Comparative Analysis of Functional Test Automation Tools

H.K.D. Prabhashi[1], M.S Shafana[2], M.J. Ahamed Sabani[3]

[1,2,3]*South Eastern University of Sri Lanka*

[1]dilprabhashi@gmail.com , [2]zainashareef@seu.ac.lk , [3]mjasabani@seu.ac.lk

Abstract

The software development life cycle has several stages. Among them, software testing is a continuous process. It begins after the requirement-gathering phase. Today software testing has become a mandatory process. It comes with a variety of challenges also. Although manual testing is an easy task, sometimes it will not be effective due to its incompatibility, lack of coverage, and repetition of test scripts. This study mainly focuses test automation process and its working mechanism, software testing strategies, and different types of software testing tools. For clarification, test automation tools are divided into functional, test management, and load testing tools. The objective is to provide a comparative analysis of test automation tools with similar characteristics such as platform support, the programming language used, and compatibility with web browsers. This research paper also helps the test teams to select the appropriate test tool according to the customer's needs.

Keywords: Software Testing, Test Automation, Test Management, Functional Testing, Load testing

I. INTRODUCTION

Software testing evaluates the performance of software applications to find out if the developed software meets its particular requirements and identifies the shortcomings. Software testing aims to find defects and bugs in the software, system, or product. It also allows us to evaluate the software's quality based on its functionality, usability, efficiency, maintainability, and portability.

We can categorize software testing as manual testing and automation testing. Manual testing, also known as static testing, is performed by testers. In manual testing, the tester acts as an end-user and uses a written test plan to help with the selection of important test cases. There are many drawbacks to manual testing, such as being more time-consuming, less accurate, having no scripting capability, needing much human engagement, and challenging to explore more significant or minor bugs. Automation testing (Polo *et al.*, 2013) has been presented to overcome these errors, which investigates the challenges in manual testing.

Automation testing(Garousi and Elberzhager, 2017) is a software testing technique that uses specialized automated testing software tools to execute a set of test cases. In today's world, most businesses and companies use automated testing tools (Wiklund *et al.*, 2012). Even though testing is a mandatory process in software development, choosing the appropriate tool for testing is a complex task (Bansal, Muli, and Patil, 2013).

The focus of automated testing tools research is collecting evidence that gives developers and testers access to relevant information that can help improve the tool selection efficiency.

II. METHODOLOGY

The testing process plays a significant role in software development, as every small module must be tested to confirm its validity and accuracy. Nowadays, various test automation tools are introduced to improve the efficiency of the testing process.

In searching for the information relevant to this research, indexed journals and conferences have thoroughly published on this topic. Among them, several studies compare different test automation tools. According to our observation, test automation is described in a small number of research articles based on its functionality. This study analyzed test automation tools by categorizing them as test management, load testing, and functional testing tools.

III. DISCUSSION

A. Software Testing Strategies

In the SDLC (Software Development Life Cycle), the testing process runs concurrently with the

software development, and a test strategy can be described as an overview illustrating the test approach (Sei, 2015). The test team, project managers, functional testing team, and software engineers are primarily responsible for developing these techniques. There are four software testing strategies (Sneha and Malle, 2018).

1) Unit Testing

In coding, the programmer performs several tests to determine if the program unit is faulty. Tests are conducted under the white box text approach. Unit tests help determine if each unit in the program operates as required and is error-free. The developers and testers can both save time if a proper unit testing procedure is in place because bugs can be found at the early stage of the testing process (M. A. Umar and C. Zhanfang, 2019). Furthermore, avoiding or limiting the practice of unit testing might raise problems and it is difficult to correct them at a later stage.

2) Integration Testing

Integration testing is often done in parallel with unit testing. Integration testing ensures that the individual modules of the code work properly as a team of QA professionals. Many modern applications run on microservices and self-contained applications designed to handle specific tasks. Integration testing examines whether the software modules are by the specification in the SRS document (Akinsola *et al.*, 2022).

