Test Case Recording using JavaScript for Automation Testing

T Kishan Singh, Pavithra H

Abstract*: Test automation is the usage of software to monitor the execution of experiments and the comparison of actual results with expected results, apart from the software being evaluated. In a formalized web testing method already in place, test automation may automatize certain routine but required tasks and carry out extra testing which can be troublesome to try and perform manually. There is a need to simplify the test case recording process as most of the methods used at present requires the testers to generate scripts using any coding language. Hence, a light weight script less model is developed which uses JavaScript to record mouse and keyboard actions. Selenium framework is used to induce the JavaScript in the browser. Therefore, this paper focuses on developing a lightweight script less model for recording test cases on browsers.

***Keywords*: Test Automation, JavaScript, Selenium Web Driver, Chrome Automation.**

I. INTRODUCTION

Test automation shall automatize repetitive tasks which are required to perform testing in a formalized testing method which is in place, and accomplish further testing which might be troublesome to try and perform manually. Automated testing is crucial for testing and delivering products continuously.

Test automation monitors the carrying out of experiments and compares the actual results with expected results, apart from the software being evaluated. In a formal testing method already in place, test automation could automatize certain routine but required tasks and perform extra testing which is troublesome to perform manually.

Web testing automation tools have record and rerun functionality that allows users to record user actions and re-run them any number of times, comparing test results to the predicted results. The benefit of the strategy is that there is almost no software development needed. It is possible to extend this approach to any program that has a GUI. Dependency on the features poses maintainability problems and major reliability issues. The test can need to be re-recorded by re-labelling buttons or shifting it to a different section of the window. Record and re-run features sometimes add wrongly recorded activities or irrelevant activities.

This variant is used for web site research. The web page is the "interface". This model, however, uses completely different approaches since HTML is rendered and the DOM events are listened to in place of OS events. Selenium Web Driver-based headless browsers or solutions are usually used for this purpose.

Script-less test automation is another variation which does not make use of recording and re-run, but it generates an application model and then allows tester to generate test cases by merely performing the actions that do not include scripting abilities.

Software testing has always been a time-consuming activity, despite the availability of the most proficient quality assurance teams and instruments. In most software industries, test automation is also thoroughly practiced in order to leverage the overall development period. Although the automation of tests has its own merits and de-merits and affects other stages of growth.

In particular, management is highly interested in estimating its impact on the time, cost and quality of total software. Since automation by itself a costly operation, it needs efforts and considerable time, on three separate software, and have tried to list the effects of test automation on the budget, time frame and quality of the software services rendered. The outcome of the experiment clearly illustrates the beneficial outcomes of test automation on software budget, quality and time-frame.

I. RELATED WORK

Maurizio et al. [1] have built a method to generate DOM locator (usually indicated as XPath) to recognize the active elements of the web application and the web application data which is used in declaration. Sophisticated, the cost of development and testing is higher. The defective locator of the target element cannot be found in the later versions of the application. By using the best greedy algorithm to create a reliable XPath locator as a graphical problem, the problem can be solved automatically. With space complexity and exponential time, a generic algorithm is generated.

Maurizio et al. [2] have developed a novel algorithm so as to decrease the growing old of web test cases with the aid of using routinely producing strong XPath locators that shall work while new versions of the application are developed. Preliminary analysis has proven that XPaths produced with the aid of using ROBULA are appreciably robust than absolute than relative locators, generated with the help of using FirePath. Jin-lei et al.

Manuscript received on May 06, 2021.
Revised Manuscript received on May 13, 2021.
Manuscript published on May 30, 2021.

