that will last through the many changes that applications undergo[8]. If modifying or refactoring the test script does become necessary, there are ways to make sure this job is done quickly and correctly. The main way is to avoid test duplication. By keeping specific tests self contained, they can be reused in several places and only one modification would be necessary for all instances. An Open Source test tool, Selenium IDE has many advantages, including an easy to use record and playback tool, and the ability to test JavaScript inside of the browser. However, as test cases can only be run sequentially and cannot be embedded in one another in the IDE, writing higher level test scripts can sometimes be difficult[9]. In addition, the log, which reveals whether or not tests have run successfully, evidently cannot be exported. However, all in all, the user friendly nature and the ability to customize commands via user extensions make Selenium IDE an ideal test suite development environment in many ways.

REFERENCES

[1] M. Fewster and D. Graham. Software Test Automation Effective Use of Test Execution Tools Addison-Wesley ACM Press, 1999
[2] E. Dustin, J. Rashka, and J. Paul. Automated Software Testing: Introduction, Management, and Performance. Addison-Wesley, 2004
[3] S. Berner, R. Weber, and R. K. Keller. Observations and lessons learned from automated testing. In Proceedings of the ICSE'05 New York, USA, 2005. 571579
[4] http://seleniumhq.org/docs/03_selenium_ide.jsp#id4.
[5] http://www.infoq.com/articles/testing-ajax-selenium.
[6] http://sahi.co.in/
[7] http://www.seleniumhq.org/
[8] http://en.wikipedia.org/wiki/Selenium_(software)
[9] GrigGheorghiu. "A Look at Selenium." 2005.
[10] Anta wan Holmes and Marc Kellogg. "Automating Functional Tests Using Selenium."
[11] R.R. Palacio ,A. Vizcaino ,A.L. Moran et all .Tool to facilitate appropriate interaction in global software development[J].IET software,2011,5(2):157-171.
[12] Quan Zhou, Ruixiang Bian,Yuchun Pan et al. Design Of Electric Power Web System Based On Comet[C].//2009 Second International Conference on Intelligent Computation Technology and Automation (ICICTA 2009). Volume 3A.2009:42-45.
[13] Guangzhu Jiang,Shujuan Jiang.A Quick Testing Model of Web Performance Based on Testing Flow and its Application[C].//2009 Sixth Web Information Systems and Applications Conference (WISA2009).2009:57-61