3) System Testing

QA professionals test the software as a complete product with system testing. With this type of functional testing, the tester validates the complete and integrated software packages to ensure that the requirements are satisfied.

4) Acceptance Testing

Acceptance testing is used to ensure that the end-user is satisfied and able to achieve the goals set in the business requirements. Acceptance tests are customer-owned and defined tests that check whether the developed modules fit the acceptance requirements. Acceptance testing is essential to improve stockholders' awareness of the requirements for the product (Nasir, 2021).

B. Test Automation Tools

Automation tools (Polo *et al.*, 2013) in the market can be classified as open-source and payment tools. The tools chosen here are based on their ease of use and availability for testing (Garousi and Elberzhager, 2017).

The following are several categories of software test automation tools.

- Test management tools
- Load testing tools
- *Functional testing tools*

1) Test Management Tools

There are unique ways to deal with test management tools, and they also have different characteristics. Test management tools allow the assignment of test procedures and easy access to data analytics and simple correspondence with various business groups (Chandraprabha, Kumar, and Saxena, 2015). Test management tools are used to store information on how tests should be performed, plan test activities, and report the status of quality assurance activities (Chaudhary, 2017).

i. *TETware*

TETware is a test implementation management system that administers the test, sequences tests, and reports the results in the standard format. This tool supports Unix and 32-bit Microsoft Windows operating systems; it has portable capabilities with enhanced testing opportunities (Abbas, Sultan, and Bhatti, 2017). TETware gives a platform for the testers to work on consistent, decisive test scripts and allows the software system to be handed over quickly.

ii. Test Manager

Test Manager can be defined as an automated software testing tool used for everyday testing activities. This tool uses the java programming language to develop. It simplifies ordinary software development tasks and automates and manages them.

iii. RTH

The meaning of the acronym RTH is "Requirements and Testing Hub". This is an open-source test management tool that also provides troubleshooting facilities.

2) Load Testing Tools

Load Testing is a type of testing that evaluates the system's performance. Automated load testing helps to determine any system's performance and functional problems when it is put under load

(Alferidah and Ahmed, 2020). These tools are applied when software projects near their completion level.

i. Load Tracer

Load Tracer is a widely used web performance tool created by Trace Technology. It is a highly convenient tool the company uses to test web app loading and performance. In addition to testing, viewing, and managing the performance of internet applications, this tool provides a variety of techniques to load on the web servers in the manner that the user prefers.

ii. WebLoad

WebLoad has become another open-source load-testing tool but has some limitations, such as the capacity to produce a large number of concurrent users. It is often used for the web application's stress testing and performance testing. It imitates hundreds and thousands of users for load testing and examines the outcomes of any application to address the web application's flaws and limits. It also works with different types of protocols.

3) Functional Testing Tools

i. Selenium

Selenium (Rishab Jain and Kaluri, 2015)is an open-source and freeware test automation framework for inspecting web browsers on various platforms and browsers. Selenium test scripts are written in programming languages such as JavaScript and Python. Selenium also offers support for many browsers. Chrome, Safari, Firefox, and Internet Explorer are a few of these browsers. It also supports many programming languages and operating systems (Mishra and Atesogullari, 2020). Selenium consists of Selenium IDE, Selenium RC, and Selenium Grid.

Selenium IDE is a selenium testing development environment. It is a Firebox add-on that lets to record, edit, and playtests. Selenium IDE allows saving tests in various formats, including Java, Ruby Script, HTML, and others. Selenium RC is a cross-browser testing solution. It is developed in Java and runs on significant platforms (Ramya, Sindhura, and Sagar, 2017).

