Automated Testing in Java-Comparative Analysis of Automated Testing Tools

Hareesh Kumar Rapolu

hareeshkumar.rapolu@gmail.com

Abstract

The following research project has signified with a detailed analysis of several automated testing tools which are being constructed specifically for Java applications. It has risen with the rising complexity of software systems in which effective testing has been considered for determining quality and reliability. Moreover, the combination of testing tools such as TestNG and JUnit has paved the path for supporting the community. Furthermore, the utilisation of valid recommendations such as TestNG for integration testing and Selenium for UI testing has proved to be suitable for specific testing needs. Therefore, this has advantaged the developers and the users to choose the most appropriate testing tools.

Keywords: Automated testing, Java, Testing Tools, Comparative Analysis

I. INTRODUCTION

This research project will provide a nuanced understanding of automated testing in Java-comparative analysis of automated testing tools. In the dynamic infrastructure of software testing, the role of test automation will be considered to be of paramount importance as it will render it becoming increasingly significant. At the same time, this will aid in conducting a comparative analysis of several test automation frameworks that will empower and make justified decisions in the field of testing endeavours. The use of Java which is identified as one of the most widely used programming languages will cater for the utilisation of automated testing tools to benefit the developers to make sure that their application platforms are intended. Furthermore, the report will incorporate the importance of automated testing in Java and shedding with a curated understanding of performance metrics. This will demonstrate a comparative analysis of tools and will propose recommendations for automated testing tools.

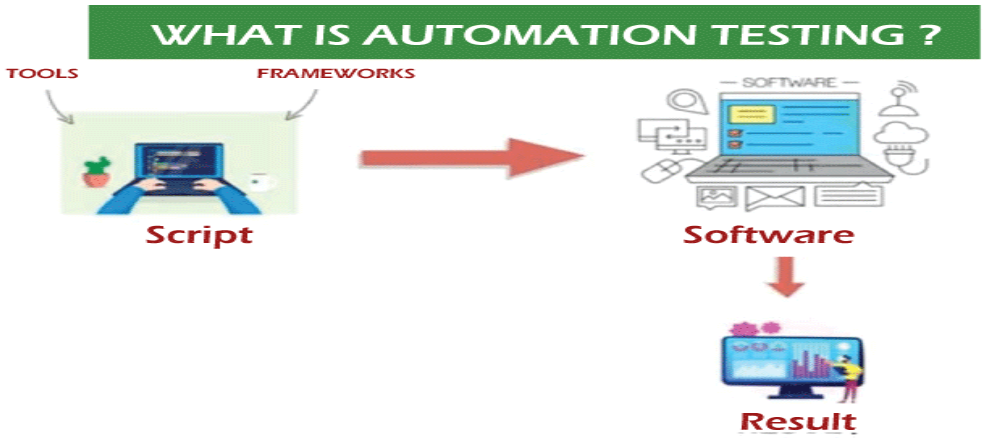

Figure 1: Demonstrating Test Automation

II. STATING THE OVERVIEW OF TESTING TOOLS

This section states that testing tools are specialised software frameworks which enable the developers to get complete access to the functionality of their applications by creating a simulated user environment. At the same time, this also seeks to grant automated testing of several characteristics and functionalities with minimal human intervention in Java. It consists of testing tools such as JUnit, TestNG and Selenium are the reputed ones. The first testing tool, JUnit is a widely used framework that is capable of unit testing in Java. This is done by providing the users to get access to design repeatable tests in a simpler form thereby making it foundational for Test-Driven Development (TDD)[1]. Additionally, this consists of key features such as annotations along with assertions and test runners and is regarded to be lightweight and foster robust community support. Another testing tool which is termed to be of immense relevance refers to TestNG. It has unique features such as parallel execution with flexible test configurations which allows data-driven testing. This tends to improve the possibilities of testing features. However, this is attained by supporting BDD which is widely used in Java-comparative analysis of automation testing tools.

