To Design and Implement a Flexible, Reusable and Maintainable Selenium Automation Framework

Deepthi Wilson R
PG Student
Dept. Computer Science & Engineering
GSSSIETW, Mysuru, India

Manjuprasad B
Assistant Professor
Dept. Computer Science & Engineering
GSSSIETW, Mysuru, India

Abstract: **Test automation is a field of research around for a very long time. Still, testing and related concepts are considered to be one of the most time-consuming and expensive parts of the software life cycle. Although it is a field with a relatively long research background, many existing test automation systems are still relatively simple and not very different from the early days. They still focus on executing an existing, usually manually crafted, set of tests over and over again. In this Survey, we are exploring about Automation testing. Automation testing is useful to achieve and maintain quality of software's. And for this purpose, detailed planning is required. There are different approaches to implement automation testing. There are number of automation tools that we can use as per our requirements. Here we are using selenium automation tool for the project and implementing a comprehensive selenium framework.**

***Keywords: Selenium, Automation Testing Tools, Automated testing, Test Automation, Testing Framework*.**

I. INTRODUCTION

Software testing is most important phase in software development life cycle to ensure that our product is up to the requirement. If testing is not done properly them all other phases right from requirement review to implementation will be useless. Software testing mainly has two types: manual testing and automation testing. Testing software manually means without using any script or automated tools. Here, software tester plays role of user and looks software from user's perspective to find the defects in system. In, automation testing is process in which testing activities are done through some script. Here, tester uses another software test product.

Today most of the companies are implementing automation testing for ensuring quality of their product from this only we can understand the importance of automation tools in software engineering. Creating automated test cases will cause initial development overhead but compared to manual execution, overall execution time will be decreased for each test case regression. [3] Test automation is not the same as testing, and the skills needed to be a good automation tester, are not the same skills needed to be a good tester. [11]

For each test activities, we are having different tools still we cannot make testing fully automatic, we still need some manual testing. Automation testing has different approaches and methodologies of implementation. The testing methodologies can be used together or individually. We must select according to our requirements of testing because these methodologies provide different resources for different tasks. In this report, approaches and methodologies are explained further. The main objective of this paper is to study automation testing in detail.

II. TEST AUTOMATION

Today's world is full of competition and speed. As technology is becoming a strong weapon to acquire market place for product. Along with this quality and cost of product are becoming most important factor to stay in market. This is the reason why companies' selects automation testing for their product. Automation process has many advantages like efficiency, cost effectiveness and coverage. But automation testing cannot eliminate all the bugs from system so in some extent manual testing is necessary. According to "Douglas Hoffman" test case is complete when all the following elements are present [12]:

- Ability to run two or more specified test cases
- Ability to run a subset of all the automated test cases
- No intervention is needed after launching the tests
- Automatically sets-up and/or records the relevant test environment parameters
- Runs the test cases
- Captures the relevant results
- Compares actual with expected results and flags differences
- Analyses and reports pass/fail for each test case and for the test run

Need of Test Automation

- Companies not only want to test software adequately, but also as quickly and thoroughly as possible. To accomplish this goal, organizations are turning to automated testing.
- To increase the test coverage
- Reduces the need for manual testing and discovers defects manual testing cannot expose and also

manual testing is error prone and a time-consuming process.

- Running the tests again and again gives us the confidence that the new work we added to the system did not break the code that used to work and to make sure that the changes we introduced are working.
- Executing the tests can also help us understand what portion of the desired functionality has been implemented.
- The set of the automated test suite can form a regression test suite. The purpose of the regression suite is to make sure that the software behavior is unchanged unless due to data change or latest software.
- Automating also reduces the time taken for regression testing.
- Automated unit test suite helps find the problems at an earlier stage and solve them.

III. TEST AUTOMATION USING TESTING FRAMEWORK

A testing framework or more specifically a testing automation framework is an execution environment for automated tests. It is the overall system in which the tests will be automated. It is defined as the set of assumptions, concepts, and practices that constitute a work platform or support for automated testing. The Testing framework is responsible for:

- Defining the format in which to express expectations.
- Creating a mechanism to hook into or drive the application under test
- Executing the tests
- Reporting results

If we have a group of testers and suppose if each project implements a unique strategy, then the time needed for the tester become productive in the new environment will take long. To handle this, we cannot make changes to the automation environment for each new application that comes along. For this purpose, we use a testing framework that is application independent and has the capability to expand with the requirements of each application. Also, an organized test framework helps in avoiding duplication of test cases automated across the application. In short Test frameworks helps teams organize their test suites and in turn help improve the efficiency of testing.

