IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Vol.-7, Issue-5, May 2019 E-ISSN: 2347-2693

Automation Testing Using Selenium+Sikuli Scripting

Ashish[1*], Nishu[2]

[1,2]Dept. of Computer Science and Engineering, NC College of Engineering, Israna, Panipat, Haryana, India

Corresponding Author: ashishlathwal4@gmail.com, Tel.: +91-8529355272

DOI: https://doi.org/10.26438/ijcse/v7i5.346351 | Available online at: www.ijcseonline.org

Accepted: 16/May/2019, Published: 31/May/2019

***Abstract*—** Automation testing is a methodology that uses an application to implement entire life cycle of the software in less time and provides high efficiency and effectiveness to the software. In automation testing the tester writes scripts by own with the help of any suitable application software in order to automate any target software application. Automation is basically an automated process that comprises lots of manual activities. In other words, Automation testing uses automation tools like Selenium, Sikuli, Appium etc. to write test script and execute test cases, with no or minimal manual involvement while executing an automated test suite. Usually, automation testers write test scripts for any test case using any of the automation tool and then group several test cases into test suites. Here, we will discuss a neat case study explaining the automation testing using hybrid test script.

***Keywords*—** Automation testing, Selenium, Sikuli, ROI (Return on Investment), Hybrid automation testing.

I. INTRODUCTION

Automation testing deals with automated manual efforts that a tester usually perform in any software application testing with the help of automation testing tools like Selenium, Appium, Sikuli [1].

Generally, automation is carried out to reduce software testing lacks in repeated tasks of various types of manual testing like UI testing, functionality testing, negative testing because automation can perform repeated steps of testing without any human intervention and without any mistake [2]. Therefore, overall performance of the testing can be improved up to one more high scale. It is quite simple to understand if we make any mistake in software testing then testing will go in wrong direction. Also, wrong testing may lead to either wrong product development or more cost to the software testing. More cost to the software testing means that some extra cost will add to the project budget to track the defect in defect tracking system (testing). So, ultimately the project budget will suffer from un-expected increase in software cost [3]. And, these are some basic reasons due to which companies are focussing on the automation of test cases in order to avoid extra cost of bug tracking in testing. One more thing we should point out that once we have implemented the automation in software testing then the requirement for the manual software testers will come to null or diminish completely. In today's era IT companies are focussing on automation along with manual testing. And the credit of automation goes to the manual testing also because automation is only carried out after manual testing. So, we cannot eliminate manual testing completely but we can introduce automation testing to improve the testing efficiency.

This paper presents a new approach to automate any test case by using hybrid automation tool with the help of Selenium and Sikuli automation testing tools [4]. As, in current era Selenium is the most popular automation testing tool in the industries but it has many limitations also which includes less automation capacity (it can't automate outside the web browser body section). And, if any automation tool is unable to automate any test case then we consider that it is the right time to move on to next automation tool. And, if any company is using only Selenium then it becomes quite difficult to change the whole environment which comprises new talent, more hardware and software requirements. So, there is one more solution to overcome the limitations of the Selenium tool that is hybrid scripting in which we will use Selenium as the base tool but along with that tool we will integrate some features of Sikuli to handle those scenarios which Selenium can't handle alone [5]. In this way we will focus on Sikuli over Selenium concept instead of Selenium over Sikul so, we will use Selenium as base tool for hybrid scripting [6]. The objective of this research is to increase the automation of test cases which can't be automated using a single scripting tool.

This research sill explain how we can use Selenium and Sikuli tools together in order to create a hybrid test script. This type of hybrid scripting will overcome the limitations of Selenium and Sikuli tools (individual tools).
This paper is organized as follows, Section I contains the introduction of the research paper, Section II contains the

architecture of Selenium + Sikuli scripting, Section III contains Proposed work or implementation of hybrid scripting, Section IV contains the comparison of this paper with earlier research or advance features of current research and Section V concludes the research with future directions and scopes.