* Correspondence Author
T Kishan Singh*, Student, Department of Computer Science and Engineering, Rashtreeya Vidyalaya College of Engineering, Bengaluru, India. Email: kishansinghtks@gmail.com
Pavithra H, Assistant Professor, Department of Computer Science and Engineering, Rashtreeya Vidyalaya College of Engineering, Bengaluru, India. Email: pavithrah@rvce.edu.in

[3] are researching a method to record capture-replay generation using primarily based totally at the open supply tool—Sikuli, that generates scripts with precise clarity primarily based totally at the testers operations and may report and playback any visible interface. Toshiyuki et al. [4] have built a way for routinely producing test scripts by the use of static evaluation and dynamic evaluation on the application executable documents and the source code is proposed. At the end result of the assessment experiment, it's far proven that this approach can lessen the quantity of man hours approximately by 61% in comparison with the traditional approach of making test scripts manually.

Joseph [5] are researching on a key hassle in data extraction which large-scale empirical analyses of the reasons of shift and mathematically are carried out to quantify the stages of domain difficulty primarily based totally on entropy. XTreePath annotation approach is employed to seize the contextual node data from the training DOM. It is then making use of this annotation in a supervised way at some point of the take a look at time with the proposed Recursive Tree Matching approach which locates nodes maximum comparable in context recursively the use of the tree edit distance. The seek is primarily based totally on a heuristic characteristic that takes under consideration the similarity of a tree in comparison to the structure that turned into a present inside the training data. XTreePath has evaluated the use of 117,422 pages from seventy-five numerous websites in eight vertical markets.

Shin-Jie et al. [6] are researching on expanding Selenium IDE and endorse a method to deal with the said problem. Selenium IDE is a wonderful contemporary integrated development environment used to perform recording and rerunning web test cases using selenium. However, it has stopped support to find popup windows which are unnamed and it also does not find internal frames helps developing a series of check instructions throughout distinct windows and internal frames.

Mari et al [7] are researching on an XPath authoring the aid for an annotation editor, and it explains the varieties of XPath expressions. An empirical assessment of the XPath expressions is then presented. Finally, the benefits and boundaries of the XPath expressions are taken to the limelight, via way of means of taking account of the real web page modifications, and look at opportunities for a similar development of the addressing method.

II. KEY TECHNOLOGIES

A. Test Automation

The demand for projects to be completed faster has increased more than ever as technology is evolving at a rapid rate. The full processes followed during a software life cycle must also be speeded up in order to get projects completed quickly. Automation may be introduced in the field of software testing to save cost and time, but only when used in projects that take time. When it comes to performing regression testing, large scale testing, automation testing is the way to go. It can be a good choice.

Test automation has several essential benefits, such as increasing the standard of the software and reduces the software testing operations manually and also eliminating redundant testing efforts, creating additional systematic repeatable software tests, minimizing repetitive work and generating more consistent test results, improving consistency.

Perform further software tests and reach a very small timetable of improved testing coverage. Productivity increases and so on. It is an efficient way to eliminate/read manual effort during regression and to conduct usable test case execution. Plus, the chances of defect escape will be greatly reduced because once the script is highly established, errors will not occur. Well, given the reality of quality and time, automation would be considered more economical in the coming days.

B. Selenium WebDriver

Selenium WebDriver is an open-source collection of APIs used for testing various web applications. The Selenium WebDriver tool automates web application testing to verify its correctness. broadly supports browsers like, Chrome, Firefox, Internet Explorer, Safari and. It allows the execution of cross-browser testing.

WebDriver allows the usage of a programming language in creating different test scripts.

WebDriver API provides various built-in methods to identify different Web elements based on different properties like ID, Class, Name, link Text, XPath, CSS Selectors, etc.

C. XPath

WebDriver API provides various built-in methods to identify different Web elements based on different properties like ID, Class, Name, link Text, XPath, CSS Selectors, etc.

XPath can be designed and generated in many different ways and the simplest way to generate XPath is by using tags, All the tags from the parent to the element can be used with a delimiter in between. These XPaths are robust as they do not change with a new version of the application until and unless the structure of the document has been changed.

