Behavior Driven Test Automation Framework

Ramasswamy Subramanian  
Dept. of Computer Science, California State University Fullerton.  
ram2020@csu.fullerton.edu

Ning Chen  
Dept. of Computer Science, California State University Fullerton.  
nchen@fullerton.edu

Tingting Zhu  
Dept. of Computer Science, California State University Fullerton.  
miranda.zhu@csu.fullerton.edu

Abstract—The technological advancements and the concept of cloud computing made today’s industry to adopt agile development practices in software engineering and made web based software applications an integral part of our life. As the cost reduction and cost control are the two important driving factors for any business, the success of any business relies on the quality of its software systems. Web application test automation is a key component to achieve the superior quality, but complex technological features and rapid application development environment creates multitude of challenges in implementing a powerful, reliable and robust test automation framework which can support testing multilayer architectural software systems like automated testing of the user layer, technical, business layers and data layer of the software system. In this paper our proposed solution “Behavior Driven Test Automation Framework (BDTAF)” can address all the above constraints, as the framework can be implemented and maintained without involving much cost and can support test automation of all the different layers of a three-tier architectural system which will act as a driving factor behind substantially improving the test coverage, quality and reliability of the software system.

Keywords—Web application testing, GUI automation testing, Database testing, Services automation, Business driven testing, Agile test automation, Test driven development, Selenium WebDriver, FitNesse.

I. INTRODUCTION

With a rapid advancement in the technological arena, introduction of more and more robust and easy to use open stack web development tools and cloud computing technologies, web applications become predominant in today’s era than the standalone user and desktop based applications. The advancement in server-side and client-side scripting and programming languages made web application development easier by compressing complex business logic into couple lines of code, which makes web applications prettier than ever before. In addition, following agile development methodologies require more frequent releases of working software systems to the customer. These technological advancements and project environment create unique challenges for the software project teams involved in Web application testing, especially the test automation. It is true that many organization do not invest enough resources for test automation due to the cost and complexity involved in the setting up and maintaining test automation projects but only a robust test automation framework and appropriate automated regression test cases are the guaranteed way to thoroughly regression the software systems and support continuous releases. Without test automation and proper regression testing the product release can become an absolute failure on the very first day of the release. A classic example was the technical issues encountered by the users of HealthCare.gov [11], a health insurance exchange website operated under United States federal government on its first day (October 1, 2013) of the launch.

In this paper, we are proposing a powerful and robust test automation framework “Behavior Driven Test Automation Framework (BDTAF)” to support web application test automation and regression testing simple and robust enough that both technical and non-technical project team members can build and execute automation test cases without any prior experience in writing automation tests. BDTAF can support test automation of web applications built on three tier architecture seamlessly and the framework can easily be implemented in agile development projects, which is the hottest trend in today’s market. Such that, the business analysts and even business users of the organization can provide their requirements in a customized pre-defined test format, and the test can be easily incorporated into the existing suite of tests. This cost-effective test automation solution is built by integrating two powerful open source technologies, Selenium WebDriver [1] and FitNesse [2], which helps in keeping the test automation implementation and maintenance cost very small for the project management and helps in quickly recognizing the return on investment by achieving the highest level of automation success, improvement in test efficiency, highest possible test coverage and superior product quality with every release of the product.

II. RELATED WORK

A successful test automation requires very powerful test strategy and as per R. Potter [3], the execution of test automation using the user interface interaction exists from early 1990’s. A very common methodology is utilizing Record and Playback as per [4], where a script or a tool will capture the keyboard and mouse events when users executes manual tests in a software application and another driver script will playback these recorded tests against the same application with multiple sets of data. This approach heavily relies on the screen co-ordinates and hence very brittle and susceptible to failures.

Later, more improvised test automation modeling frameworks like WTM [5]. Web application test model has been presented by Kung et al., where by using Forward Engineering tools and Reverse Engineering tools, static and dynamic test artifacts of web application are extracted to create WTM instance model and through that model, behavior and structural tests are
derived to complete the test automation. This test model achieves optimal testing by deriving structural and behavioral test cases but the set up and maintenance of the test models requires heavy investments in terms of cost and resources, as it requires Forward and Reverse engineering tools to decode the web application such that WTM can parse necessary information. All these moving parts makes maintenance of the framework more expensive and the process more cumbersome.

