International Journal of Recent Technology and Engineering (IJRTE)
ISSN: 2277-3878 (Online), Volume-10 Issue-1, May 2021

Scriptless GUI Automation Testing Tool for Web Applications

Kamble Namita Mohan, Ramakanth Kumar P

Abstract*: Software tests must be repeated frequently throughout development cycles to attain certain quality. Every time program code is changed software assessments need to be repeated. Once created, automated tests may be run repeatedly at no extra value and they may be tons quicker than manually conducted test and free from human errors. Automated software program testing can lessen the time to run repetitive tests from days to hours. Test automation can easily run thousands of different complex test cases in each test run, so there is no manual testing involved. But Automation testing has its own disadvantages one of it is that the testers should come from a programming background. To eliminate this dependency over programmers Scriptless automation testing tools are emerging. There are many Scriptless GUI automation testing tools in the market that use various methods to achieve the goal, this paper proposes a new record and playback method to achieve the same using Selenium framework and JavaScript for web application.

Keywords*: GUI Automation Testing, Scriptless, Locators, Data-Driven Architecture.

I. INTRODUCTION

Traditionally, automated testing requires the development of complex scenarios, and these scenarios usually require strong technical skills. In addition, these test cases are usually "fragile", which means they will be interrupted if the application changes or the application is launched on another device. Hence the call for Scriptless automation platforms, which provide companies with a simple and inexpensive way to take advantage of automated testing capabilities without the complexity or cost of scripting.

GUI testing is a method of testing GUI software to identify errors that have been overlooked during the design and development stages. It tests the functionality of the GUI according to the specifications and according to the technology used. GUI testing also organizes third party developers or users to evaluate controls, such as menus, buttons, icons, text boxes, lists, dialog boxes, layouts, colors, font sizes, text formats, etc. It is used to run attribute values for each GUI element and perform GUI operations, such as: Press a key or click the mouse.

In this modern era of GUI automation testing, tools are very high maintenance as they generate scripts and store them. Auto generated codes are a problem because small changes to the test cases require that all the test scripts need to be updated or recorded. Remote execution, parameterization, configuration, image comparison, blinking control identification, test management integration are few of the features you can use to create highly reliable and affordable automation. Unfortunately, most recording and playback tools lack most of these features. In this paper a new method to perform the GUI testing is proposed that does not require any programming skills. The proposed architecture is data-driven framework that does not generate any scripts hence no storage or maintenance issues arise.

Following are the main features of the proposed Scriptless GUI Test Automation Tool.

User can record actions performed on the test application. Users can re-run the recorded actions. Generate basic reports with screenshots in MS-Word for each re-run. Browse and execute the recorded Test case. Re-run multiple recorded test cases. Perform assertions like Image Comparison.

II. RELATED WORK

Panchuri Ramya, et al. They show that automation is the easiest way to test an application. One of the best advantages of test automation is that the software is reusable and all test scripts can be easily documented and kept up to date. [1] The Selenium web driver/controller for testing web applications is very efficient, simple and the results obtained are more accurate. It supports the use of other tools like construction tools etc besides itself for ease of use and precision.

Toshiyuki Kurabayashi, et al. Suggested a method to automatically generate test scripts from source code and test targets to solve the problem that test scripts take time to create manually. The method meets the following three requirements. Set up and operate the tool. View the screen transition for the test script that could not be auto-generated for the user.[6] Testing of all screen transitions with automatically generated tests and manually supported tests. We confirm by evaluation that the method reduces the number of man-hours by approximately 61% compared to a method that generates test scripts using the traditional capture and play method. Rakesh Kumar Lenka, et al., state that the automation testing tool is really useful for dynamically changing web applications. It reduces the time spent by the tester for writing the test cases.

Manuscript received on May 12, 2021.
Revised Manuscript received on May 22, 2021.
Manuscript published on May 30, 2021.