D. SQLite Database

SQLite a library which is in-process and it implements a SQL transactional database engine which is serverless, zero-configuration required and self-contained. SQLite is an embedded engine for SQL databases. It explicitly performs read and write operations on the ordinary disk files.

E. HTML Document Object Model (DOM)

The web browser creates a DOM of the web application when it is loaded. The DOM is a language and a platform neutral interface which allows the scripts and applications to access dynamically as well as amend the content, structuring, and styling of an application. The HTML DOM model can be represented by a tree of Objects. DOM allows JavaScript to respond to HTML events.

A JavaScript can be executed when an event occurs, like when a user clicks on an HTML element.

III. PROPOSED MODEL

A new recording process for web application testing, Selenium framework is used to induce the JavaScript code for recording and is used for running the test cases, JavaScript model is used for capturing data of user actions and web element locators, Protractor framework is used for handling angular JS use cases.

The different JavaScript event handler functions that can be used to listen to events are onclick, onkeydown, onkeyup, ondrag, ondrop, onresize, onscroll etc.

These event handlers are used to extract the properties of the web elements where the action is performed.

The following are the steps in which the process is carried out:

1. Once the HMI is launched, the JavaScript is injected into the chrome browser using the selenium web driver.
2. The chrome browser performs "record" operation.
3. It generates Strings and queries.
4. It then inserts the table query into the SQLite Database.

IV. IMPLEMENTATION DETAILS

A combination of JavaScript and WPF application using C# is used as the development platform.

Selenium JavaScript Executor shall be used to inject JavaScript into the web browser. Event handlers in JavaScript are used to save & capture the user actions such as left mouse click, right mouse click, scroll, resize of the web application and keyboard strokes which performed on the web application.

JavaScript captures the web page URL, web element properties and the type of actions performed over the web elements. JavaScript HTML DOM Event Listeners will look for mouse events and keyboard events on the DOM object model. JavaScript sends the above data to C# WPF application which saves it in the database.

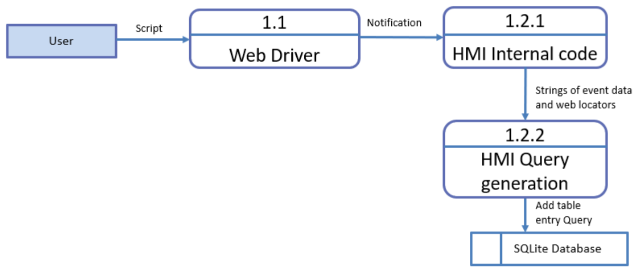
SQLite Database will be used to save the information as it is serverless and self-contained.

Assertion events such as image comparison can also be performed using the Human machine Interface (HMI) application which is then saved in the SQLite database as a test step.

Selenium framework is used to rerun the saved data using web element locators and XPath. Selenium reads the Database for the Web Application test case, i.e., if browser type is chrome and fetches the Web Application URL from the table. It then uses the data and web element properties saved in the database to perform user actions.

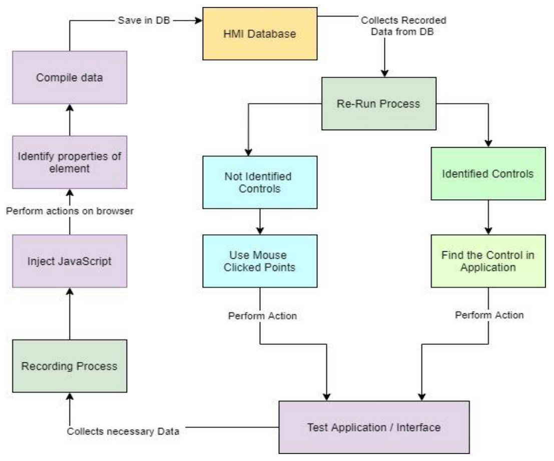
The C# WPF application performs the Verification or validation steps using the assertion data stored in the database.

The results of all the test steps are combined and a report is generated with the test case results.



Data flow diagram

The data flow diagram of the proposed model is shown in the figure 1.