DART [13] is a regression testing framework to support nightly builds of web applications by ripping GUI’s underlying document object model information, creating event flow graphs and setting up integration trees. This framework is primarily intended to do smoke and regression testing for GUI of a web application which is still under development. Having an automated framework which can creates test oracles based on ever changing GUI during development will increase the efficiency of development process. Nevertheless, this framework does not support full-fledged regression testing of all the business functionalities as well as all the layers involved in the application under test.

STIQ [6] a story testing framework helps building complex test cases by using reusable test components written in form of stories in the agile context. This framework supports data driven testing and helps users defining acceptance tests in the form of user stories. Xebium [7], a data driven testing framework to support integration testing of different components in a web application by recording tests using Selenium IDE, which is an open source tool but has been deprecated in the market. Both frameworks rely primarily on Selenium IDE to support test automation. As the underlying IDE support, has been deprecated, these frameworks cannot support robust test automation needs for any software applications developed using current technology.

NTAF [14] is an automation tool developed to support test automation of services layer of web applications to facilitate primarily the performance and stress testing by leveraging FitNesse and STAF to facilitate communication among the developers, stakeholders and at the same time executing test both locally and remote client/server based systems. As the goal of this framework indicates its main intention is to only support test automation of API layer only. This framework cannot be used to support test automation of GUI or Database layers of the web applications.

Bures et al. presents SmartDriver [8], which is an extension of Selenium WebDriver which helps users in creating more efficient automation test cases to support Web and mobile application automation by providing advanced mechanism to write adaptive test cases and mechanism to understand test failures easily for script developers and in turn makes script maintenance very adaptive. Nevertheless, this framework focuses only on the GUI layer of a Web application, which makes this framework supporting only one layer of any software systems developed using three tier architecture frameworks.

Castor et al. presents Extension of Selenium RC to perform Database testing [9], which helps in validating data involved in any web application by extending features of Selenium RC to connect and query the databases and compare the stored data against the expected data set. This framework also relies on deprecated Selenium RC framework as underlying principle and support only one part of testing needs in three tier architecture web applications.

Our new suggested framework BDTAF overcomes all the constraints we face in the existing test automation solutions and can be used to build automation test cases to test any Web application’s (i) GUI or User layer (ii) Services (API) or business layer and (iii) Data layer. Thus it can provide well rounded automation coverage needed for any software project. In addition, the automation test cases can be written in a simple, easy to understand, table like format where any business analyst and business users can write agile story based automation test cases which can be reliably used to test complex business functionalities.

### III. Behavior Driven Test Automation Framework

Crispin et al. [10] states that Agile testing means a way to learn the application such that the customer stories should drive the testing to be in par with agile values of working software by responding to the change. Some key factors for successful agile testing projects are (i) Continuous collaboration with customers, (ii) Automated regression testing (iii) Adopting an agile mindset, (iv) Implementing an end to end test solution and (v) Using the whole team approach.

The core concept of our new approach “Behavior Driven Test Automation Framework (BDTAF)” lives by above principles. BDTAF can be implemented as an end to end test automation solution for any Web based software applications by means of writing (i) Automation tests to validate the front end GUI layer, (ii) Automation tests to validate the middle layer or services layer of a software application, (iii) Automation test can also include database validation and more importantly (iv) Automation tests can be written and executed by technical team members like Developers and Automation test analysts, Non-technical team members like Functional Quality Assurance analysts, Business Analysts, Business users and the Stakeholders. The below sections will explain the BDTAF architecture, deployment and implementation details.

#### A. BDTAF – Architecture

BDTAF is built by integrating two powerful test automation solutions, “Selenium WebDriver” and “FitNesse” available in current market. Both are open source frameworks built with the capability to extend any test automation needs for Web based software applications. Each of these frameworks have
its own advantages and disadvantages, but integrating these
two frameworks gives users humongous potential and power
to successfully complete any kind of test automation needs for
a software project.

Selenium WebDriver is known for its robustness and
flexibility in terms of automating web based software
applications in multitude of browsers. An array of available
programming language bindings gives users the flexibility to
write tests in their own choice of programming language to
test the system. Leveraging the remote web driver features
from Selenium, the automated test execution can be done
either in the local development environment or in the
dedicated remote test environment.

Likewise, FitNesse comprises of FIT engine, FitLibrary and
FitLibraryWeb is very popular for its easy to use wiki based
test writing style. An open source tool used for automating
customer tests is called Fit. A wiki built on top of Fit is the
FitNesse used for automating acceptance testing [15], such
that any project team member can write automation tests using
predefined syntax in table like wiki format. FitNesse allows
excellent collaboration between all the stakeholders of the
project and enables communication between the project team
members and customers.