Types of Testing Frameworks

As we have eliminated Record/Playback method, let us explore about the existing automation methodologies. There are several test automation frameworks available, among these the selection is made based on the factors such as reusability of both the scripts and the test assets. The different test automation frameworks available are as follows,

- Test Script Modularity
- Test Library Architecture
- Data-Driven Testing
- Keyword-Driven or Table-Driven Testing
- Hybrid Test Automation

A. Test Script Modularity

The test script modularity framework is the most basic of the frameworks. It's a well-known programming strategy to build an abstraction layer in front of a component to hide the component from the rest of the application. This insulates the application from modifications in the component and provides modularity in the application design. When working with test scripts (in any language or proprietary environment) this can be achieved by creating small, independent scripts that represent modules, sections, and functions of the application-under-test. Then these small scripts are taken and combined them in a hierarchical fashion to construct larger tests. The use of this framework will yield a higher degree of modularization and add to the overall maintainability of the test scripts.

B. Test Library Architecture

The test library architecture framework is very similar to the test script modularity framework and offers the same advantages, but it divides the application-under-test into procedures and functions (or objects and methods depending on the implementation language) instead of scripts. This framework requires the creation of library files (SQA Basic libraries, APIs, DLLs, and such) that represent modules, sections, and functions of the application-under-test. These library files are then called directly from the test case script. Much like script modularization this framework also yields a high degree of modularization and adds to the overall maintainability of the tests.

C. Data-Driven Testing

A data-driven framework is where test input and output values are read from data files (ODBC sources, CVS files, Excel files, DAO objects, ADO objects, and such) and are loaded into variables in captured or manually coded scripts. In this framework, variables are used for both input values and output verification values. Navigation through the program, reading of the data files, and logging of test status and information are all coded in the test script. This is similar to table-driven testing (which is discussed shortly) in that the test case is contained in the data file and not in the script; the script is just a "driver," or delivery mechanism, for the data. In data-driven testing, only test data is contained in the data files.

D. Keyword-Driven Testing

This requires the development of data tables and keywords, independent of the test automation tool used to execute them and the test script code that "drives" the application-under-test and the data. Keyword-driven tests look very similar to manual test cases. In a keyword-driven test, the functionality of the application-under-test is documented in a table as well as in step-by-step instructions for each test. In this method, the entire process is data-driven, including functionality.

E. Hybrid Test Automation Framework

The most commonly implemented framework is a combination of all of the above techniques, pulling from their strengths and trying to mitigate their weaknesses. This hybrid test automation framework is what most frameworks evolve into overtime and multiple projects. The most successful automation frameworks generally accommodate both Keyword-Driven testing as well as Data-Driven scripts.

This allows data driven scripts to take advantage of the powerful libraries and utilities that usually accompany a keyword driven architecture. The framework utilities can make the data driven scripts more compact and less prone to failure than they otherwise would have been.

The utilities can also facilitate the gradual and manageable conversion of existing scripts to keyword driven equivalents when and where that appears desirable. On the other hand, the framework can use scripts to perform some tasks that might be too difficult to re-implement in a pure keyword driven approach, or where the keyword driven capabilities are not yet in place.

IV. IMPLEMENTATION OF FRAMEWORK

Though commercial testing tools are well-defined and easy to use, they are often accompanied by two major limitations -cost and license constraints. Many IT organizations should pay a huge price for commercial automation tools and maintaining their inflexible licenses over long periods of time.

This is where open source tools like Selenium add a lot of value to contemporary commercial tools like QTP, Silk Test, etc. However, implementation of Selenium in project automation space is also not devoid of challenges.

Challenges in Selenium Implementation

Selenium, in its existing form, offers great value for automation. However, the tool's existing framework cannot scale up to enterprise test automation needs and poses certain challenges, such as:

- Lack of re-usability and maintainability of code base
- A user-friendly interface for tester to create test case
- Backward compatibility
- Migration of old test scripts in the event of version upgradation of Selenium engine.

Beside these, often there is a complex test case structure based on Selenium API call, which requires decent coding knowledge.

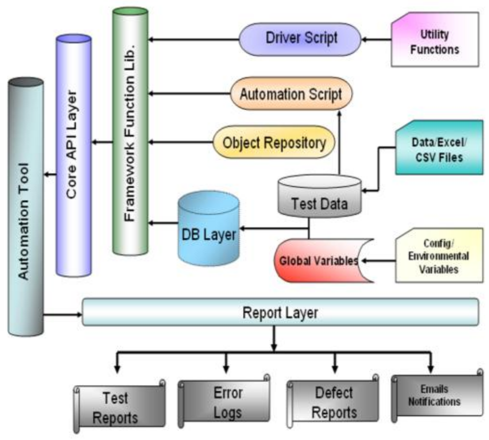
Comprehensive Selenium Framework(CSF)

To overcome the challenges associated with implementation of Selenium for project automation, we have developed a framework – the Comprehensive Selenium Framework (CSF).

