International Journal of Engineering & Technology, 7 (2.8) (2018) 345-349

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research Paper

Business driven automation testing framework

R. Anand*, M. Arulprakash

[1]*PG Scholar*, [2]*Assistant professor*,
Department of CSE, SRM Institute of Technology, kattankulathur, Tamil Nadu, India.
**Corresponding Author E-mail: anandmca1903@gmail.com*

Abstract

Due to versatile growth of software industry, we need to invent and adapt new technologies to reduce the production cost and to increase the quality. Now a days the industry is moving towards the 100% of automation testing and is being done by the different kind of users. The same test script will be executed by the domain expert instead of a technical expert, so we can't expect him to understand the complexity of scripts and to relate the test script with business scenarios. To simply the users view of automation testing we need a layer to hide the technical implementation and to introduce some simplified way to understand the test script and relate with business scenarios, if needed he should be able to modify the test scripts without having the technical knowledge. The gap between the customers' expectations and the actual product behavior will be reduced using this proposed approach.

1. Introduction

Automation testing is a hot matter in the modern software industry. Due to heavy competition in the current industry, the software products should not only fulfil the customers' needs. It should go beyond the customers' expectation. So the service providers need to enhance their product and make it available for their customers. Then only they can sustain in business and lead the concern with expected profit. To achieve this excellence, they need to make sure the quality of the product. Since they need to measure the quality in frequent intervals, it is cumbersome to attain it manually, there the automation testing is pitched in.

Automation testing is taking a vital role in the industry to make sure their products are in the acceptable quality within the stipulated time. There are a lot of automation testing framework are available in the industry like modular driven, data driven, keyword driven, hybrid testing framework. We can have our customized framework in two tier or three tier or n tire framework.

Now the automation testing technology has been developed drastically, the following concerns can be achieved due to the technological growth.

1. We can run the test asynchronously.
2. The tests can be executed in the distributed environment.
3. Multithreading in test execution is possible.
4. We can schedule the test execution at any time and run using the tools like Jenkins.
5. The test report can be integrated with management tools and get the report automatically.

All frameworks and technological improvements are dealt with how we can develop the script and organize our work. Now a days the testing is not only done by the technical expert, it is also being done by the end users. All end users definitely will not be a technical person. If we give our automation test scripts to them to execute and assess the quality of the product, it is transparent that they will be perplexed. The current frameworks are all focused with how the scripts are organized and how the test execution will be. So as per end user point of view, we should infer them what to execute instead of how to execute. Here the behavior driven testing concept is introduced. In this approach we are concentrating about the business scenarios verification. This approach will be suitable for the application development life cycle like incremental model, agile, dev ops. This framework is very useful for Quality Assurance people, User Acceptance Testing specialist, product owner, clients, and end users. It can be understood and used by all stakeholders. One of the important features in this framework is, all tools used by this framework are open source. So it is cost effective for implementation and maintenance.

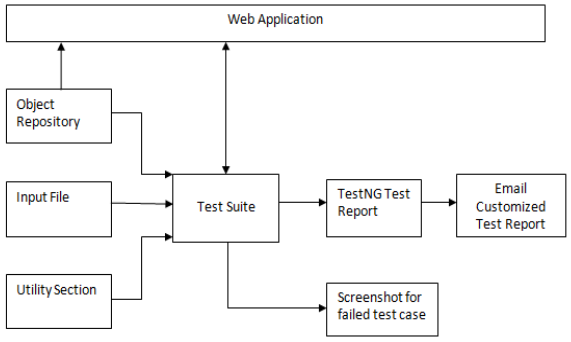
2. Related Work

They have designed test automation framework based on selenium web driver and TestNG testing framework.

The framework was designed in this related paper includes five components listed below.