One of the drawbacks of the selenium web driver is that it has no built-in functionality for generating screenshots for failed test cases. The selenium web driver cannot generate test results (Rishab Jain and Kaluri, 2015).

ii. Junit

JUnit is a free and open-source framework for regression testing. It primarily uses specific software updates to implement unit tests, help accelerate programming and enhance the quality of the Java code. The key objective of this framework is to make it easier for Java developers to script and run recurring test cases. We can identify this framework as one of the most widely used Java testing frameworks. The JUnit framework (Chaudhary, 2017) is primarily used among specialists to test tiny code. We can perform automated testing of a website by integrating JUnit with the selenium web driver for Java automated testing.

iii. HP UFT

HP UFT is a functional testing tool that is ideally suited for the implementation of regression testing. It is a licensed/commercial tool from HP that is commercially available. It measures the current and projected outcomes and reports the results in implementing summarized information. It is also a simple and effective tool that operates great with Windows and Windows-based applications (Sneha and Malle, 2018). It also supports dynamic testing and saves snapshots on every page visited during activation.

iv. Test Complete

Test complete is a test automation platform developed by SmartBear software. It enables testers to generate automated test scripts for Microsoft Windows, Web, Android (iOS), and iOS applications. Test Complete is a software testing tool that allows for building and automating various types of software testing. Record and replay testing is performed manually by an examiner and enables it to be replayed and maintained as an automated test (Kaur and Kumari, 2011).

v. Ranorex

This is a simple, efficient, and low-cost automated testing tool. It is an enhanced option to other testing tools because it analyzes applications from the user's perspective, utilizing main programming languages and methods such as C#, VB.net, and Iron Python. Commercial software firms and businesses deploy it all around the world. Future work for Ranorex (Kakaraparthy, 2017) includes the development of an open, well-documented platform that enables customers to create their

plug-ins and provide their applications with full recognition.

C. *Comparative Analysis of Functional Test Automation Tools*

The above table analyzes the different kinds of test automation tools based on various characteristics. By seeing Table 01, we can identify the primary key factors we have to consider when selecting the test automation tool. Here we have only considered some of the most commonly used test automation tools available in the market (Chaudhary, 2017).

IV. CONCLUSION

Testing plays an essential part in the development of any software product. Multiple test automation tools have been proposed and are currently available on the market today. Researching various software tools to compete with other software developers and offer the highest quality software product becomes a critical challenge for the team. The drawbacks of manual testing compared to automation testing and some of the most important and frequently used software automation tools used in different platforms have been reviewed in this article.

When selecting test automation tools, it is important to consider ease of use, cost, and supporting platforms. Moreover, before deciding to automate testing, be sure to have skilled workers to work with automation tools.

Table 01: Comparison of Functional Testing tools

Automation Tool	**Developed By**	**Supported Platforms**	**Browser compatibility**	**Programming skills required**	**Cost**	**Scripting Languages**
Selenium	Jason Huggins	Cross Platforms	Cross-browser	Needed	Open-source	Java, Ruby, Python, PhP, C#, .net
JUnit	Kent Beck, Erich Gamma, David Saff	Cross-Platform	Cross-browser	Needed	Open-source	Java
HP UFT	MicroFocus	Windows	Google Chrome, Internet Explorer, Firebox	Not required (only needed to test advanced test scripts.)	Commercial	VBScript
TestComplete	SmartBear	Windows	Chrome, Firebox, Opera, IE	Needed	Commercial	VBScript, C# ,js JScript++
Ranorex	Ranorex GmbH	Windows	Chrome, Firebox, Opera, IE, Netscape	Partial	Commercial	VBScript supports .Net, C++, C#

REFERENCES

Abbas, R., Sultan, Z. and Bhatti, S. N. (2017) 'Comparative analysis of automated load testing tools: Apache JMeter, Microsoft Visual Studio (TFS), LoadRunner, Siege', *in International Conference on Communication Technologies, ComTech 2017*. IEEE, pp. 39–44. doi: 10.1109/COMTECH.2017.8065747.

