Test Coverage Enhancement using Hybrid Modular Approach

[1]Roopesh R, [2]Dr. Manimala S

[1]Student, [2]Assistant Professor
[1]Department of Computer Science and Engineering,
[1]JSSSTU, Mysuru, Karnataka, India

Abstract : Automation testing is a process that compares the actual results to the desired outcome in the software application. Automation testing is one of the most effective ways to test web- based applications since it allows you to cover more scenarios in less time. Selenium is one of the most popular and effective automation frameworks that has been widely used for automating the testing of applications. The modular testing framework is one of the approaches that is used for automation testing in which the test scripts are written by splitting down the entire application into smaller, independent modules. Multiple data sets cannot be used due to hardcoded data in the test scripts. In this paper, a data driven approach is proposed, where test data and test script are split to support multiple datasets and better coverage. Further, this split process used across multiple modules for better test coverage.

***Index Terms* - Selenium, Automation, Modular, Data-Driven, Keyword Driven.**

I. INTRODUCTION

Automation testing is one of the software testing techniques that is used to check the quality of the software application using a set of tools and Automation Testing Frameworks (ATFs). Selenium is one of the most widely used automation frame- work in the world because of its open-source availability, flexibility, cross browser and cross platform support, re-usability and integration and also it is easy to implement [1].

Selenium framework has a bunch of tools which are required by the organizations.

- Selenium RC - It is the first automation tool that was de- signed. because of complex architecture and less object- oriented APIs, it led to the development of Selenium WebDriver. Now Selenium RC is deprecated.
- Selenium IDE - An IDE which is specifically designed for selenium tests.
- Selenium Grid - used for executing multiple tests in cross browsers and cross platforms in parallel
- Selenium WebDriver - Used for web-based software application testing.
-

Selenium RC and Selenium WebDriver are merged into one calling it as Selenium 2. [2].

In most modern online applications, a significant number of users send several requests at the same time is very huge. To guarantee that a device can handle the number of concurrent requests that it was designed to handle, it must be tested with extensive load. Load testing is performed to determine how an application performs under various load situations. Failure to do a load test might have devastating consequences. A typical web application of an organization comprises of huge number of modules. Delivering error-free application is very challenging. Performing testing of such kind of application is nothing less than a tedious job.

Modular approach of testing promises of testing huge ap- plication can be made easily by splitting of application into smaller modules and testing of each module separately and computing the results, but here the test data is hardcoded. For every test data, we need to modify the code. Data driven approach is another way of testing web applications where we split the test data from the test script logic to load the multiple datasets for better coverage.

The infusion of keyword-data driven(hybrid) approach into modular approach helps in more test coverage which results in good product quality.

II. LITERATURE SURVEY

For load test execution, the asset utilization of Selenium- based load tests in various designs were read [3]. The most important findings are that headless browsers use far fewer resources than other types of browsers. Similarly, by sharing instances amongst client cases, a load driver's limit may be increased by at least 20%.

Software testing is shown to be the most essential requirement in an SDLC [4]. The key objective of testing methods is to compare the obtained results to those that the end-user is aware of. This focuses on utilizing Selenium web driver to test an application and demonstrating its use in conjunction with other devices such as TestNG, Maven, and so on, enabling ever simpler ways to deal with enhancing the nature of testing.

The pros and cons of adopting Git are addressed here. Git provides a number of commands that make it much easier for developers to deal with and use the repository [5]. It enables developers to work more quickly. Git allows developers to branch more often in order to avoid disrupting the master code. The developer has complete control over the branch that has been set up locally. If the merging method is not rigorously supervised, these actions might lose or harm information about changes in the history of the code, which is necessary for ongoing data development.

Software testing is the process of testing the software with the objective of finding flaws and completing software with no defects [6]. As a result, a variety of testing methods have been employed over time. Various tools also have been developed to automate the testing. One of the automation tools is Selenium which is predominantly used for certifying the quality of the product.

Let us see in brief about the different testing approaches we are addressing in this paper.

