|  |
| --- |
|  |
| Platform–X Test Strategy |
| White Box Test Strategy Document |
|  |
| **Baskaran, Karthiga** |
| **3/9/2016** |

**V1.0**

|  |
| --- |
| *The document details about the white box test strategy plan for OSMP Requirement achieved through Platform-X Architecture* |

[](https://www.wellsfargo.com/)

**Revision History**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Rev. No.** | **Author** | **Date** | **Revision Description** | **Sections Affected** |
| 1.0 | Karthiga Baskaran | 03/9/2016 | Initial Draft | Initial Draft |
|  |  |  |  |  |
|  |  |  |  |  |

Table of Contents

[1. Introduction 2](#_Toc445284751)

[1.1 Overview 2](#_Toc445284752)

[1.2 Why White Box Testing? 3](#_Toc445284753)

[1.3 In-Scope 3](#_Toc445284754)

[1.4 Out of Scope 3](#_Toc445284755)

[2. White Box Test Strategy Plan 3](#_Toc445284756)

[2.1 Logical Diagram 3](#_Toc445284757)

[2.2 Test for Race Conditions 4](#_Toc445284758)

[2.3 Entry Criteria 4](#_Toc445284759)

[2.4 Exit Criteria 4](#_Toc445284760)

[2.5 Test Cases Update Plan 4](#_Toc445284761)

[2.6 Testing Tools 4](#_Toc445284762)

[3. References 5](#_Toc445284763)

[4. Glossary 5](#_Toc445284764)

[4.1 Acronyms 5](#_Toc445284765)

[4.2 Font Color Legend 5](#_Toc445284766)

# 

# Introduction

The document provides detailed Platform-X white box testing strategy planned out for any Wells Fargo application.

## Overview

*Platform-X Architecture* is a cross platform testing solution to enterprise application world. It leverages various automation testing frameworks to support cross platform testing of the enterprise application. The following are the advantages of *Platform-X Architecture*

* Redefines the testing philosophy
* Reduces Manual Testing effort and cost
* Provides solution to cross platform testing using various automation testing tools
* Better Productivity and code health
* Innovative and Addresses flaws quickly
* Provides User Friendly Dashboard to the management and team - across the applications code coverage, code quality and defect rate, etc
* Complete solution with Devops setup for any enterprise application

## Why White Box Testing?

*Platform-X Architecture* recommends “Shift-left” Testing Approach, to help in finding and fixing the problems/defects earlier in a project lifecycle. The following are the advantages in adapting this approach,

* Provide extended support to development team in ensuring all the functional requirements are met in the earlier stage of the project lifecycle.
* Provide in-depth/extensive test scenarios for the complex business requirements.
* Provide support to Black Box Testing team in focusing more on UI testing and less on functional testing.

## In-Scope

The scope of this document is to list down the features to be white box tested, facilitate the white box testing strategy plan for each feature listed, tools used for the white box testing and archive and update plan of the test cases.

## Out of Scope

This document does not cover the manual test strategy or the E2E test strategy applicable to any Wells Fargo Application.

# White Box Test Strategy Plan

## Logical Diagram

The flow diagram (refer *Figure 2.1*) depicts the white box testing approach for any Wells Fargo Application.