- Object Repository (To identify the web objects)
- Input File (To pass the scenario related information)
- Utility Section (To write reusable functions)
- Test Suite (To organize the test flow)
- Customization Test Report (To generate the report in PDF format)

2.1 Design of Existing Automation Framework



3. Existing Problem

Most Software product development groups have two quality gaps:

Producer Gap - Difference between what is specified vs what is delivered
Customer Gap - Difference between what the producers actually delivered vs what the customer wanted.
With Existing Automation tools, the automation test suite can be developed against the stakeholder's requirements and executed on continuous basis to close the gap. In acceptance testing, the stakeholder needs to verify whether the build software product is right. But the stakeholder is unaware of things happenings in the test suite as it is built with the help of programming languages.

4. Proposed Solution

The following are proposed as a solution for this producer and customer gap problem.

4.1 Solution with Cucumber

Cucumber is a software tool based on Behavior Driven Development (BDD) framework which is used to write acceptance tests for web application. It allows automation of functional validation in easily readable and understandable format (like plain English) to Stakeholders, Business Analysts, Developers, Testers, etc. The test suites are then translated into the programming language by Cucumber, which supports multiple programming and scripting languages. Selenium is used to drive the browser. Cucumber facilitates collaboration and hence a more quality product as a result of better collaboration and communication between various roles in software development life cycle aligning with the business stakeholder. Along with above solution, the existing problem that would be addressed are:

- Test Coverage
- Execution Time
- Economical Open Source Tool
- Scheduling the job to run at different intervals

4.2 Simple Cucumber Scenario Feature:

Sign in to application Scenario: Users should see sign out button once signed in Given I open the Login page of the application When I enter valid credentials and click sign-in button Then I verify the application Home Page is displayed And I verify the sign out button is displayed at bottom of home page The above cucumber test is easy to validate by the Stakeholder and reduces the gaps further close.

4.3 Objectives & Scope

The objective of Automated Test Framework with Cucumber are as follows:

- To reduce the gap between software developers and stakeholders
- To reduce Testing Cost and Time
- To improve Test Coverage
- To reduce Redundancy
- To gain confidence in the system
- To reduce the number of defects found by users
- To run regression test overnight and weekends

4.4 Need for this Project

The need for Automated Test Framework with Cucumber are as follows: · We need to deliver the project using a good reliable tool which is open source · The Management should also understand what the project does, using Cucumber a layman can understand what is test case · Deliver quickly with a high standard compared to QTP · Now a days, agile has become the most commonly used software developmental model which involves stakeholders and developers work closely. In that case, cucumber is more useful to validate the requirements. · Software Testers with less coding knowledge can use this framework to develop the test suites.

5. Analysis of Test Automation Tool

Cucumber Tool Cucumber is a software tool used by software development team for testing the software. It uses Behavior Driven Development (BDD) style to run the automated test suites. It uses a language called "Gherkin". It uses the plain English text to describe the behavior of the application which is easily understandable and readable format to Developers, Testers, Business Analyst, etc. Initially, cucumber supports only Ruby as the programming language but later it has been extended to support other programming language which includes Java, C# and Python. Cucumber can be used along with Selenium framework. Most software projects have teams of several people working together, so the high quality communication is critical for their success. Good communication is not just about elaborately describing the ideas to others. It also requires to obtain feedback to make sure that it has been understood by others correctly. Because of this agile development teams build the software in small increments as they get regular feedback from stakeholders to confirm the small developments.

5.1 Ruby vs Java

Cucumber can be used along with Selenium. It supports many programming languages like Java, Ruby, PHP, Python, etc. But the most commonly used programming language is Ruby and Java.

Java:

- Java is a well-known technology
- Lots of developments have been made with it.

- Easy to find experts.
- Still not have much available Ruby developers.
- Java is open source.
- IDE: Eclipse (Free License)

Ruby:

- Ruby is more modern and more flexible.
- Ruby has a terser syntax and often requires much less effort
- Ruby is often slower to run and requires more memory than Java
- Ruby requires less time and effort to get a website up and running
- IDE: Ruby mine (Licensed Version)

By comparing both Java and Ruby, came to conclusion to use Java as the IDE
Eclipse is open source and it saves cost and the java has broader base of support.

5.2 Features and Scenarios Specification written for Cucumber have two parts:

Feature files containing scenarios expressed in the Gherkin language and Java files containing automation script for the steps in the scenarios.