A. Module Based Testing Framework

Abstraction is one of the most well-known OOPs principles, and the Module-based Testing Framework is built on it. The framework breaks down the whole" Application Under Test" into logical and discrete components. We write a distinct and independent test script for each module. As a result, when several test scripts are combined, a bigger test script representing many modules is created. The major disadvantage of this framework is that test data is hard-coded and because of this, for every data, the test script needs to be rewritten every time, which is an arduous job for bigger applications [7]. Figure 1 shows the high-level working model of module-based testing framework works.

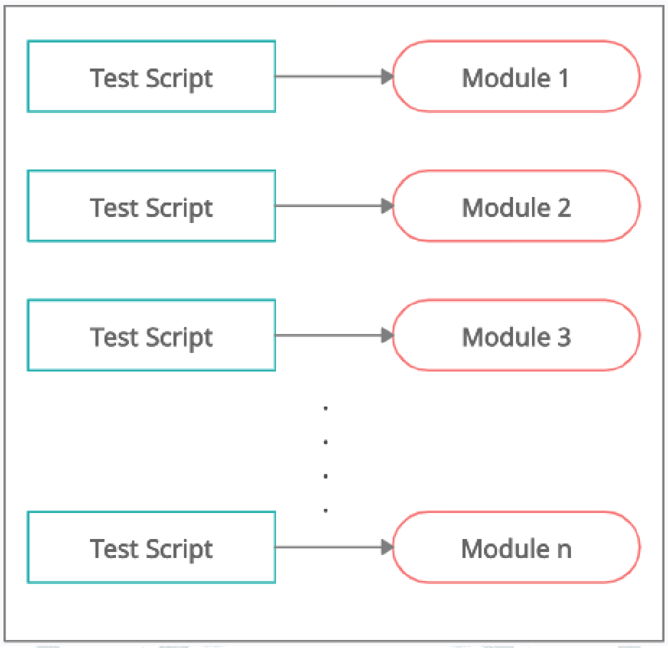
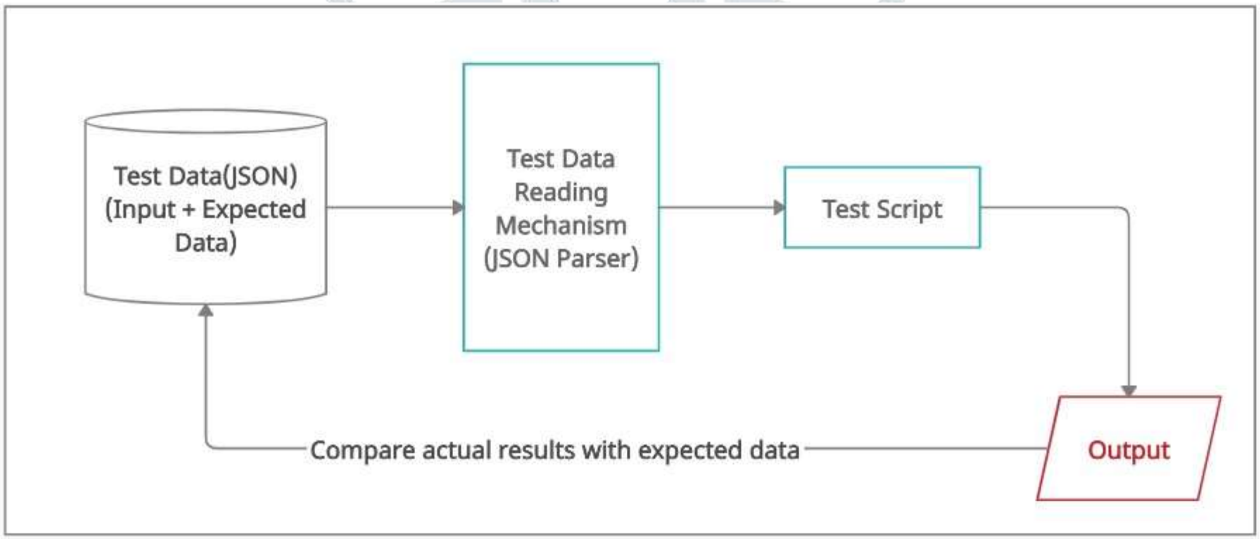