System Architecture

The system architecture of the proposed model is shown in the figure 2.

Table- II: Experimental Results

Sl. no	Element Locator	No of test cases executed	No of test cases passed	Accuracy (%)
1.	Class name	50	42	84%
2.	ID	50	38	76%
3.	XPath	50	48	96%
4.	Name	50	40	80%

V. EXPERIMENTAL RESULTS

Test cases were recorded using different web element locators and were run on the same and different versions of the application being tested. Table 1 shows the results of the test case runs.

The test cases which were recorded using XPath developed in this paper shows greater accuracy than other locators such as class and ID. As the class and ID are not present for each and every element of a web application.

The XPath generated has proved to be robust and very reliable in practice.

VI. CONCLUSION

This paper has proposed a scriptless web automation test case recording model which is very robust, reliable and independent. The objective of the model is to simplify the testing process and thus improving the efficiency by reducing man hours of testers and thus reducing the time required to deliver updated version of an application.

The proposed model is a light weight solution which can be also be used on low end systems and work without utilizing much CPU cycles thereby not reducing the application performance. Thus, this can also be used on development systems which have less storage and CPU performance.

The model when compared to other models is much more accurate, reliable and give the users a more stand-alone solution to automation testing.

ACKNOWLEDGMENT

This study and research is carried out in Department of Computer Science and Engineering, Rastreeya Vidyalaya College of Engineering, Bengaluru, Karnataka, India, under the guidance of Prof. Pavithra H. I also thank our Head of the Department Dr. Ramakanth Kumar P and 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. M. Leotta, A. Stocco, F. Ricca and P. Tonella, "Meta-heuristic Generation of Robust XPath Locators for Web Testing," *2015 IEEE/ACM 8th International Workshop on Search-Based Software Testing*, Florence, Italy, 2015, pp. 36-39.
2. M. Leotta, A. Stocco, F. Ricca and P. Tonella, "Reducing Web Test Cases Aging by Means of Robust XPath Locators," 2014 IEEE International Symposium on Software Reliability Engineering Workshops, Naples, Italy, 2014, pp. 449-454.
3. Jin-lei Sun, Shi-wen Zhang, Song Huang, Zhan-wei Hui, "Design and Application Of A Sikuli Based Capture-Replay Tool", Proceedings of the IEEE International Conference on Software Quality, Reliability and Security Companion, Lisbon, Portugal, 2018, pp 42-44.
4. Toshiyuki Kurabayashi, Muneyoshi Iyama, Hiroyuki Kirinuki, Haruto Tanno, "Automatically Generating Test Scripts for GUI Testing", Proceedings of the 2018 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW), Vasteras, Sweden, 2018, pp 146-150.
5. Joseph Paul Cohen, Wei Ding, Abraham Bagherjeiran, "XTreePath: A generalization of XPath to handle real world structural variation." *arXiv: Information Retrieval,* 2015.
6. Shin-Jie Lee, Jie-Lin You and Sun-Yuan Hsieh, "Automatically Locating Unnamed Windows and Inner Frames for Web Regression Testing", Proceedings of the IEEE International Conference on Applied System Innovation, Sapporo, Japan, 2017, pp 184-187.
7. M. Abe and M. Hori, "Robust pointing by XPath language: authoring support and empirical evaluation," 2003 Symposium on Applications and the Internet, 2003. Proceedings., Orlando, FL, USA, 2003, pp. 156-165.
8. M. Leotta, A. Stocco, F. Ricca, and P. Tonella. "Using multi-locators to increase the robustness of web test cases." In Proceedings of 8th International Conference on Software Testing, Verification and Validation, ICST 2015. IEEE, 2015.
9. M. Leotta, D. Clerissi, F. Ricca, and P. Tonella. "Visual vs. DOM-based web locators: An empirical study." In Proceedings of 14th International Conference on Web Engineering (ICWE 2014), volume 8541 of LNCS, pages 322–340. Springer, 2014.