II. ARCHITECTURE OF SELENIUM + SIKULI SCRIPTING

Any automation tester has a biggest ever challenge of deciding the automation tool [7]. Now a days, only one automation tool is selected for any automation project due to budget and other constraints. But, this Selenium + Sikuli architecture propose a good hybrid scripting approach that can easily fit in any cost effective and qualitative scripting budget [8]. Actually, automation tool selection is a time-consuming activity because it decides the automation coverage of application or we can say that as an ideal automation tool it (automation tool) should be capable to automate our application software around 90% [9]. And, if any automation software can achieve this 90% feasibility for target application software then it is preferred to use for that automation project. Here, we consider a hybrid test automation tool approach in which all the functionalities of Selenium are available to the automation tester to write any test automation script for automating any web application testing with the help of Selenium and Sikuli automation tools (hybrid scripting). Generally, if there is no requirement of Sikuli tool in automation of any software application test case then the below architecture is followed for the automation of that test case (only Selenium work from hybrid approach) [10].

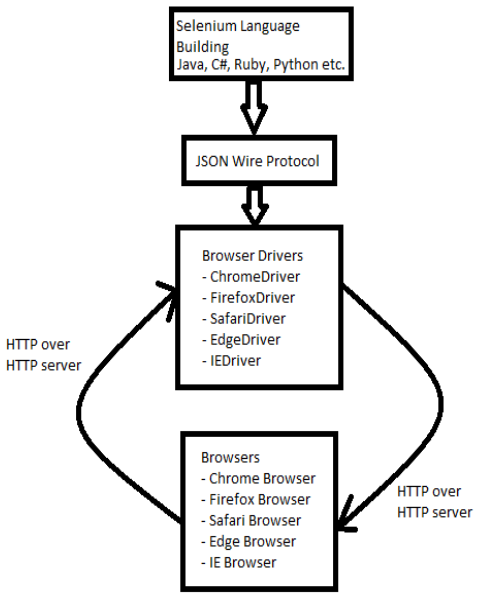

Figure 1. Core Selenium Architecture

Here, Selenium Language Building consists the languages which Selenium supports for scripting of any test case [11]. Also, the requests reach to server using HTTP protocol in JSON format. In the same way response is received in the form of JSON using HTTP protocol.

When there is the requirement of Sikuli in the scripting then, the complete handover is given to the Sikuli and automation scripting follows below architecture in that scenario [12].

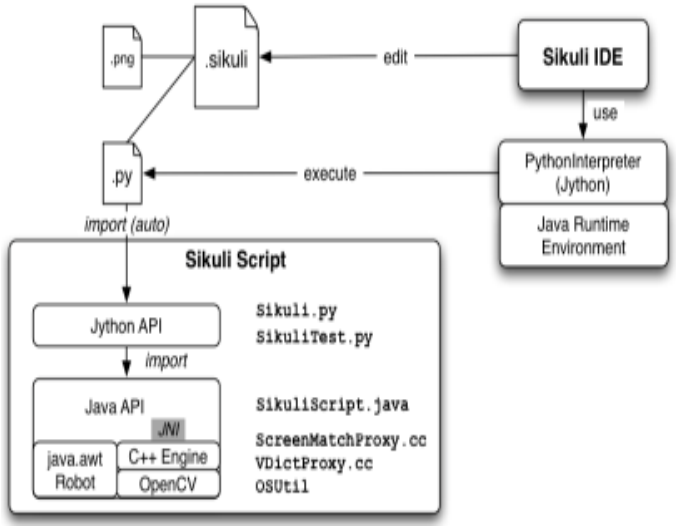

Figure 2. Sikuli Architecture

Therefore, we will have all the advantages of the Selenium. Ultimately, we can show this hybrid architecture in simple block diagrams as shown below [13].