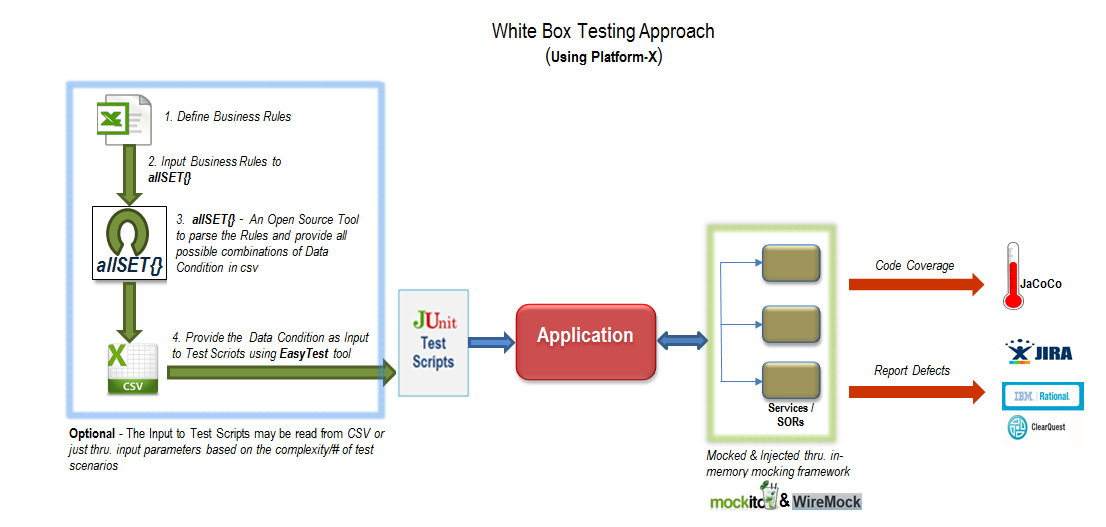


Figure: 2.1

* Define the Business Rules of each feature to be white box tested and input to *“****allSET{}****”*(i.e.an Open Source Tool to generate all possible combinations of Data Conditions (in a CSV file) to be tested).
* The Data Conditions (that are determined by ***allSET{})*** are read by JUnit Test Scripts using “*EasyTest*”, (i.e. an Open Source Tool to parse the input for JUnit Test Scripts through Excel/CSV/DB etc.).
* The Data Conditions determined for each Use case of a feature [Refer *Section 2.3*] will be translated to *JUnit* Test cases.
* The third party services/SORs or any interfacing system required for the test scenario will be mocked and injected using in-memory Mocking framework i.e. *Mockito.*
* Report *Defects* found, with a categorization as “*White Box Test Defect*” in *Clear* *Quest/JIRA (any defect tracking tool)*.
* Report the test case (white box test) coverage code often, so that team set goals (e.g. getting to a certain amount of coverage, or amount of test case that need to be written) and then monitor progress toward those goals.

## Test for Race Conditions

Bugs that are hard to fix, are the ones that are hard to reproduce. Generally these hard-to-reproduce bugs happen because of the [racing conditions](https://en.wikipedia.org/wiki/Race_condition) in the application. Typically functional tests are conducted in a single threaded mode. But racing conditions happen only in a multi-threaded mode. Thus to validate the application that don't suffer from racing condition bugs, white-box testing strategy will be leveraged. In this strategy targeted white-box tests will be developed which will fire multiple threads on the application component to validate they produce consistent results every single time and don't suffer from racing condition bugs.

## Entry Criteria

The maximum benefits of white box testing can be unleashed only if requirements are signed off and hence the following are considered as entry criterion for the white box testing.

* Frozen Functional Specifications related to requirement to be white box tested.
* Approved and Released System Design Document.
* There is no item in query issue register, which is related to requirement to be white box tested.
* Completed, Approved and Released Technical Design Document.
* Completion of the code development (abstraction layer) related to requirement to be white box tested.

## Exit Criteria

The following are considered as exit criterion for the white box testing.

* All/Accepted white box test cases have been passed.
* The code is complete with respect to Requirements under test i.e. there are no missing features or element.
* Upon the arrival of accepted code coverage i.e. 80%~100%.
* All reported white box test defects (that are documented into Clear Quest / JIRA (any defect tracking tool)) are resolved.
* Any defects remaining unresolved from white box test been documented/logged into Clear Quest /JIRA (any defect tracking tool) and explained why they are unresolved.

## Test Cases Update Plan

The test cases for the features identified (or additional features) will be re-used with revised or appended test cases based on the enhancements or requirement changes with peer discussion with the stakeholders.

## Testing Tools

The following are list of the testing tools required for white box test execution, defect reporting and test documentation repository

|  |  |  |
| --- | --- | --- |
| **S. No** | **Tool(s)** | **Comments** |
| 1 | Platform-X | White Box Unit Testing Framework |
| 2 | allSET{} | An open source tools to generate complex/all possible test scenarios for the provided data conditions |
| 3 | JUnit | Create in-depth/extensive test scenarios for the complex business requirements |
| 4 | Mockito | In-memory mocking framework to support integration with SORs |
| 5 | WireMock | Open source service virtualization solution |
| 6 | EasyTest | An open source tool to provide data to Junit test cases i.e. test data will be injected to each scenario |
| 7 | JaCoCo | Provide code coverage report |
| 8 | Clear Quest / JIRA | Defect Logging System |
| 9 | Subversion / GitHub | Source control |

# 

# References

# Glossary

## Acronyms

|  |  |  |
| --- | --- | --- |
|  |  |  |

## Font Color Legend

|  |  |  |
| --- | --- | --- |
| **S. No** | **Legends** | **Comments** |
| 1 | Blue font | New Requirements |
| 2 | Red | Needs More Clarity or confirmation |
| 3 | Green font | Existing Requirements |
| 4 | [Blue Font with Link](file:///C:\Users\U476998\Projects\Complex%20Splash\Blue%20Font%20with%20Link) | Hyperlinks |
| ~~5~~ | ~~Light Blue Font~~ | Not Required/Applicable scenario to be tested |
| 6 | ~~Font~~ | Will (or May) Not be addressing at this stage |