10. K. Bajaj, K. Pattabiraman and A. Mesbah, "LED: Tool for Synthesizing Web Element Locators," *2015 30th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, Lincoln, NE, USA, 2015, pp. 848-851
11. Paruchuri Ramya, Vemuri Sindhura, "Testing using Selenium Web Driver", Proceedings of the Second International Conference on Electrical, Computer and Communication Technologies (ICECCT), Coimbatore, India, 2017.
12. Toshiyuki Kurabayashi, Muneyoshi Iyama, Hiroyuki Kirinuki, Haruto Tanno, "Automatically Generating Test Scripts for GUI Testing", Proceedings of the 2018 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW), Vasteras, Sweden, 2018, pp 146-150.
13. Insha Altaf, Jawad Ahmad Dar, Firdous ul Rashid, Mohd. Rafiq, "SURVEY ON SELENIUM TOOL IN SOFTWARE TESTING", Proceedings of the 2015 International Conference on Green Computing and Internet of Things (ICGCIoT), Noida, India, 2015, pp 1378-1383.
14. A. Bruns, A. Kornstadt, and D. Wichmann. "Web application tests with Selenium." IEEE Software, 26(5):88–91, 2009.
15. P. Chapman and D. Evans. "Automated black-box detection of sidechannel vulnerabilities in web applications." Proceedings of the 18th ACM Conference on Computer and Communications Security, CCS 2011, Chicago, Illinois, USA, October 17-21, 2011, pp 263–274.
16. S. R. Choudhary, D. Zhao, H. Versee, and A. Orso. "Water: Web application test repair." Proceedings of the First International Workshop on End-to-End Test Script Engineering, 2011, pages 24–29.
17. M. Grechanik, Q. Xie, and C. Fu. "Maintaining and evolving GUI-directed test scripts." 31st International Conference on Software Engineering, ICSE 2009, Vancouver, Canada, May 16-24, 2009, pages 408–418.
18. M. Kowalkiewicz, M. E. Orlowska, T. Kaczmarek, and W. Abramowicz. "Robust web content extraction." Proceedings of the 15th international conference on World Wide Web 2006. pp 887-888.
19. M. Leotta, D. Clerissi, F. Ricca, and C. Spadaro. "Comparing the maintainability of Selenium WebDriver test suites employing different locators: A case study." In Proc. of 1st International Workshop on Joining AcadeMiA and Industry Contributions to testing Automation, JAMAICA 2013 at ISSTA 2013, pages 53–58.
20. Abe, M. and Hori, M.: A visual approach to authoring XPath expressions. Proceedings of Extreme Markup Languages 2001 , pp. 1–14 (2001).
21. Asakawa, C. and Takagi, H.: Transcoding system for non visual Web access (2): annotation-based transcoding. Six-teenth International Conference on Technologies and Persons with Disabilities (CSUN2001) (2001).
22. Brush, A. J., Bargeron, D., Gupta, A., and Cadiz, J. J.: Robust annotation positioning in digital documents. Proceedings of the 2001 ACM Conference on Human Factors in Computing Systems (CHI 2001) , pp. 285–292, Seattle, Washington (2001).
23. Cadiz, J. J., Gupta, A., and Grudin, J.: Using Web annotations for asynchronous collaboration around documents. Proceed ings of ACM 2000 Conference on Computer Supported Coop erative Work (CSCW 2000) , pp. 309–318, Philadelphia, PA (2000).
24. Denoue, L. and Vignollet, L.: An annotation tool for Web browsers and its applications to information retrieval. Pro ceedings of the 6th Conference on Content-Based Multimedia Information Access (RIAO 2000) , Paris, France (2000).
25. Erdmann, M., Maedche, A., Schnurr, H.P., and Staab, S.: From manual to semi-automatic semantic annotation: about ontology-based text annotation tools. Proceedings of the COLING 2000 Workshop on Semantic Annotation and Intel ligent Content, Luxembourg (2000).
26. Heflin, J. and Hendler, J.: Semantic interoperability on the Web. Proceedings of Extreme Markup Languages 2000 , pp. 111–120 (2000).
27. Hori, M., Kondoh, G., Ono, K., Hirose, S., and Singhal, S.: Annotation-based Web content transcoding. Proceedings of the 9th International World Wide Web Conference (WWW9), pp. 197–211, Amsterdam, Netherlands (2000).
28. Hori, M., Ono, K., Kondo, G., and Singhal, S.: Authoring tool for Web content transcoding. Markup Languages: The ory & Practice ,Vol. 2, No. 1, pp. 81–106 (2000).