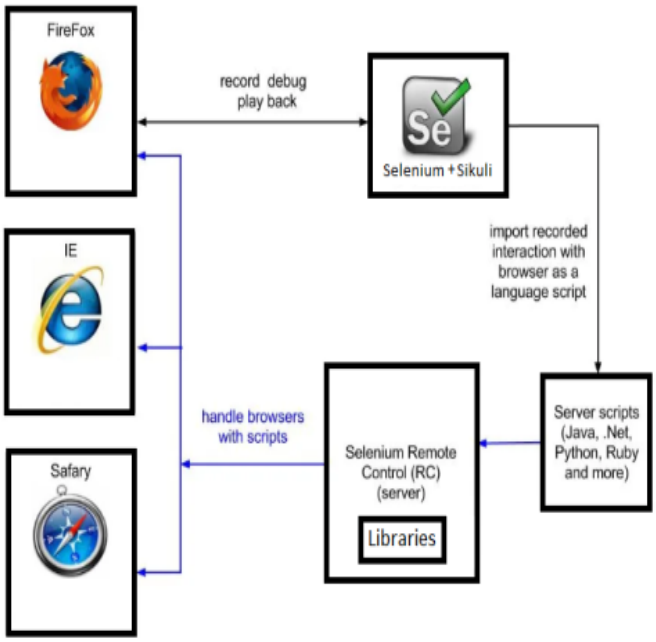

Figure 3. Selenium+Sikuli scripting

Initially, script would be designed using any of the existing programming language like Java, C#, Ruby, Python, .Net etc. As per scenario, Selenium or Sikuli is used and correspondingly response is captured [14]. Later on, the command will reach at Selenium remote control server which will pass this command to corresponding browser driver and that driver will hit the command at browser using HTTP protocol. This request reach at server end and corresponding response will appear at browser. Here, drivers may be different as per the browser on which we are running ours script. Also, up to Selenium 2.0 there was not requirement for the Firefox driver but from Selenium 3.0, Selenium has been enhanced and there is the requirement of geckoDriver.

This paper presents couple of approaches to automate various types of web projects for example: Java applet projects, platform independent test scripting with high automation coverage of any project.

III. PROPOSED WORK

Many automation projects include various types of complex scenarios and these scenarios may increase challenges in automation [15]. Some of these challenges are understandable from the most famous Agile testing environment because in present world almost every company is moving towards Agile methodology to improve theirs work load and performance of project deliverables [16].

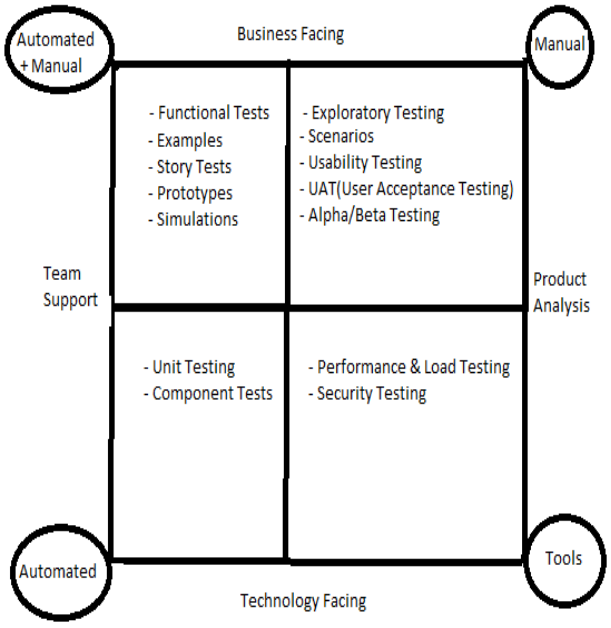

Figure 4. Automation Project

As the diagram represents the most common challenge of automation testing is the tool selection. Tool is selected very carefully, so that we can cover automation of as much test cases as possible or almost complete automation. If any test case can be automated with 90% accuracy then, it is suggested as a good automation.