* Correspondence Author

Kamble Namita Mohan*, Department of Computer Science and Engineering, Rashtreeya Vidyalaya College of Engineering, Bengaluru (Karnataka), India. Email: kamblenamita20@gmail.com

Dr. Ramakanth Kumar P*, Department of Computer Science and Engineering, Rashtreeya Vidyalaya College of Engineering, Bengaluru (Karnataka), India. Email: ramakanthkp@rvce.edu.in

Automation testing tool is the best way to improve the productivity, effectiveness, and coverage of software testing. It automatically generates a customized report, easy to analyze, easy to understand the bugs, less maintenance. There are a wide variety of tools available to test the software in terms of performance, service testing, functioning, load, security and stress testing of web applications. [5]

III. INTRODUCTION TO KEY TECHNOLOGY

A. Scriptless automation Tool

A Scriptless test automation tool provides an abstraction of the underlying test automation code so that building a test focuses more on validating the business logic. The human effort involved in creating a test is drastically reduced: instead of writing code, it is simply indicated which steps need to be taken. The Scriptless Tool then internally translates these easy-to-use steps into the actual running code behind the scenes during test execution. The end result: a highly accessible interface that allows you to define your tests without writing any code. Ideally, a Scriptless test tool should have the following functions; easily generate test steps in a test case. Save reusable elements or objects Record test cases. Execute test cases. Execute test cases in multiple browsers / versions. Report test results and integrate them into our other tools.

B. Data-Driven Architecture

Data-Driven architecture, conceptualizes an automatic framework where testing is triggered on a group of information stored in databases. This framework resolves time-consuming and the protracted method of making individual tests for every test case execution.

Data-Driven Tool constitutes of a methodology where a sequence of test steps structured in test code as reusable test logic are machine-controlled to run repeatedly for various permutations of information picked up from data sources so as to match actual and expected results for validations.

Following are sequence of actions that occur in Data-Driven architecture while running a test case. Firstly, Retrieving test data from a storage facility such as file or a database. Second, Execution of the test logic that's capable of reading this data, passing it through needed layers of the code that then mechanically triggers simulation of actions. Third, storing the retrieved results so to retrospect the results in terms of what was expected and what was truly obtained.

C. Selenium C# Webdriver

When cross-browser testing comes into picture, Selenium is a widely used set of tools. Selenium cannot automatically execute desktop applications. It can only be used in a browser. It is considered as one of the most popular toolkits for automated web application testing because it supports popular web browsers. The versatile web browser compatibility makes it very powerful. It is compatible with a number of browsers (Google Chrome 12+, Internet Explorer 7, 8, 9, 10, Safari 5.1+, Opera 11.5, and Firefox) and operating systems (Mac, Windows, Linux/Unix). To run test cases in different browsers Selenium Webdriver is a well known and widely adapted framework. The tool can automate the testing of web applications to ensure that they work as expected.

IV. MODULE DESCRIPTION

A. Test Case Creation Module

When recording is started by clicking on record button, the timers start running and the timers start detecting the elements wherever the mouse hovers. The automation framework has inbuilt features that help to generate the DOM Html tree structure in other words the xpath; that is the path of the element on the application under test. When we hover over the element, the element gets highlighted and it captures the details of the Element (control) like name, id, xpath, class. Once the user has performed the action on the element the details pertaining to the action is also captured in the form of string with required delimiters and saved into the database.

B. Saving of Test case Data Module

In the first module the Test case Data is collected that is the element properties and the action performed on the element. This data is saved in the form of string. This string is further divided into events; events are nothing but the substrings of the string, and each event like mouse click action or keyboard strokes are separated and stored as individual events in the database. Each event contains again more detailed information of that event.

C. Test case execution and Test case Data retrieval Module

When a user clicks on the play button, he is prompted to choose the test case he would like to execute. From the database, the details with respect to the events of the selected test case are retrieved and stored in an object. If it encounters mouse event another function will select the window and goes inside window. UI Automation Framework has methods that retrieve the Tree-view data using which the element is selected inside the window. Findby Descendents method searches for the element inside the application window and the method returns the automation id of the element. Once the automation id of the element has been retrieved the method checks whether the element has parent id or child id and if so the desired action is performed on the element for example mouse click action is performed on the button element.

