****

**TEST AUTOMATION FRAMEWORK**

**THE PROPOSAL**

**By: Optimum Paradigm**

**Dated: July 2018**

****

Contents

[TEST AUTOMATION - An Overview 3](#_Toc519750233)

[Scope 3](#_Toc519750234)

[About Optimum Paradigm 3](#_Toc519750235)

[Problems & Issues Faced 3](#_Toc519750236)

[Case Study 4](#_Toc519750237)

[Offered Solution 4](#_Toc519750238)

[Solution Details 4](#_Toc519750239)

[How OP Test Framework Works 5](#_Toc519750240)

[Benefits 7](#_Toc519750241)

[Conclusion 7](#_Toc519750242)

**TEST AUTOMATION - An Overview**

Software testing is an essential, requisite, costly, and time-consuming activity in the software development life cycle. As is true for software development in general, reuse of common artifacts can provide a significant gain in productivity.

Test Automation becomes increasingly critical and strategic necessity. Assuming the level of testing in past was sufficient, how do we possibly keep up with this new explosive pace of web enabled deployment while retaining satisfactory test coverage. The answer is more people or greater level of automation.

Automation plays a vital role for applications/products intended for customers who undergo continuous customization and number of releases. One can automate the regression freeze suite and run them on need basis when new features are implemented. Moreover, as testing deals with testing the application or the system under a variety of platforms, configurations and circumstances, automation of execution-related activities offers another potential source of savings in the testing process.

**Scope**  
The scope of this document is to present Optimum Paradigm Test Automation approach and practice for all development and maintenance projects. This white paper also covers the complete details on architecture, design and implementation of automation framework.

**Definitions and Acronyms**

* kLOC: 1000 Lines of Code  
  Regression: (End to End processes designed by the product owner). No coded business logic.
* Feature Regression: (Specific regression tests for work done within the scrum). No coded business logic.
* Pages: This is the business logic layer.
* Objects: This contains the properties of the object for identification.

**About Optimum Paradigm**

Being a Technology Oriented Company, our specialty is complete Software Quality Assurance/Testing cycle. We offer wide range of Quality Assurance/Testing and Development solutions to meet your business requirements, from engagement models to tailored testing by industry.  
We always rely on our comprehensive, creative and dynamic industrial standards and processes for  software Quality Assurance/Testing.

**Problems & Issues Faced**

Through our extended experience in an IT organization and after working with some industries most renowned clients, we have observed and faced some very crucial issues and problems, throughout the software life cycle. Trying to justify some important ones below;

* A new software module in an existing software, results in new Lines of code.
* New Lines of code can break the existing functionality.
* More Lines of code means more defects.
* More defects result in more time spent on testing (Unit, System, Regression).
* If testing require more time then, this will impact software delivery.
* Customers can be frustrated due to slippage in software delivery.
* Overall Customer dissatisfaction will influence the business.

**Case Study**

To better gauge the overall situation, that has been described above, let’s take an example below;

There was a new module requested by the customer, that is being implemented in the last release, reworking the software architecture of 40% of the code, which increased the complexity and at the same time it also introduced the new 50 kLOC i.e. 50 kLOC. An analysis of the data took place in the defect tracking database shows that in the code base, the number of defects per kLOC in the field has risen by 15% over each previous release.

Another concern is the number of patches issued to fix defects. More patches released means more unattended piece of code, which provides an additional opportunity to introduce undetected new defects, because patches do not undergo formal product verification. Waiting for patches is also a source of costumer’s dissatisfaction. Finally, the number of defects found in the product verification phase has also increased. Fixing each defect introduces another opportunity to create new defects, which also may go undetected before the release goes to the customer.

**Offered Solution**

The major goal is to set up the process that must minimize the number of defects that escape from one phase of the software product development to the next, this also means that reduce the number of the defects that escape from software developer testing through to product validation, which ultimately reduce the number of defects that escape to the customer and become the biggest cause of customer dissatisfaction.

Our proposed solution is the latest one in the software industry having the updated tools and technologies that is very flexible and can be fitted in any organization. Our Solution is having the following criteria;

* Our Solution won’t have any impact on the existing software budget as most of the tools that are being used are open source.
* Our Solution has the minimal or no impact on the overall project schedule.

**Solution Details**

The proposed solution is to introduce test automation framework to reduce the problems described above. Our Test automation framework is the latest automation framework that best fits in any organization. Our test automation framework will automate a selection of existing tests as part of the upcoming release, area accounts for 40% of undetected defects. The tests are of high value, well defined and has low maintenance.

If 50% of existing tests are automated and tests are written to cover all new functionality, after three releases, the tests will cover 75% of software functionality. Most importantly, the automated tests will run as part of the nightly software build, to detect defects early in the life cycle, before they escape to the product validation, or to the customer.

Our tests will have the following characteristics;

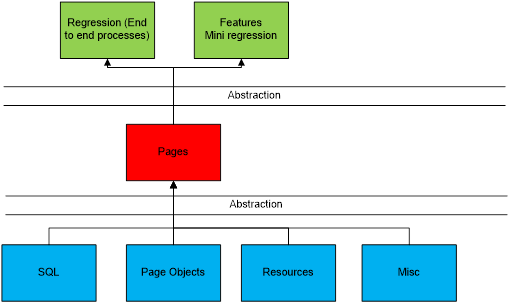
* Have well defined boundaries
* Have unambiguous pass/fail criteria
* Will use actual customers use cases
* All the tests will be reviewed by the subject matter experts
* Test would be generic and are environment independent (Browsers, Databases, Deployment Servers etc.)