The aim of BDTAF is to exploit the advantages from both the
tools, such that automation tests can be easily developed and
executed against multitude of environments, browsers based
on the project needs and at the same time automation tests can
be easily maintained. In addition, by implementing continuous
integration model, the regression test suites can be scheduled
to execute against deployed build servers after every check-in,
and even automation tests can be written before completing
the development activity to support the core concept of agile
principle called test driven development methodology. Test
driven development or Test first development is a strategy for
shortening the cycles between test development and test
execution [12] and the aim of this approach is to find the
ambiguous or incomplete requirements early in the
development life cycle such that the rework effort can be
dramatically reduced at the later stages of the development or
the maintenance cycle.

Few important and notable features of our approach are:

- Test automation support for GUI layer, API layer and
  Database layer of web application.

- Automated test execution of application under test
  using multitude of browsers (Firefox, Chrome, Edge,
  Safari, Internet Explorer, etc.,) and operating systems
  (Windows, Linux, etc.,), by leveraging the power of
  Selenium WebDriver.

- Enables both technical and non-technical team
  members to write and execute automation test cases.

- Test execution support in local development
  environment as well as remote and distributed
  environments by implementing remote web driver
  capabilities inside our framework.

- Enables continuous collaboration between project team
  members and the business users as well as stakeholders
  of the application, resulting in faster feedback on the
  developed features and requirements can be delivered
  in the form of acceptance tests.

- Supports agile model by implementing test driven
development as well as involving all the project team
members in writing and executing automation tests.

As illustrated below in Figure 1, Selenium WebDriver will act
as the underlying platform for executing the automation tests
against the system under test (SUT). The users will interact
with FitLibraryWeb and write automation test cases in
predefined wiki format.
understandable and intuitive format, which completes the full cycle for test execution activity.

B. BDTAF – Automation support for three tier architecture systems

As shown in Figure 2, BDTAF consists of three independent Java classes which acts the core modules of the framework. These java classes understand the commands coming from the user triggered tests and determines what type of testing to perform on the SUT and provide result back to the user.

The three core java classes of the framework are
- SeleniumWebFixture
- SeleniumServicesFixture
- SeleniumDatabaseFixture

As the names clearly indicates, SeleniumWebFixture is the primary class where all the necessary commands and methods have been implemented to drive Web application and perform GUI based test automation. This module acts the heart of the framework where all the necessary command transformation will happen for successfully completing the test automation. Few important and key implementations in this fixture are listed below. These are few very common commands used in GUI based automation test cases to perform testing.

- **start browser**: Implementation to start the user defined browser.
- **open url**: To navigate to a website that user wants to perform testing
- **sendKeys**: To type any user supplied data into web application input fields
- **selectItem**: To select appropriate data from select boxes and combo boxes in the web application
- **clickElement**: To click on a button or hyperlink on the web application
- **verifyElement**: To perform an assertion in the test and make sure the expected element is available in the SUT
- **verifyText**: To perform an assertion in the test and make sure the expected text is available in the SUT
- **captureScreenshot**: To capture necessary screenshot of the SUT wherever user wants to see the state of the system during the test at run time and during any failures
- **close browser**: To close the browser successfully after completing the testing.

By reading the above sample commands, it is clearly visible that anyone in the project team who wants to write and build automation test cases shall use the above illustrated commands, follow the sequential steps of functional tests cases and write the automation tests in the predefined wiki format. The user who is writing automation tests does not need to worry about following java syntaxes or declaring variables or following any kind of programming paradigms, as BDTAF encapsulates all those nuances for the end user and enables them to write automation test cases in simple plain English statements.

Similarly, the test commands required for automating service or API layer of SUT are implemented in SeleniumServicesFixture class. This class has commands like GET and POST and helps users in writing API layer automation test cases.

Finally, SeleniumDatabaseFixture has necessary methods implemented to perform all the database related validations as part test automation. As any web application testing, cannot be a well-rounded complete test without validating the underlying data stored in the database, BDTAF has mechanism to evaluate the resulting data as part of either GUI based or API based tests. The framework can support validating expected data against the actual data from multitude of databases like Microsoft SQLServer, IBM DB2, MySQL, Oracle and PostgreSQL, provide valid connection string and query syntax are passed appropriately in the test along with expected data.