Features:
The Cucumber test are written in a file called as Feature Files. ".feature" is the extension of the feature file and the file begins with a feature title and description:

Feature: Login to the Application in order to login to application, provide valid username and password and click the Log in button.

5.3 A Simple Feature File has Following Keywords:

- **Feature –** Name of the feature
- **Description** – Description about the feature (Optional)
- **Background** - Context to the scenarios in a single feature
- **Scenario** – Test scenario for the requirements

A feature usually contains a list of test scenarios. Apart from scenario, a feature file contains background, scenario outline and examples.
Background allows you to add some context to the scenarios in a single feature. A Background is much like a scenario containing a number of steps. The difference is when it is run. The background is run before each of your scenarios.
Cucumber Scenarios
Teams typically discuss scenarios in natural language first. Then, they express the scenarios more precisely using Gherkin's scenario structure:
Given <Initial Context>
When <To perform an event>
Then <Outcome of an event>
Each line in a scenario is called a step. Cucumber scenarios consist of steps, also known as 'Given', 'When' and 'Then'. Cucumber is not technically capable to distinguish the difference between these keywords. But it is strongly recommend to write the scenarios in a meaningful manner keeping the purpose of BDD in mind.
If the scenario has several Given, When and Then, then there is option to use 'And' or 'But'. Cucumber has a feature to use tables in the test scenarios. The table can be converted to list or map that can be used in step.
Given: The purpose of 'Given' is to put the system in a known state before the user starts interacting with the system. Avoid talking about user interaction in givens.
When: The purpose of 'When' is to describe the user interaction to the system or the key action which the user performs.
Then: The purpose of 'Then' is to describe what would happen if the condition mention in the 'when' is satisfied.

5.4 Step Definitions

The place where the automation code is written to make the cucumber scenarios to automated test is known as Step Definitions. The steps in the feature file (Gherkins) are directly mapped to step definitions. Each step in the feature file are mapped to step definition which contains the automation code, gets executed when the test scenarios are ran.
If any step definition is not available for any step in the feature file, it will throw an error. Similarly, if there is any error in the step definition code or returns exception to the runner, then the step will be marked as fail.
A step definition is a block of code written with any one of the programming language such as Java or Ruby, associated with steps in the feature file by a regular expression. The step definitions for the steps in the feature file will look like this:

```
Feature: TestCase_One
Scenario: Verify item in Mobile List Page can be sorted by Name
  Given I open the E-Commerce site
  Then The Home page is displayed
  And I verify the title of the Home Page
```

Step Definitions for the above steps:
(Java Programming Language)

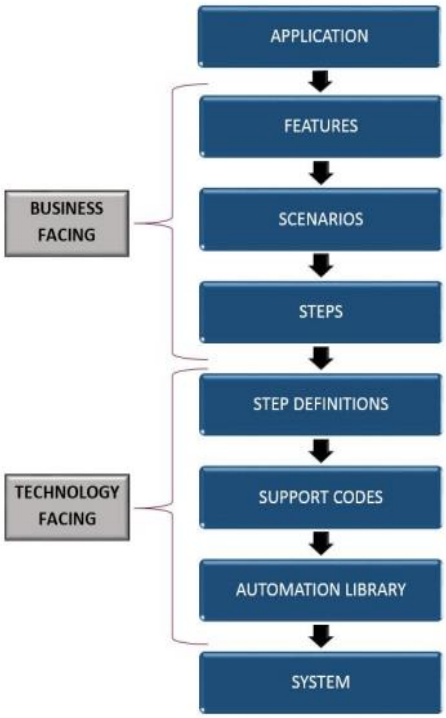
```
@Given ("^I open the E-Commerce site$")
public void i_open_the_ecommerce_site () throws Throwable {
      driver.get("http://live.guru99.com/");
      try {Thread.sleep(5000);}catch(Exception e){};
}
```

```
@Then ("^The Home page is displayed$")
public void The_Home_page_is_displayed() throws Throwable {
      assertEquals("Home page", driver.getTitle());
}
```

```
@And ("^I verify the title of the Home Page$")
public void I_verify_the_title_of_the_Home_Page() throws Throwable {
      PageFactory.initElements (driver, HomePage.class);
      HomeAction.VerifyHomePageTitle (driver);
}
```