Akinsola, J. E. T. et al. (2022) 'Qualitative Comparative Analysis of Software Integration Testing Techniques', *Journal of Science and Logics in ICT Research (UIJSLICTR)*, 7(2), pp. 67–82.

Alferidah, S. K. and Ahmed, S. (2020) 'Automated Software Testing Tools', *2020 International Conference on Computing and Information Technology, ICCIT 2020*, pp. 183–186. doi: 10.1109/ICCIT-144147971.2020.9213735.

Bansal, A., Muli, M. and Patil, K. (2013) 'Taming complexity while gaining efficiency: Requirements for the next generation of test automation tools', *AUTOTESTCON (Proceedings),* pp. 123–128. doi: 10.1109/AUTEST.2013.6645055.

Chandraprabha, C., Kumar, A. and Saxena, S. (2015) 'Data Driven Testing Framework using Selenium WebDriver', *International Journal of Computer Applications,* 118(18), pp. 18–23. doi: 10.5120/20845-3497.

Chaudhary, S. (2017) 'Latest Software Testing Tools and Techniques: A Review', *International Journal of Advanced Research in Computer Science and Software*

Engineering, 7(5), pp. 538–540. doi: 10.23956/ijarcsse/sv7i5/0138.

Garousi, V. and Elberzhager, F. (2017) 'Test Automation: Not Just for Test Execution', *IEEE Software*, 34(2), pp. 90–96. doi: 10.1109/MS.2017.34.

KAKARAPARTHY, D. (2017) 'Overview and Analysis of Automated Testing Tools: Ranorex, Test Complete, Selenium', *International Research Journal of Engineering and Technology*, 04(10), pp. 1575–1579.

Kaur, M. and Kumari, R. (2011) 'Comparative Study of Automated Testing Tools: TestComplete and QuickTest Pro', *International Journal of Computer Applications*, 24(1), pp. 1–7. doi: 10.5120/2918-3844.

M. A. Umar and C. Zhanfang (2019) 'A Study of Automated Software Testing: Automation Tools and Frameworks', *International Journal of Computer Science Engineering (IJCSE)*.

Mishra, A. and Atesogullari, D. (2020) 'Automation Testing Tools : a Comparative View Automation Testing Tools : a Comparative', *International Journal on Information and Computer Security*, 12(December), pp. 63–76.

Nasir, N. (2021) 'Acceptance Testing in Agile Software Development Perspectives from Research and Practice', *The IPSI BgD Transactions on Internet Research*.

Polo, M. et al. (2013) 'Test automation', *IEEE Software,* 30(1), pp. 84–89. doi: 10.1109/MS.2013.15.

Ramya, P., Sindhura, V. and Sagar, P. V. (2017) 'Testing using selenium web driver', *in Proceedings of the 2017 2nd IEEE International Conference on Electrical, Computer and Communication Technologies, ICECCT 2017*. IEEE, pp. 1–7. doi: 10.1109/ICECCT.2017.8117878.

Rishab Jain, C. and Kaluri, R. (2015) 'Design of automation scripts execution application for selenium web driver and testng framework', *ARPN Journal of Engineering and Applied Sciences,* 10(6), pp. 2440–2445.

Sei, L. M. (2015) 'Automating Test Activities: Test Cases Creation, Test Execution, and Test Reporting with Multiple Test Automation Tools', *International Journal of Computer and Information Engineering,* 9(10), pp. 2213–2216.

Sneha, K. and Malle, G. M. (2018) 'Research on software testing techniques and software automation testing tools, *in 2017 International Conference on Energy, Communication, Data Analytics, and Soft Computing, ICECDS 2017*. IEEE, pp. 77–81. doi: 10.1109/ICECDS.2017.8389562.

Wiklund, K. et al. (2012) 'Technical debt in test automation', *Proceedings - IEEE 5th International Conference on Software Testing, Verification and Validation, ICST 2012*, pp. 887–892. doi: 10.1109/ICST.2012.192.