C. BDTAF – Elegant automation test cases

As detailed in section III-B above, BDTAF provides mechanism for users to write automation test cases in simple, easy to read and understandable format by following a structured mechanism as below. The test case will be in wiki format tables which will have three columns. The first column will specify the command to perform, the second column specifies the identifier or the element locator in the web application and the third column specifies the test data to be passed into SUT to complete the testing.

Consider the following example of testing a Login page of any Web email application. The necessary steps for a positive test are given below:
- Start browser → **FitNesse will send raw test request to BDTAF to launch a browser. BDTAF parses and identifies the browser from the request and invoke Selenium WebDriver to launch the correct type of the browser.**

- Navigate to the target web email application (gmail.com) → **BDTAF’s SeleniumWebFixture will pass commands to browser to navigate to gmail.com.**

- Verify login button is available → **Verification method in SeleniumWebFixture of BDTAF will be called to complete the assertion.**

- Type user name into Login field → **BDTAF’s SeleniumWebFixture will send parsed element ids and data to WebDriver to identify the correct element Login field in the web form and type the data into the field.**

- Type password data into Password field → **BDTAF’s SeleniumWebFixture will send parsed Password element id and encrypted data to WebDriver to type the data into the password field.**

- Click the Login button → **Click command will be issued from SeleniumWebFixture to the browser to identify and click the login button of the Gmail application.**

- Verify there is no error → **Verification method in SeleniumWebFixture will be used to complete the assertion that application did not display any validation error.**

- Verify the link with my username displayed in the home page of the email application → **Verification method in BDTAF’s SeleniumWebFixture will be used to complete the assertion and parsed user friendly result will be displayed back in FitNesse.**

- Log out of the email application → **FitNesse will send raw test request to log out. SeleniumWebFixture will parse the request and identify the element id and operation to perform on the element and instructs Selenium WebDriver to complete the operation.**

- Close the browser → **SeleniumWebFixture will issue shutdown command to Selenium WebDriver to close the browser and complete the test. BDTAF will pass all the parsed verification assertion results back to FitNesse to display to the user in a user-friendly format.**

The above sequence of steps can be written in wiki test table format as below.

Once the test is executed as explained in section III-A, the results will be displayed to the user in a nice formatted layout as below. The green shades confirm the successful results, whereas the red shades explain the failure and deviation of the application behavior from an expected test result. Below example explains a successful test run.
If there is any error in the input data, for example when the username and password combination are mistyped in the above example, the resulting failure test case will be formatted in red color as below:

![Test Case Example]

D. BDTAF - Maintenance

Maintenance being the pain point for any automated test framework, we have architected the BDTAF in a way that the core three modules of the framework will act completely self-contained and will not be affected by any new releases of underlying open source tools. Whenever there is a new release of Selenium WebDriver or FitNesse, we can upgrade to new stable versions of those tools by dropping the new jar files into framework library and continue supporting test automation with below simple steps.

- New stable releases of underlying open source tools (Selenium WebDriver or FitNesse) can be downloaded from their official website download pages.
- Copy the jar files and their source files into BDTAF library directory with all the dependencies.
- Remove the old, deprecated libraries.
- Update class path settings of the project to reference new libraries.
- Recompile the framework core classes referencing the new libraries.
- Execute the tests in the existing framework set up with all new dependencies.
- Check in the new libraries in the source code control systems to trigger the continuous integration builds using new libraries.

The whole process of upgrading will act as a simple Plug and Play approach, such that framework maintenance and upgrade can happen without affecting the productivity of the team.

IV. Conclusion

This paper comprehends the key features and benefits of BDTAF, a powerful and robust test automation framework that can be quickly implemented and leveraged for creating and maintaining a solid test automation solution. This framework enables management and project teams to participate in test automation process and advances the visibility on test reporting. It also improves communication between project teams involving developers, testers, business analysts, business users and stakeholders.

BDTAF’s strength in comparison with related works includes its elegant test automation support for web applications developed using multi-tiered architecture. An end to end test automation, which includes automated testing of web application’s GUI layer, API layer and Database layer can be implemented for both simple and complex software applications using this framework.

With continuous integration support model, BDTAF enables its users to write automation tests even before the development completion and enables users to follow test driven development methodology. In addition to its strength, the test framework can be easily extendable to support multiple browser versions and operating systems. Software project teams can be greatly benefited from BDTAF by bridging the gap of complexities involved in developing and maintaining a powerful and robust test automation solution to ensure the product quality every release supporting the core Agile principles without affecting the project cost.

REFERENCES