V. IMPLEMENTATION DETAILS

This section explains detailed design specifications for the sub modules / components of Automation Tool. Hardware requirements for the system are as follows, Intel compatible processor with a clock frequency of at least 860 MHz, with minimum 8 GB of RAM. Minimum hard disk space shall be 5 GB. The front end structure of the tool is developed as a WPF application using C# Language in Microsoft Visual Studio 4.5. The structure is designed in such a way that it can be extended to support windows application also. Following is the System architecture depicting the major components and data flow of the application.

Retrieval Number: 100.1/ijrte.A58490510121
DOI: 10.35940/ijrte.A5849.0510121
Journal Website: www.ijrte.org

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication

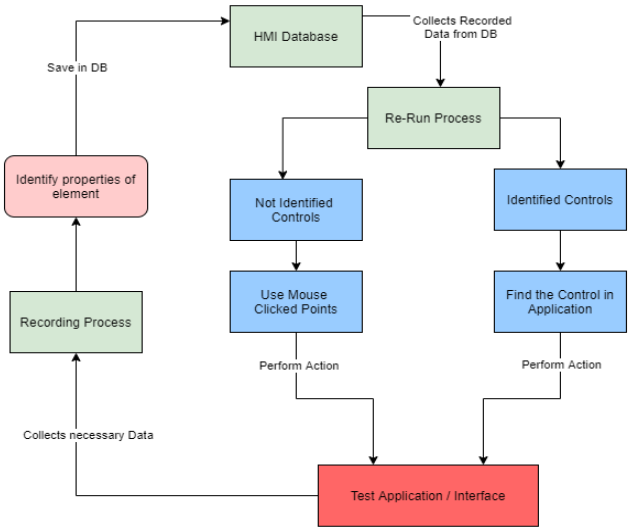

Figure1. System Architecture

The method of retrieving the locator of the web elements and capturing the action performed on the web element has been achieved using JavaScript. The data with respect to every test data is saved in database. The database used is SQLite database and is stored in the form of string along with delimiters.

1.1 Record Test case – Design

This module records actions performed on the test application.

1.1.1 Responsibilities/ Functionality

Capture user actions (Mouse & Keyboard) performed on test application.
Create or add assertions events.
Save recorded test case data to database.

With the help of test case data saved in the database for example testcase id, testcase name, action performed, web page url etc all this data is used at the time of execution. Firstly the URL is retrieved and the webpage is loaded in the browser. Next the web element is located on the web page with help of locators example xpath. Further check is performed for the action on the web element. If the action saved in the database is 'KB' that is keyboard action then it retrieves the necessary data like the text to be entered, example 'ABBC' and perform the keyboard strokes on the web element such as 'Textbox'. When all the events of the test case are performed successfully then the test case is marked as passed and the result for each test case execution is saved in the database.

1.2 Test case Re-run – Design

This module performs automated actions on the test application, as recorded by the user.

1.2.1 Responsibilities/ Functionality

Get recorded test case actions from the database.
Perform automated actions.
Generate report.

