**Documentation**

The documents included in this Software Quality Plan are as follows:

* Software Requirement Specification:
  + A description of what the system should do, a bussiness analyst is responsible for this, designer and developer shoudle be responsible of reviewing this.
* Software Design Description:
  + A description in blocks of the intended implementation to comply with the software requirements, a senior developer or designer should be responsible for this and developer and business analyst are reviewing this with designer.
* Verification and validation plans:
  + Describe the processes that will be used to verify and validate the application. Included in here are tasks such as code analysis, inspections, and testing. Test engineers should be responsible for this, and business analyst, developers should review together.
* Verification and validation results report. Test engineers should be responsible for this by using test management tool to generate data and formulate to a report.
* User documentation: user guide and plugin developer guide. Designer is responsible for this. Business Analyst reviews this document.
* Software configuration management plan

Software Requirement Specification (SRS) is basically an organization's understanding or blueprint of a customer or client's system requirements and dependencies at a particular point of time prior to any actual design or development work.

The Information in this SRS for this project is based on IEEE 830 and will include the following components:

* Interfaces
* Functional Capabilities
* Performance Levels
* Data Structures/Elements
* Safety
* Reliability
* Security/Privacy
* Quality
* Constraints and Limitations

Software Design Description (SDD): A description in blocks of the intended implementation to comply with the software requirements, describe the components and subcomponents of the software design, including database and internal interface, should be expanded in details afterwards for facility the implementation. Its content contains the followings:

* Identification the Date of issue and status, scope of software, issuing organization, responsibility or copyright information, references and context.
* Identify the design stakeholders for the design subject.
* Identify the design concerns of each identified design stakeholder.
* Address each identified design concern.
* Organized into design views: contain design elements, and different design concerns (functionality, reliability, performance and maintainability).
* Design viewpoint: functional views of system (such as functions, input/output), and also should select design languages for viewpoint specification.
* Design overlays: to represent additional information which only relevant to one design viewpoint.
* Design rationale: lists the reasons of designing the system and how to make those design decisions, includes any design issues and options being considered.

~~The design of this software would be documented in Product Perspective and Functions, User Classes and Characteristics, Operating Environment and Design and Implementation Constraints.~~

**Verification and validation plans**

1.0 Introduction

The purpose of this validation describes verification requirements of the project. These include an overview of inspections, test planning for individual components of the project.

2.0 Inspections

After the design of a module is completed, the design description should be immediately reviewed. It includes the analysis of use cases, whether the designed functions meet the user’s requirements. Also the whole software architecture and security factor are concerned. After implementation of specific modules, we have moved on to code inspections. This part would mainly focus on logic error and potential performance issues, which should be done in pair every week. The coding style should match the organization standard. Each review feedback is given to a specific author.

3.0 System test plans and procedures

* Testing process:

Write test programs to test each module and unit of the system

* Test recording procedures:

Trac is an open source, Web-based project management and bug tracking system. This would be used to manage the tests, testers issue the bug ticket, and corresponding developer would need to fix the bug, and then close the ticket. When the percentages of each tested component are all 100%, the whole test is done.

**Software configuration management plan**

* Developer submits the code to the development branch.

During development, developer teams should set up fresh regression test to ensure old function are compatible with new requirements and work properly.

After each new crucial submit to the development branch, a new development build and test request is issued and current development version would deploy into integration test environment and carry out the integration test. If integration test have other system dependency, stable fork of those systems should be used instead of connecting with development version of those system, to minimize the interference of other system.

* Test engineers build a new version and deploy it into the staging testing environment, carrying out the system test.

Before system test, tester would have plan for test task milestone, issuing bug ticket and assign timeline for each milestone. During system test, tester should assign timeline and developer for each bug. At each milestone, a test progress report should be sent to the testers and developers. When the bugs found during this test procedure are fix, testers should issue a request for the system is ready to be deployed into the beta testing environment.

* A group of testers conduct the acceptance test in this pre-production environment, ensuring it meets the pre-release standard.

Before UAT, users should have enough amount of business test cases to test the new requirements and proved by developer and tester.

During UAT, if a major bug is found, the priority and the influence to the system should be considered, decide whether to fix. After all the test cases are passed, testers should issue a request for ready to release to actual production environment.

* Deploy the pre-release version into production environment.
* For daily, there is a fresh environment for regression test, ensuring the stability of daily works.
* Developers will also be responsible for the integration tests, ensuring compatibility of different modules in the system.

modified:

1. Introduciton

the purpose of this SCM plan describes guidelines and standards for software configuration, tracking issues during software developement lift cycle. It applies to software developed in JAVA and other developement file types.

* 1. Definiton

Software Configuration Management (SCM) is a discipline for ensuring the integrity of a software including its components and to make its evolution more manageable.

It consists following three principle activities:

identify the components in the software and state relation betwee them

determine the constraint placed on accessing and changing the components.

documents records the components, relationships and functionality of the software.

1. Organization

Software configuration management tasks contains software configurations manager and individual developers. Manager should publise changes and results to Change Control Board, and disscues with developers with any issues.

Responsibilities of Software configurations manager:

Administrate the Software Configuration Management tool

Ensure developers should be able to use software management tool

Maintain the management procedures document

Responsibilities of developers:

use software configuration management tool

follow the chang control procedues

1. Configuration identification

The software contain consists of controller, view, model, plugin and Application modules. Those 4 modules are integrated together, except the plugin module, the changes of rest of 4 modules should not submitted individually. The release of different version number should be based on this format: Vxx.yy.zz,aa, where xx is major version number of current working requirements, yy is the minor version of sub set of those requirements, zz is the requirements number, aa is the increment number of bug fixes.

Libraries:

All dependencies libraries should be resides in the lib directory except the native java library.

The jar package in lib directory is under source control of apache ivy dependency manager.

Any changes made on the dependencies should be able to rollback to previous versions based on ivy version number.

Release control:

All release and bug fix must provide changes notes of known issues and what is in the release.

The main 3 release channels are:

Development branch: it publish to integration environment for implementation specific used.

Staging branch: it published to staging test environment for system test, functional test and other performance test.

Beta branch: it published to pre-production environment for user acceptance test.

Master branch: it is official release of the software.

1. Configuration Control

Contents in of change control board:

Member

Role

procedures of changes

Content of code audting reports:

lines of code

code complexcity

comment ratios

1. Configuration Auditing

Internal peer reviewing and code inspection and formal review of software design review should be host by senior developer. Those resulting documents should be placed on a server for secure storage purpose.

In addition, all audits task should contain following information:

who performs the audits

what is audited

how formal is the audits