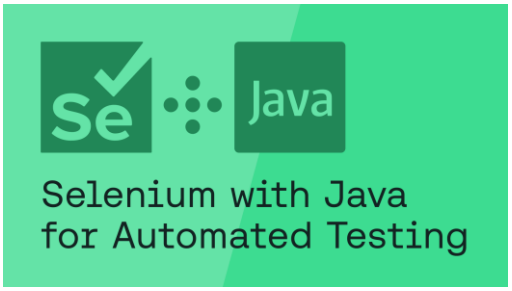

Figure 2: Selenium as a Reputed Testing Tool

III. DISCUSSING THE IMPORTANCE OF AUTOMATED TESTING IN JAVA

The following section illustrates that automated testing in Java-comparative analysis is beneficial as it has the probability to develop the chances of efficiency and quality of software development. This is achieved by enabling valuable feedback loops along with covering a wider test range and the detection of bugs at the early stages[2]. This leads to faster time-to-market and thus minimises the costs specifically in terms of dealing with critical Java applications regarding automated testing tools. Additionally, the integration of test automation with Java and Selenium fosters enormous benefits ranging from seamless amalgamation of CI/CD pipelines to cross-platform compatibility. This amalgamation of core practices such as modulated test configurations and stringent reporting aids in constructing dependable and scalable automation frameworks. Furthermore, automation testing in Java also allows for a broader range of scenarios such as edge cases and regression tests that are critical for covering manually[3]. This seems to maintain consistency in accurate and efficient steps in a precise form thereby lowering the chances of mistakes committed by humans and rendering fruitful results progressively.

IV. UNDERSTANDING THE PERFORMANCE METRICS

This section discusses that performance metrics stand to be a necessary aspect in depicting the overall effectiveness of automated testing tools. It is obtained by the involvement of test execution speed followed by build stability and scalability.

Test Execution Speed: This performance metric is termed to be essential as it provides precious insights into the efficiency of the testing process. It indicates that faster execution means swifter feedback for the developers for the facilitation of iterative development[4].

Build Stability: Build stability is also considered to be another important performance metric because it measures the frequency of building failure. Now, this seems to highlight the overall stability of the software during the development process of the testing tools[5].

Scalability: Scalability is termed to be another vital performance metric as it poses the potential of testing tools to manage a rising number of test cases. This replicates that the testing tools can get familiarised with emerging preferences of software applications.

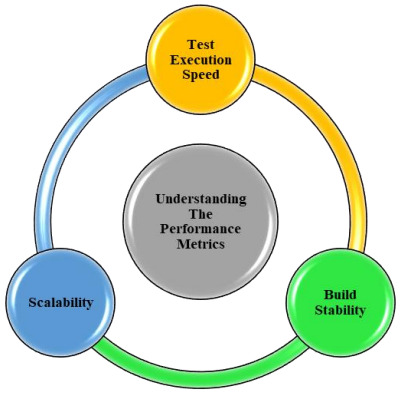

Figure 3: Highlighting the Performance Metrics

V. ELUCIDATING WITH RECOMMENDATIONS FOR AUTOMATED TESTING TOOLS

The following section elucidates measurable recommendations which are used in a sophisticated fashion while implementing automated testing tools.

Utilising TestNG for Integration Testing: TestNG is broadly utilised for integration testing. This is because of its powerful features such as dependencies with test grouping and data-driven capabilities. It also involves detailed reporting and flexible annotations that result in granting for the effective testing of critical communications among different components within the system[6]. As a result, this makes it an appropriate selection for the optimum management and thus organising large data sets of test suites.

Examining Selenium for UI Testing: It is observed that Selenium is necessary for UI testing as it fosters an open-source platform that tends to support several browsers. It allows the development of test scripts in various programming languages which makes it easier to implement[7]. As a result, this gets combined with other testing frameworks making it suitable for a varying range of testing scenarios in different platforms.