Here, we have many automation tools in the market. Selenium is one of the trending automation tools which is used from small scale organisations to large scale organisations. We will discuss about the writing test script using Selenium automation test tool and the difficulties in scripting with Selenium which we will remove using Sikuli automation tool in various ways. Here are some ways of writing hybrid test script using Selenium and Sikuli tools which are following:

Core Selenium: Any test case written for web application testing which does not require to handle any window alert, any captcha, no applet-based application then, we can use core Selenium to automate such test case.

Selenium is enough powerful tool suit to automate around 80% web applications. We require a browserDriver and a "selenium-server-standalone" JAR file to automate using any of the Selenium testing tool like jUnit, TestNG etc. Following is the example for jUnit test script which automate a simple test case of home Page of a web application. We can use core Selenium to automate a web application with the help of its following features:

- Locate elements by the selector
- Retrieve their state
- Perform actions on UI

Let us consider a test case which verifies the home page of Google, it may consist following steps:

- Launch Google chrome browser.
- Hit "https://www.google.com" URL.
- Verify text field visibility on the web page before 30 seconds after hitting the URL or assert on response should receive within 30 seconds.

To automate this flow, we can write following test script in Selenium and ensure the test case will pass or fail on the basis of which we can escalate the error to our client (in case of failure):

```
private static WebDriver driver;
@Test
private void seleniumTestCase()
{
System.setProperty("webdriver.chrome.driver","chromedriver.exe_path");

driver = new ChromeDriver();
driver.manage().window().maximize();

String url = "https://www.google.com";
driver.navigate().to(url);
```



```
new WebDriverWait(driver, 60L).
until(ExpectedConditions.visibilityOfElementLocated(By.id
("textfield")));
}
```

Selenium with Sikuli: We can integrate Sikuli with Selenium in two ways (basically). Sometimes there came the requirement to handle window alerts using automation script, applet-based software application automation etc. And, these such types of scenarios can't handle using core Selenium. So, we can handle these scenarios using any third-party software. Also, Selenium supports integration of other tools so we can use this feature also for Sikuli integration. Actually, such type of integration allowed by Selenium makes it flexible in nature and more powerful tool. We can integrate Selenium with Sikuli mainly in following two ways:

- **Using DesiredCapabilities class**
- **Using Screen class**

Using DesiredCapabilities class: We can integrate Selenium with Sikuli automation testing tool easily in case of any browserDriver by passing "sikuliExtension" capability as "true" to the browserDriver. Hence, we will only pass few propertes to handover the automation handle to Sikuli.

```
// Create a DesiredCapabilities object
DesiredCapabilities cap = new DesiredCapabilities();

//Set capabilities
cap.setCapability("sikuliExtension", true);
System.setProperty("webdriver.chrome.driver","
chromedriver.exe_path ");

// Initialize WebDriver object
WebDriver dr = new ChromeDriver(cap);

//Hit the URL
dr.get("https://www.google.com/");

SikuliExtensionClient sikuliClient = new
SikuliExtensionClient( GridSettings.HOST,
GridSettings.PORT, remoteWebDriverSessionId);
sikuliClient.uploadResourceBundle(ACE_IMAGES_BUND
LE);
SikuliHelper sikuliHelper = new SikuliHelper(sikuliClient);

TextBox sikulihndl =
sikuliHelper.findTextBox("textfield.png");
sikulihndl.click();
sikulihndl.write("Search Text");
sikulihndl.press(KeyEvent.VK_ENTER);
```