Fig. 1. Module Based Testing Framework

B. Data Driven Testing Framework

Data-Driven Framework splits test script logic and test data to provide flexibility on choosing multiple sets of test data for better coverage. In this approach, the test data is processed through test data reading mechanism (XML reader, JSON parser, etc.,) and then to test script logic. The outcome of the execution is validated against expected results, so that, the feature or a module can be tested in multiple ways to improve the quality of the application better [8]. Figure 2 shows the high-level working model of data driven test framework. Many data files are required for each test scenario. We may have a lot of data inputs and need a lot of verification depending on how many screens are viewed. As a result, the data files in the test case must be kept in distinct folders [9].



Test Data(JSON) (Input + Expected Data)
Test Data Reading Mechanism (JSON Parser)
Test Script
Output
Compare actual results with expected data

Fig. 2. Data Driven Framework

C. Keyword Driven Testing Framework

Test data and test logic are integrated with the keyword-driven test script. The data keyword is analogous to data-driven technology in that it is relatively basic. In a test script, the keywords refer to test data. We simply need to substitute the real test data with the key phrase in this framework during the script run. The major drawback of keyword driven approach is keyword management, just by looking into the keyword and without the test data it is difficult to write the test cases and also, it requires substantial knowledge of scripting skills to write the test script.

D. Hybrid Framework (Keyword-Data Driven Framework)

Hybrid Framework is nothing but combining the test frame- works available to overcome the disadvantages of one another. Keyword-Data Driven Framework is a well-known hybrid framework, as the name suggests here both keyword as well as it's associated test data is available to push into the test script. Figure 3 represents the high-level working model of the Hybrid Framework

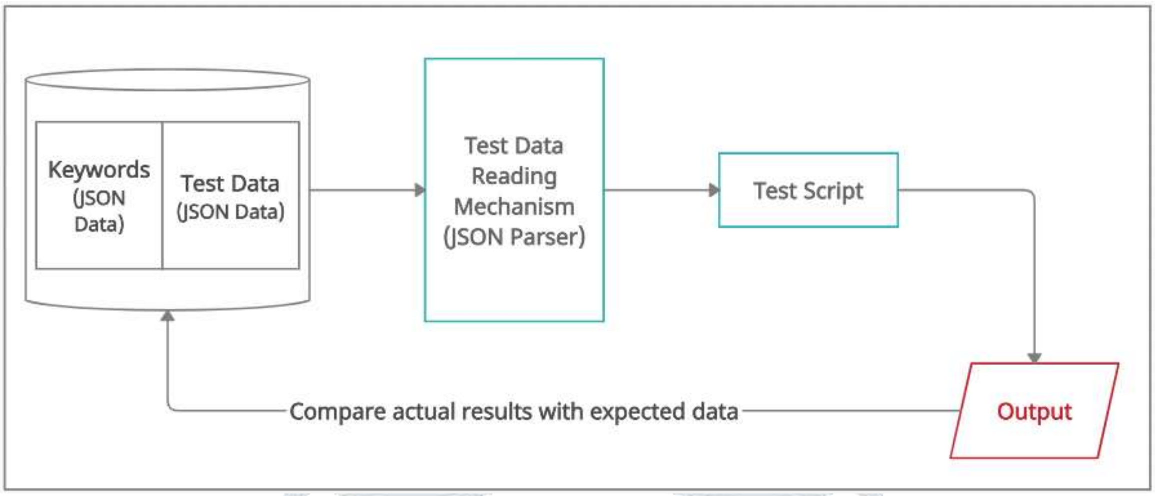

Keywords (JSON Data)
Test Data (JSON Data)
Test Data Reading Mechanism (JSON Parser)
Test Script
Output
Compare actual results with expected data

Fig. 3. Hybrid Testing Framework

III. PROPOSED WORK

Through this paper, the inclusion of a Hybrid testing framework (Keyword-Data Driven Framework) into the module-based testing framework to improve overall testing efficiency is proposed. Here, the same is achieved by replacing the hardcoded test data with a JSON parser loading it with JSON files designed for that particular module. Figure 4 represents the high-level working model of the proposed approach.