**How OP Test Framework Works**

We are on a mission to make your life simpler with an easy installation process, a framework that includes the best tools and SDKs available, while reducing the maintenance efforts. Test Framework is designed for the entire test automation community, and is collaborative at its core. It is driven by passionate automation experts that create and contribute valuable add-ons (handful and reusable test actions) for Mobile, Web and API testing, all according to the latest technologies.

Application

(Browsers)



Drivers

(Selenium WebDriver)

Runner Framework

(TestNG, Maven)

Custom Code

(DSL)

**Supports all common platforms**:

Since most products these days are targeted for multiple platforms, our desire was to enable you to create automated tests for Web and Mobile (Android/iOS) platforms, and run them on a variety of supported browsers and mobile devices, no matter whether you use Windows, Linux or mac OS.

Setup & DevOps:

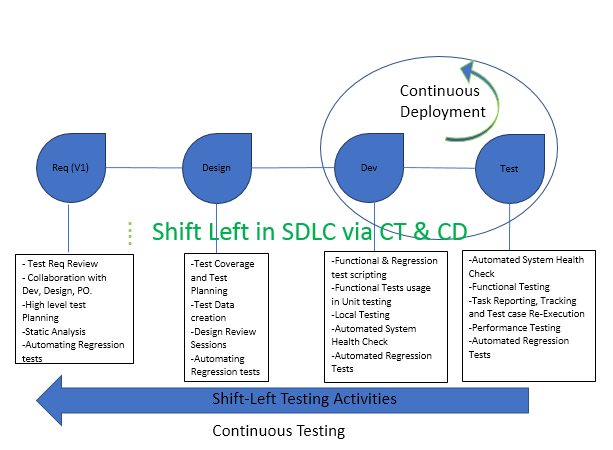
Configuring different tools and frameworks into a single [continues integration](https://blog.testproject.io/2017/05/11/jenkins-ci/) flow can be challenging, it requires expertise and consumes valuable time. Focus on creating the automation and we’ll focus on always keeping you up-to-date with the latest stable components and drivers. The following essential elements of DevOps are achieved for continuous integration and continuous delivery.

* **Code repository with version control**
* **Continuous Integration (CI)**
* **Automated Build**
* **Automated Release through Continuous Delivery or Continuous Deployment (CD)**

**Unified SDK**:

One known challenge with open source tools is the variety of SDKs that test automation experts need to adopt in order to support their products. Our team has invested great efforts to develop just a simple and unified SDK which targets both Web and Mobile applications. The SDK is based on standard Selenium commands, so developing new tests or porting existing is as easy and straightforward as possible. At this point, the SDK is available only for the Java language, but support for additional technologies (JavaScript, Python and C#) will follow very soon.

Shifting Left:

Shift Left Testing focuses on moving testing as early as possible during development.  By moving testing as early as possible, we can pick issues earlier.  This is good because it is much easier to address issues in the earlier phases when we can make architectural adjustments.  In order to get value out of doing Shift Left Testing, and **DevOps** is the way to make this happen with automation.

**Cloud based dashboard, reports & management**:

One of the most important parts of a test automation framework is informative reports. In OP Test Framework we have created a dashboard that provides insights and detailed logs about your projects and tests which allow you to easily troubleshoot errors by quickly identifying failed steps. Our cloud based web interface was created with team collaboration in mind. Easily share your tests, UI elements and any other component that will help your teammates to create better automation in less time. Schedule job executions, get a real time execution status and utilize your automation across your entire organization.

**Benefits**

There are many benefits that can incur after implementation of the Automated Test Framework, which includes the following;

* As the automated tests run as part of the nightly software, build and produce a report of test failures that indicate defects introduced by the previous day’s code.
* Automated tests that will be created by the Testers can also be used by the developers for the unit testing.
* The nightly test runs catch defects introduced during the development phase and ensures, that the fixes for defects found during product verification do not break another functionality.
* It reduces the number of defect that escape from software development through to product. verification, which reduces the chance of undetected defects escaping to the field.
* It provides an opportunity to detect defects in patches before they are released to the field
* No Cat Mouse chase game between the dev and test.
* Increase customer satisfaction shown by fewer support calls and complaints to sales staff.
* Increase discipline applied to analysis of the software under test and test planning.
* As the automated test suite grows, the benefits of automation accumulate over time.
* Increase morale in the product testing department because it eliminates monotonous defect retesting.
* A framework provides future opportunities to further improve quality by automating additional tests.

**Conclusion**

With the above facts and information, the proposed solution is to introduce test automation with the goal of reducing the number, undetected defects that escape through to product verification and into the customer. Automation is key capability we need and is the backbone of any “Shift Left” practice.  Automating a consistent deployment of an application across all the stages (DEV, TEST, PROD, Public Cloud, Private Cloud, etc.) will help build confidence in the deployment.  Sharing the results with operations will help build trust in building a culture that supports continuous delivery and helps us do this Shift Left paradigm shift.

--------------------------------------------------------------------------------------------------------------------------------  
THANKS FOR THE CONSIDERATION