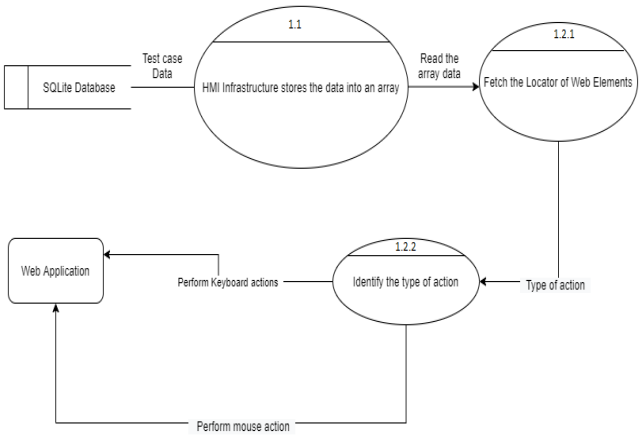

Figure2. Data Flow Diagram for Re-run process

VI. EXPERIMENTAL RESULTS

Manual Testing was performed on the Proposed Tool to measure the performance of the tool with respect to the response time and maintenance cost. The other open-source tools in the market like Katalon Studio and Ranorex exhibit same execution time but these tools come with High Maintenance cost. The proposed tool cannot automate windows application whereas Ranorex supports both desktop and web application. Ranorex requires prior advanced scripting knowledge to use the tool whereas the proposed is completely Scriptless that is no prior knowledge of programming is required neither any scripts are auto generated when recording the test cases. The following table depicts the detailed performance analysis of the proposed tool.

Proposed sytem	**Overall passed test case rate (%)**	**Overall failed test case rate (%)**	**No. of test cases**	**Execution Time (hrs.)**	**Maintenance Cost**
Data-Driven Selenium based architecture	97.41	3.68	20	3.3	Low

VII. CONCLUSION

The proposed Data-Driven architecture in combination with Selenium Webdriver framework makes this GUI Testing Tool not only machine-handled that is automatic but also makes it script free and low maintenance due to low storage requirements. This solution is Unique and very cost effective.

The proposed system does not generate scripts for every test case instead the same data-driven logic is executed for various test cases that is for different test case data.

Selenium Webdriver adds feather to the cap by identifying the web elements on the browser application faster and accurately. Xpath is the best locator to identify the web elements. The actions are performed on the controls that is on the web elements irrespective of their screen co-ordinates or the changing positions of the controls on the web application under development.

Further work can be done to integrate support for automating windows application using UI Automation framework.

ACKNOWLEDGMENT

This study and research is carried out in R.V. College of Engineering, Bengaluru, Department of Computer Science and Engineering, under the guidance of Head of the Department Dr. Ramakanth Kumar P. I also thank the principal of the institution, Dr. K N Subramanya for providing us all the required facilities and suitable environment to successfully complete this research.

REFERENCES

1. Paruchuri Ramya, Vemuri Sindhuri, P Vidya Sagar, “Testing using Selenium Web Driver”, Second International Conference on Electrical, Computer and Communication, pp. 1-7, 2017.
2. Takamasa Tanaka, Hidekazu Niibori, Li Shiyingxue, Shimpei Nomura, Tadayoshi Nakao, Kazuhiko Tsuda, “Selenium based Testing Systems for Analytical Data Generation of Website User Behavior”, IEEE International Conference on Software Testing, pp. 1-6, 2020.
3. Vidroha Debroy, Lance Brimble, Matthew Yost, Archana Erry, “Automating Web Application Testing from the Ground Up: Experiences and Lessons Learned in an Industrial Setting”, IEEE 11th International Conference on Software Testing, pp.1-9, 2018.
4. Nicey Paul, Robin Tommy, “An Approach of Automated Testing on Web Based Platform Using Machine Learning and Selenium”, IEEE Xplore Compliant Part Number:CFP18N67-ART; ISBN:978-1-5386-2456-2, pp. 1-6, 2018.
5. Rakesh Kumar Lenka, Utkalika Satapathy, Meenu Dey, “Comparative Analysis on Automated Testing of Web□based Application”, International Conference on Advances in Computing, Communication Control and Networking, pp. 1-6, 2018.
6. Toshiyuki Kurabayashi, Muneyoshi Iyama, Hiroyuki Kirinuki, Haruto Tanno, “Automatically Generating Test Scripts for GUI Testing”, IEEE International Conference on Software Testing, pp. 1-5, 2018.

AUTHORS PROFILE

Kamble Namita Mohan, is a Master in Technology student at Department of Computer Science and Engineering, Rashtreeya Vidyalaya College of Engineering, Bengaluru, Karnataka, India.

Dr. Ramakanth Kumar P, is Professor and Head of the Department at Department of Computer Science and Engineering, Rashtreeya Vidyalaya College of Engineering, Bengaluru, Karnataka, India.