The below steps are followed for the implementation of the proposed model:

1) Creation of POM page
2) Importing Java Dependencies
3) Inheriting the required methods from ATF (Automation Test Framework)
4) Azure Logging
5) Execution of Testcases
6) Analysing the reports

Figure 5 shows the flow diagram of the how the current approach is implemented.

1. *Creation of POM page*

The Project Object Model (POM) is the most important file in any Java project since it provides all of the data or information about the dependencies that are necessary to run the Java code. A POM is the primary unit of work in Maven. The project dependencies, plugins or objectives that may be performed, build profiles, and so on are examples of configurations that can be provided in the POM. Other details, such as the project's version, description, developers, mailing lists, and so on, can also be included. [10].

Similarly for every test suite a pom.xml file necessary for holding the configuration details of the test environment

2. Importing Java Dependencies

To import the dependencies, we use Apache Maven. Maven's dependency management is a key feature. It's simple to manage dependencies for a single project. It is feasible to manage dependencies for multi-module projects and apps with hundreds of modules. Maven aids with the definition, creation, and maintenance of repeatable build with well-defined class- paths and library versions [11].

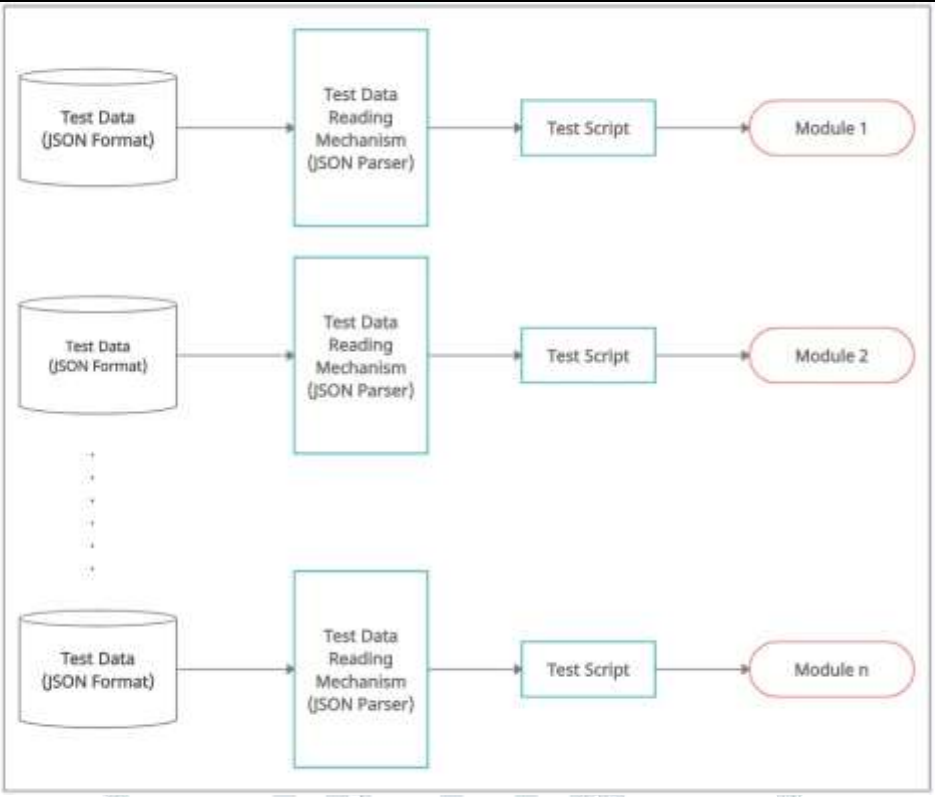


Fig. 4. Proposed Hybrid Modular Framework

3. Inheriting the required methods from the ATF

There are certain methods in ATF that are mandatory to run any test suite. Few methods has to run before the execution of testcase that helps to create the suitable environment and few methods should run post execution of the testcase to decide what should happen after the execution completion depending on the outcome.