We carried out a inclusive market analysis on different selenium-based automation frameworks and found that none of them offered a complete bundled package solution to any of challenges discussed above.

Hence, we developed a comprehensive automation framework, which offers an integrated bundled package solution to all challenges associated with implementation of Selenium for automation.

Implementation Approach



4.1: Basic Architecture of CSF

- CSF implementation approach basically involved implementation of function modularization and abstraction layer to all our integrated modules.
- We developed different modules at abstraction level, which made test case creation an easy task by utilizing reusable functions, utility methods and Selenium APIs.
- The basic constituent of CSF is a common "Function Library," which also has reusable modularized methods with application logic implemented at the abstract level.
- The chief driving engine of CSF is its integrated module called "CORE," which supports Selenium 2.0 and Web Driver technologies. This facilitates migration of test scripts from before newer versions of Selenium in the event of Selenium upgradation.
- CSF maintains a common "Object Repository," which makes editing and maintenance of test cases very easy, especially when features of an application under test change during different build releases.
- CSF provides a user-friendly UI for test case creation in domain language of application. Tester can write without a thorough knowledge of coding as CSF presents very simple test case structure.
- CSF can be easily integrated with different tools like Hudson, TestNG, JUnit, Ant, etc., depending on the project requirements. Besides, test reports can also be easily integrated with test management tools.

Benefits of CSF

- Easy migration and upgradation of test scripts
- Easy to use for testers
- Well-defined user interface for test definition and creation
- Easy maintainability and reusability of test scripts
- Complete pre-bundled package

- Saves time in test case creation/modification
- Saves testers from the complexities of test frameworks

REFERENCES

[1]. M. Sharma, R. Angmo,"Web based Automation testing and Tools," International Journal of Computer Science and Information Technonologies(IJCSIT), vol. 5, no. 1, pp. 08-912, 2014.
[2]. Singh, B. Tarika, "Comparative Analysis of Open Source Automated Software Testing Tools: Selenium, Sikuli and Watir" International Journal of Information & Computation Technology, vol 4, pp. 1507-1518, 2015.
[3]. R. M. Sharma, "Quantitative Analysis of Automation and Manual Testing", International Journal of Engineering and Innovative Technology, Vol. 4, Issue 1, July 2014.
[4]. Meenu, Yogesh Kumar, "Comparative Study of Automated Testing Tools: Selenium, SoapUI, HP Unified Functional Testing and Test Complete", Journal of Emerging Technologies and Innovative Research, Vol. 2, Issue 9, September 2015.
[5]. M. Kaur, R. Kumari, "Comparative Study of Automated Testing Tools: TestComplete and QuickTest Pro", International Journal of Computer Application, vol 24, no. 1, 2011.
[6]. RasulNiyazimbetov,"Web application testing solutions with selenium".[5] Deepti Gaur, Dr. Rajender Singh Chhillar ,"Implementation of Selenium with JUNIT and Test-Ng", IJCSMS International Journal of Computer Science and Management Studies, Vol. 12, Issue 03, Sept 2012.
[7]. Sherry single, Harpreetkaur,"Selenium keyword automation testing framework", International Journal of
[8]. Advanced Research in Computer Science and Software Engineering, Vol.4,2014
[9]. Gaurav Saini, Kestina Rai, " Software Testing Techniques for Test Cases Generation" , International Journal of Advanced Research in Computer Science and Software Engineering, Volume 3, Issue 9, 2013.
[10]. Monika Sharma and RigzinAngmo, "Web based Automation Testing and Tools",inte rnational journal of Computer Science And Information Technology (IJCSIT), Vol. 5(1),2014, ISSN:0975-9646, pp. 908-912.
[11]. Navaraj Javvaji, Anand Sathiyaseelan, Uma Maheswari Selvan,"Data Driven Auomation Testing of Web Application Using Selenium" Conference Proceedings, STEP-AUTO2011.
[12]. .C. Kulkarni, Y.C. Kulkarni,"Automating the web applications using the selenium RC",ASM's International Journal of Ongoing Research in Management and IT e-ISSN-2320-0065, 2011.
[13]. H. Kaur,Dr. G. Gupta, "Comparative Study of automation testing tools:selenium, quick test professional and testcomplete," International Journal of Engineering Research and Application, vol. 3, no. 5, pp. 1739-1743, 2013.
[14]. ShivkumarHasmukhrai Trivedi, "Software Testing Techniques" , International Journal of Advanced Research in Computer Science and Software Engineering, Vol. 2, Issue 10, October 2012.
[15]. Neha Bhateja, "A Study on Various Software Automation Testing Tools", International Journal of Advanced Research in Computer Science and Software Engineering, Vol. 5, Issue 6, June 2015.