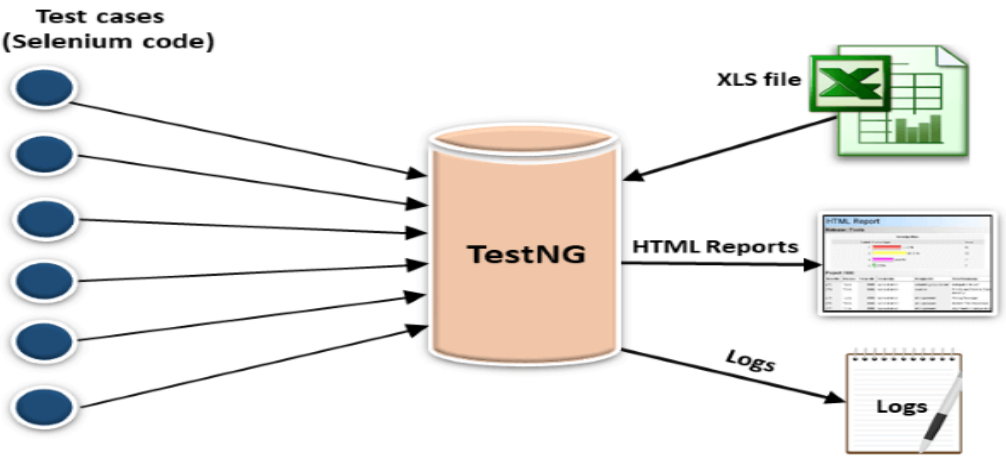

Figure 4: Elucidating Automated Testing Tools

VI. CONCLUSION

This research project has explained that automated testing is a necessary component in the Java-comparative analysis of automated testing tools. It has used testing tools such as JUnit with TestNG and Selenium for the enhancement of quality while getting flexible and staying familiarised with the testing uniqueness. This has enabled fast feedback and wider test coverage. Furthermore, the emphasis on the performance metrics has highlighted the overall effectiveness of these tools. This has proved to be reliable thus facilitating successful software delivery and getting adhered with modern development facilities. Therefore, this has facilitated effective test script development along with execution and maintenance within the automated testing tools.

Abbreviations and Acronyms

- JUnit- Java Unit Testing
- TestNG- Test Next Generation
- CI/CD-Continuous Integration or Continuous Deployment
- BDD- Behaviour Driven Development
- TDD- Test Driven Development
- UI- User Interface

Units

- Time is measured in seconds
- Length is calculated in meters
- Force is measured in Newton

Equations

- Test Coverage (%) = [{Number of Executed Statements / Total Statements} X100]

REFERENCES

[1] E. Mashhadi and H. Hemmati. “"Applying codebert for automated program repair of java simple bugs." *In 2021 IEEE/ACM 18th International Conference on Mining Software Repositories (MSR)*,pp.505-509.IEEE,Mar.2021,arXiv.orge-Printarchive.Available: https://arxiv.org/pdf/2103.11626

[2] F. Dalton, M. Ribeiro, G. Pinto, L. Fernandes, R. Gheyi, and B. Fonseca. “"Is exceptional behavior testing an exception? an empirical assessment using java automated tests."”, *In Proceedings of the 24th International Conference on Evaluation and Assessment in Software Engineering*,pp.170-179.,Apr.2020,Home-GustavoPinto.Available: http://gustavopinto.org/lost+found/ease2020.pdf
[3] K. Pei, Y. Cao, J. Yang, and S. Jana, “DeepXplore,” *Communications of the ACM*, vol. 62, no. 11, pp. 137–145, Oct. 2017, doi: https://doi.org/10.1145/3361566.
[4] K. Liu *et al.* “"On the efficiency of test suite based program repair: A systematic assessment of 16 automated repair systems for java programs."”, *In Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*, pp. 615-627., Jun. 2020, arXiv.org e-Print archive. Available: https://arxiv.org/pdf/2008.00914
[5] P. Kong, L. Li, J. Gao, K. Liu, T. F. Bissyande, and J. Klein, “Automated Testing of Android Apps: A Systematic Literature Review,” *IEEE Transactions on Reliability*, vol. 68, no. 1, pp. 45–66, Mar. 2019, doi: https://doi.org/10.1109/tr.2018.2865733.
[6] V. Garousi and M. V. Mäntylä, “When and what to automate in software testing? A multi-vocal literature review,” *Information and Software Technology*, vol. 76, pp. 92–117, Aug. 2016, doi: https://doi.org/10. a1016/j.infsof.2016.04.015.
[7] Y. Tian, K. Pei, S. Jana, and B. Ray, “DeepTest: Automated Testing of Deep-Neural-Network-driven Autonomous Cars,” *In Proceedings of the 40th international conference on software engineering*, vol. 12, May 2018, doi: https://doi.org/10.1145 /3180155.3180220.