4. Azure Logging and S3 Report Generation

Microsoft Azure is used for logging the results of the testcases. Amazon S3 is used for storing the results and generating the reports.

5. Execution of Testcases

1) Choosing the data file format and Creation of data parser: JSON format is chosen over Excel format as it is a light-weight, less compact file which stores the data in the hierarchical manner. Excel is ideal for importing tiny spreadsheet files with a lot of structure. It performs poorly when loading files with 10,000 rows and 100+ columns, with some of these columns supplied by unstructured content such as reviews or descriptions. Excel, it turns out, does not follow CSV-formatting rules, so even if we encoded all of the characters correctly, Excel fails to read them [12].

JSON is the actual standard for exchanging data between web servers, browsers, and mobile apps. Its straightforward form and adaptability make it simple to read and comprehend, as well as manipulate in the programming language of your choice.

JSON Parser is the test data reading mechanism used to process the JSON data. We parse the test data through JSON parser so that, it can be fetched by the test script [13].

2) Creation of Testscripts*:* TestNG is an automated testing framework. JUnit, which utilizes annotations (@), is the in- spiration for TestNG. TestNG addresses JUnit’s shortcomings and is meant to make end-to-end testing simple. TestNG helps in finding out how many test cases are succeeded, failed, or skipped by creating a suitable report. Failed test cases can run on their own [14].

We use TestNG dataprovider to fetch the data from the JSON data parser and use them as the test data in module-based testing framework instead of hard coded data. Test scripts are created based on the scenarios that are designed. In this way the multiple test data is fed into the modular testing framework.

3) Execution of Testcases*:* Testcases are nothing but the combination of test data and test scripts. Testcases are executed based on the above steps and the results are computed.

6. Analyzing the reports

The Generated Reports are obtained analyzed to certify the quality of the application.

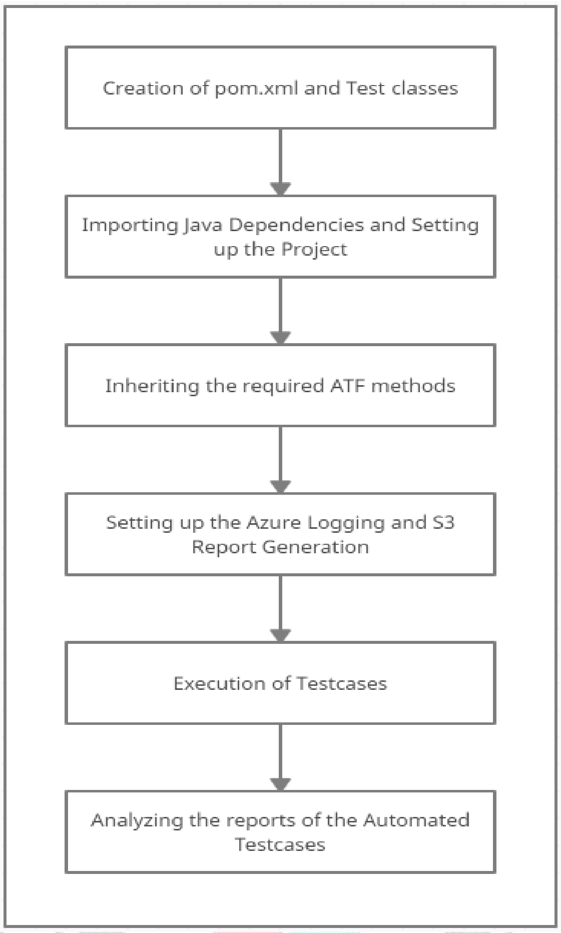

Creation of pom.xml and Test classes
Importing Java Dependencies and Setting up the Project
Inheriting the required ATF methods
Setting up the Azure Logging and S3 Report Generation
Execution of Testcases
Analyzing the reports of the Automated Testcases

Fig. 5. Implementation Methodology

IV. RESULTS AND DISCUSSION

The proposed methodology is implemented for approx. 40 modules and a drastic improvement is observed in resultsw.r.t to the time taken to execute each module and also the coverage percentage of each module so that, the application quality is certified with more accurate data. Table 4.1 shows the comparison of testing methods for few of the different modules.