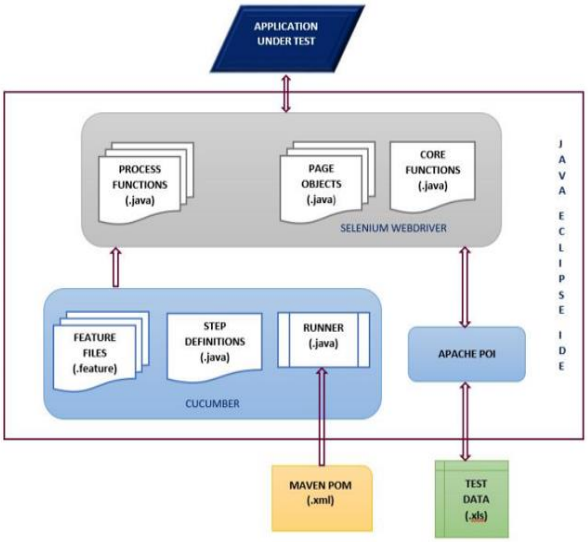
The above example is the representation of step definitions and how it is connected with steps in the feature file.
The above example is the representation of step definitions and how it is connected with steps in the feature file.

5.5 Cucumber Testing Stack



The above stack diagram shows how the cucumber works and the test cases are executed with cucumber.

5.6 Framework Architecture



The above architecture clearly defines how the Cucumber interacts with application under test with help of selenium web driver.

The key features of the framework as below:

- The cucumber test for the test scenarios are written in Feature files
- The automation code for the steps in the feature files are written in Step Definitions file
- Page Object Model is used as design pattern to capture the object repository for web elements
- Using the page object model, separate page class should be created for each web page in the application 24
- Separate page action class should be created for the each page class to perform the actions in the web page using the web element
- Reusable step definitions can be developed and used for any web applications
- Test data for filling the forms will be done through data driven framework (Excel Sheet)
- Extent Report is used to for Report Generation
- Test Reports will be placed in target folder
- If test scenario fails, the screen for each failed scenario will be taken and placed in Screenshots folder Email will be delivered to mail ids with test report
- This project can be integrated with Jenkins which is a Continuous Integration Tool which is used to schedule test and separate cucumber reports also be generated.

5.7 Extent Report

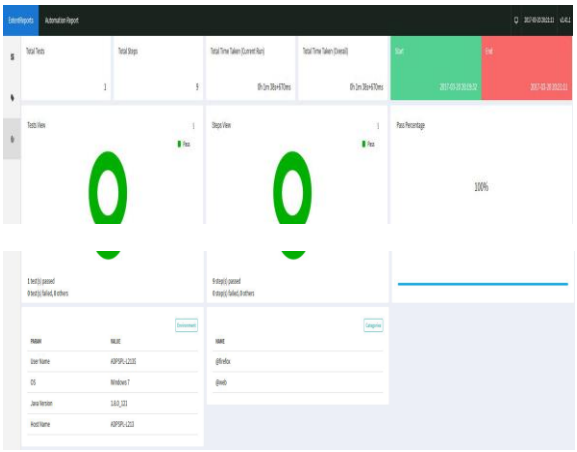
Extent Report is an interactive and real-time report. It helps to generate the custom cucumber report. This report can be integrated with this project using Extent Reports plugin.

In software test automation, after completing the test suite execution, it is necessary to have a test report for the execution status and it is the only way of evidence to get know the pass and fail status of the tests.

Most clients bother about the detailed test report of the execution status. Generally, the licensed automation testing tools will have the built-in report option to show the execution status. But in open source automation tool, the built-in report option is not available.