Using Screen class: In this method we will use a "Screen" class which would be imported from Sikuli-script.jar. This Screen class provides many methods which can be used to perform various operations on the GUI, on the basis of image resolution. Sikuli becomes more useful when we automate a stable GUI window-based application. As, in window-based applications we cannot fetch the properties of any element by using core Selenium. So, Sikuli helps a lot in that case to solve such problems of automation. A Sikuli script allow any user to use screenshots of the screen and provide particular actionable image. With the help of Selenium + Sikuli scripting, we can perform functions like-click on any web element, enter any text in text fields, wait until any particular web component appears on the screen. We can use Screen class in Selenium in following way:

```
// Create an object of Screen class
Screen s = new Screen();

// Initialize WebDriver object
WebDriverdriver = new ChromeDriver();

//Maximize the browser window
driver.manage().window().maximize();

String url = "https://www.google.com";

//Hit the URL
driver.navigate().to(url);

//Initializing the Patterns
Pattern img1 = new Pattern("D:\\IMG\\text1.png ");
Pattern img2 = new Pattern("D:\\IMG\\text2.png ");

//wait for max. 10 sec. or until img1 pattern appear
s.wait(img1, 10);

//Click on img1 pattern on screen
s.click(img1);

//Type "Search text"
s.type("Search text");
s.click("D:\\IMG\\searchBtn.png",0);

//wait for max. 60 sec. or until img2 pattern appear
s.wait(img2, 60);
```

Here, in the last line of code script will wait dynamically for 60 seconds that is if the img2 appear before 60 seconds then script will not wait till 60 seconds and it will move ahead otherwise script will wait for img2 till 60 seconds. And, if img2 will not become visible till 60 seconds then script will show FAILURE in test case.

Hence, with above two methods we can not only integrate Selenium with Sikuli but we can also take the advantages of

Sikuli to overcome the limitations of Selenium. Here, we only require the installation of an extra "Sikuli" software in order to use Sikuli with Selenium for automation of any test case.

Therefore, once we would be able to use this (Selenium + Sikuli) hybrid automation scripting then ultimately, we will be capable enough to automate any project (automation of more than 90% test cases) and as much we would be capable to automate any project in same proportion, we would increase the project efficiency.

IV. COMPARISON WITH EARLIER RESEARCH

There are multiple advantages of hybrid scripting. Below table represents few advantages of the hybrid

Table 1: Comparison Table

(Selenium / Sikuli) Earlier Research	Proposed Work Improvements (Selenium+Sikuli)
Multithreading is not possible in Sikuli	Multithreading is possible
Background functioning is impossible using Sikuli	Background functioning is possible
Desktop application's automation testing can't be done using Selenium	Desktop application automation testing is possible
Unable to handle window-based alerts in Selenium	Enable handling of window alerts
API testing is not possible in Sikuli	API testing is possible
Auto-report generation is not possible using Sikuli	Auto-report generation is possible
Error screenshots cannot be captured by Sikuli.	Error screenshots can be captured.

automation test scripting using Selenium + Sikuli tools over individual use of Selenium / Sikuli automation test tools for writing automation test scripts in any programming language [17].

These advantages of hybrid scripting increase the scope of automation of any project or we can also consider it as the increase in automation coverage of those test cases also which were not covered earlier due to project budget constraints. Even after so many advantages of hybrid automation tool captcha handling and image resolution dependency during Sikuli execution are still impossible to automate [18].

V. CONCLUSION

Automation scripts reduce the manual efforts and improves the efficiency of any software application testing. Automation testing can reveal real time defects. Selenium and Sikuli are freeware tools that can be used to automate web sites. But both the tools have some limitations. These limitations can be overcome with the hybrid test script (Selenium + Sikuli). This (Selenium + Sikuli) script does not affect the quality of the test script. It makes the test script capable to handle complex scenarios as well. Also, integration of Selenium with Sikuli is easy to implement which makes it a quite beneficial new test framework that has various good qualities from both the automation tools. Selenium with Sikuli becomes a powerful tool that can automate almost all scenarios except a few like Google captcha handling as the captcha response comes in the form of an image that does not give the captcha code information, that information is stored only at server end.

REFERENCES

[1] Sarika Chaudhary, "Latest Software Testing Tools and Techniques: A Review ", International Journal of Advanced Research in Computer Science and Software Engineering, Volume 7, Issue 5, May 2017 http://www.softwaretestingclass.com/software-testing-tools-list/.