A Manual testing pass percentage will be little higher than automation testing as the testing is done with very little data than the automation testing, the failure in manual testing was percentage was approx. 20%. In Modular approach we were able to find out few testcase fails and it was little higher manual testing i.e., approx. 40%. In Hybrid Modular Approach the failure percentage is very higher than the modular approach which is approximated between 70% - 80%.

Table 4.1: Results obtained for few of the Modules

Modules	No. of TCs	Manual QE	Modular Approach	Proposed Approach
Project	30	7 days	8 hours	4 hours
Budget	30	5 days	5 hours	2 hours
Bid	30	6 days	7 hours	4 hours
Master Contracts	18	6 days	8 hours	3 hours
Contracts	10	3 days	4 hours	1 hour
Work Orders	35	5 days	3 hours	2 hours
Library	80	10 days	12 hours	7 hours

IV. ACKNOWLEDGMENT

I express my deep gratitude to my guide and professor Dr. Manimala S, assistant professor at JSS Science and Technology University, Mysuru, for providing continuous suggestions extended to me with immense care and zeal throughout my work. I also Thank Mr. Sumit Batabyal, SDET Lead, Aurigo Technologies, Bangalore for providing opportunity to me to implement the methodology that is proposed in this paper.

REFERENCES

[1] J. Kaur, "Reasons to use selenium for automation testing," 2019. [Online]. Available: https://dzone.com/articles/11-reasons-why-go-for- automation-testing-using-sel

[2] P. Ramya, V. Sindhura, and P. Sagar, "Testing using selenium web driver," *2017 Second International Conference on Electrical, Computer and Communication Technologies (ICECCT)*, pp. 1–7, 2017.

[3] S. M. Shariff, H. Li, C.-P. Bezemer, A. E. Hassan, T. H. Nguyen, and P. Flora, "Improving the testing efficiency of selenium-based load tests," in *2019 IEEE/ACM 14th International Workshop on Automation of Software Test (AST)*, 2019, pp. 14–20**.**

[4] M. A. Umar and Z. Chen, "A study of automated software testing: Automation tools and frameworks," vol. 8, pp. 217–225, 12 2019.

[5] S. Just, K. Herzig, J. Czerwonka, and B. Murphy, "Switching to git: The good, the bad, and the ugly," in *2016 IEEE 27th International Symposium on Software Reliability Engineering (ISSRE)*, 2016, pp. 400– 411.

[6] J. Gaur, A. Goyal, T. Choudhury, and S. Sabitha, "A walk through of software testing techniques," 01 2016, pp. 103–108.

[7] K. Aebersold, "Test automation frameworks," 2021. [On- line]. Available: https://smartbear.com/learn/automated-testing/test- automation-frameworks/

[8] A. Chandraprabha, Sajal Saxena and Kumar, "Data driven testing frame- work using selenium webdriver," *International Journal of Computer Applications*, vol. 118, pp. 18–23, 05 2015.

[9] Y. Gupta, "Pros & cons of data-driven versus keyword-driven automation frameworks." [Online]. Avail- able: https://www.softwaretestinggenius.com/pros-cons-of-data-driven- versus-keyword-driven-automation-frameworks/

[10] X. Zhen-hai and Y. Yong-zhi, "Automatic updating method based on maven," in *2014 9th International Conference on Computer Science Education*, 2014, pp. 1074–1077.

[11] T. A. S. Foundation, "Apache maven dependency plugin." [Online]. Available: https://maven.apache.org/plugins/maven-dependency-plugin/

[12] Datainfiniti, "4 reasons you should use json instead of csv," 2013. **[**Online]. Available: https://blog.datafiniti.co/4-reasons-you-should-use- json-instead-of-csv-2cac362f1943

[13] J. Freeman, "What is json? a better format for data exchange," 2019. [Online]. Available: https://www.infoworld.com/article/ 3222851/ what- is-json-a-better-format-for-data-exchange.html

[14] C. Beust, "Testng," 2019. [Online]. Available: https://testng.org/doc/