29. Kahan, J., Kondoh, M.R., Prud'Hommeaux, E., and Swick, R. R.: Annotea: An Open RDF Infrastructure for Shared Web Annotations. Proceedings of the 10th International World Wide Web Conference (WWW10), pp. 623–632, Hong-Kong (2001).
30. Marshall, C. C.: Toward an ecology of hypertext annotation. Proceedings of the 9th ACM Conference on Hypertext and Hypermedia, Pittsburgh, PA, pp.40–49 (1998).
31. Mea, V. D., Beltrami, C. A., Roberto, V., and Brunato, D.: HTML generation and semantic markup for telepathology. Proceedings of the 5th International World Wide Web Con ference (WWW5), pp. 1085–1094, Paris, France (1996).
32. Nagao, K., Shirai, Y., and Kevin, S.: Semantic annotation and transcoding: making Web content more accessible. IEEE Multimedia, Vol. 8, No. 2, pp. 69–81 (2001).
33. Phelps, T. A. and Wilensky, R.: Robust intra-document lo cations. Proceedings of the 9th International World Wide Web Conference (WWW9), pp. 105–118, Amsterdam, Netherlands (2000).
34. Rousseau, F., Macias, J. A., de Lima, J. V., and Duda, A.: User adaptable multimedia presentations for the World Wide Web. Proceedings of the 8th International World Wide Web Conference (WWW8), pp. 195–212, Toronto, Canada (1999).
35. Spinks, R., Topol, B., Seekamp, C., and Ims, S.: Doc ument clipping with annotation. IBM developerWorks, http://www.ibm.com/developerworks/ibm/library/ ibm-clip/ (2001).
36. Qiang Hao, Rui Cai, Yanwei Pang, and Lei Zhang. 2011. From one tree to a forest: a unified solution for structured web data extraction. In International Conference on Research and Development in Information Retrieval.
37. Nitin Jindal and Bing Liu. 2010. A Generalized Tree Matching Algorithm Consid ering Nested Lists for Web Data Extraction. The SIAM International Conference on Data Mining (2010).
38. S. Berner, R. Weber, and R. Keller. Observations and lessons learned from automated testing. In Proceedings of 27th International Confer ence on Software Engineering, ICSE 2005, pages 571–579. IEEE, 2005.
39. S. R. Choudhary, D. Zhao, H. Versee, and A. Orso. WATER: Web application test repair. In Proceedings of 1st International Workshop on End-to-End Test Script Engineering, ETSE 2011, pages 24–29. ACM, 2011.
40. B. Daniel, Q. Luo, M. Mirzaaghaei, D. Dig, D. Marinov, and M. Pezze. Automated GUI refactoring and test script repair. In Proceedings of 1st International Workshop on End-to-End Test Script Engineering, ETSE 2011, pages 38–41. ACM, 2011.

AUTHORS PROFILE

T Kishan Singh, is a student at Department of Computer Science and engineering, Rashtreeya Vidyalaya College of Engineering, Bengaluru, Karnataka, India.

Pavithra H, is working as an assistant professor at Department of Computer Science and engineering, Rashtreeya Vidyalaya College of Engineering, Bengaluru, Karnataka, India.