[2] M. Jovanovic, Irena, "Software testing methods and techniques," IPSI BgD Journals, vol. 5, 2009.

[3] Shruti Malve, Pradeep Sharma, "Investigation of Manual and Automation Testing using Assorted Approaches", International Journal of Scientific Research in Computer Science and Engineering, Vol.5, Issue.2, pp.81-87, April (2017).

[4] T. Yeh, T. H. Chang and R. C. Miller, "Sikuli: Using gui screenshots for search and automation", In the Proceedings of the 22nd annual ACM symposium on User interface software and technology Pages 183-192.

[5] Nisha Gogna, "Study of Browser Based Automated Test Tools WATIR and Selenium", International Journal of Information and Education Technology, Vol. 4, No. 4, August 2014.

[6] Miika Kuutila, "Benchmarking configurations for Web-testing-Selenium versus Watir", researchgate, November 2016.

[7] P. Raulamo-Jurvanen, K. Kakkonen and M. Mäntylä, "Using Surveys and Web-scarping to Select Tools for Software Testing Consultancy", In Proceedings of the 17th International Conference on Product-Focused Software Process Improvement, 2016.

[8] A Surabhi Saxena, Devendra Agarwal, "Realiability Assessment Model to Estimate Quality of the Effective E-Procurement Process in Adoption", IJSRNSC, Volume-6, Issue-3, June 2018.

[9] Michiel Van Genuchte, "Why is software late? An empirical study of reasons for delay in software development", IEEE Trans. on Software Eng. IEEE, Vol 17, Issue 6, pp.582-590, 1991.

[10] N. Uppal and V. Chopra, "Design and implementation in selenium ide with web driver", International Journal of Computer Application, vol. 46, 2012.

[11] Shilpa Garg, Paramjeet Singh, Shaveta Rani, "Comparative Study of Selenium WebDriver and Selenium IDE (Integrated Development Environment)", International Journal of Computer Sciences and Engineering, Vol.-6, Issue-7, July 2018.

[12] Inderjeet Singh and Bindia Tarika, "Comparative Analysis of Open Source Automated Software Testing Tools: Selenium, Sikuli and Watir", International Journal of Information & Computation Technology, Volume 4, Number 15 (2014), pp. 1507-1518.

[13] Revathi.K, Prof.V.Janani "SELENIUM TEST AUTOMATION FRAMEWORK IN ON-LINE BASED APPLICATION",

International conference on Science, Technology and Management, 2015.
[14] Samiksha R. Rahate, Uday Bhave, "A Survey on Test Automation", International Journal of Innovative Research in Computer and Communication Engineering, Vol. 4, Issue 6, June 2016.
[15] Shuang Wang and Jeff Offutt, "Comparison of unit-level automated test generation tools in Software Testing, Verification and Validation Workshops", ICSTW'09, International Conference on IEEE, 2009.
[16] Sonia Thakur, Amandeep Kaur, "Role of Agile Methodology in Software Development", International Journal of Computer Science and Mobile Computing, IJCSMC, Vol. 2, Issue. 10, pg.86 – 90, October 2013.
[17] K. M and K. R, "Comparative study of automated testing tools: Testcomplete and quicktest pro," International Journal of Computer Application, vol. 24, 2011.
[18] Suliman A. Alsuhibany, "Evaluating the Usability of Optimizing Text-based CAPTCHA Generation", IJACSA, Vol. 7, No. 8, 2016.

Author's Profile

Mr. Ashish pursuing Masters of Technology from N.C. College of Engineering, Israna, Panipat, Haryana, India in year 2017-19. He has published literature survey on hybrid automation testing using Selenium and Sikuli tools. He has more than two years of professional working experience in manual and automation testing. He has already worked in performance testing for some web applications also. His main research focuses on software testing.