In cucumber, using json the html report can be generated but it will not be detailed and not good as the extent report. To avoid this, the extent report will be integrated to this project which will give the result in pie chart and the detailed status of test cases with Pass, Fail and Skip test status.

Sample Extent Report:



Advantages of Extent Report:

- Test status are shown in the form of PIE Chart
- Provision to replace existing report with new report
- Test results are displayed using Test Categories
- Screenshots can be embedded to the report
- Own name can be given to the test report
- Report will provide the System Information
- Possible to change the display order of the test

6. Conclusion

In this Dissertation an effort is made to automate the functional and Unit testing for the web applications to minimize the human effort involved in testing and to overcome the disadvantages of testing made with UFT/QTP tool. It has been observed that the testing with cucumber minimizes the software quality gap and reduces the communication gap between the developers and stakeholders.

So, Cucumber automation has brought into picture to cover its major part and it is successful. There are communication and misunderstanding issues which were in the system for long time and due to that the defect has been found even after the product is delivered to the customer. But with Cucumber it is fixed before coming to the knowledge of client and thus have added significant value to the system.

Regression testing which is done in every release and is time consuming has been taken first and test cases are automated. It has reduced the significant amount of time and is more effective. Now a days, even in agile projects cucumber plays a major role in automating the test cases where the stakeholders and developers involved in building a software product.

7. Recommendations and Future works

This automation framework can be enhanced to do automation in multiple browsers and parallel execution with help of Selenium Grid. With Selenium Grid, it is possible to perform multiple browser and multiple OS testing in parallel. Selenium Grid provides the flexibility to distribute the test cases for execution. It reduces the Batch processing time.

Further, this project can be enhanced by deploying the automation test source code in a common server which is accessible by all the team members. Source tree is used to integrate the changes made in the source code by the team members and the automation test can be used by the all team members whoever is having access. Using Jenkins we will be able to schedule to run the test scripts at any time. The test scripts will be run in backend and the test result will be stored in the common repository. So we can run our automation testing in non-working hours and analyze the test result in our working hours. It will drastically reduce our effort of test execution and analyze the test result.

References

[1] Satish Gojarea,*,Rahul Joshib,Dhanashree Gaigawarec, Analysis and Design of Selenium WebDriver Automation Testing Framework, 2nd International Symposium on Big Data and Cloud Computing (ISBCC'15), Available online at www.sciencedirect.com

[2] MacarioPolo,PedroReales,MarioPiattini. Computing Test Automation; IEEE Software, VOL. 30, NO. 1, January 2013.

[3] Maurizio Leotta, DiegoClerissi, FilippoRicca, CristianoSpadoaro. Comparing the Maintainability of Selenium WebDriver Test Suites Employing Different Locators; ACM,2013.

[4] AndrzaM,Giesel A. etl. Extension of Selenium RC Tool to Perform Automated Testing with Databases in Web Applications; Automation of Software Test (AST), 2013 8th International Workshop ,2013.125–131.

[5] Sherry Singla, HarpreetKaur. Selenium Keyword Driven Automation testing Framework, International Journal of Advance Research in Computer Science and software Engineering, VOL. 4,Issue 6,2014

[6] RigzinAngmo,Monika Sharma. Selenium Tool:A web based Automation testing Framework. International Journal of Emerging Technologies in Computational and Applied Science,2014.

[7] Z. Wanadan,J. Ninkang,Z. Xubo. Design And Implementation Of A Web Application Automation Testing Framework; Ninth International Conference On Hybrid Intelligent Systems, 2009.

[8] Selenium Documentation. [Online] (http://www.seleniumhq.org) (Accessed 15 DEC. 2014).

[9] TestNG Documentation. [Online]. (http://www.testng.org). (Accessed 25 DEC. 2014).

[10] F. Wang., Du.A Test Automation Framework Based on WEB.IEEE/ACIS 11th International Conference on Computer and Information Science,2012, 683-687.

[11] Liu Jian-Ping, Liu Juan-Juan, Wang Dong-Long, Application Analysis of Automated Testing Framework Based on Robot, 2012 Third International Conference on Networking and Distributed Computing.