Software Configuration Management Plan

Internet Banking System

Table of Contents:

# Scope & SCM activities

# SCM organizational role

# Identification & Description

# Work products & Documentation

# Configuration Control

# Version Control

# Reviews and Audits

## Review Guidelines

## Formal Technical Reviews

# Problem Reporting and Corrective Actions

## Responsibilities

## Data Collection & Evaluation

1. Scope & SCM activities**:**

During the time of the software development, we will be making changes to our original plans. Software Configuration Management Plan is developed so that we can identify the change, control the change, make sure the plan is implemented correctly and make sure that we report the change to others.

The main purpose of SCM is to make, report and track any changes made to the original software development plan. It is applied throughout the software development process and will help us to keep track of changes and also help us go through and make changes. SCM procedures will give us a good map out of the software so that if we need to make more changes it will be relatively easy to do so. SCM will maximize productivity by minimizing mistakes. For SCM to be successful, all the members of the software production team will have to take time to report the changes that they think are necessary and/or to notify others of changes that they may have made. This is sort of boring and time-consuming work, but it is very important.

**SCM activities are developed to:**

* Identify change
* Control change
* Ensure that change is being properly implemented
* Also have a way to document the change.

1. **SCM organizational role:**

Since we have rather a software development team and project manager, each member of the team will accept responsibility for software configuration management. This is necessary since there are only three members of the team. If one of the members reports changes remaining two members have to take up the job of authorizing change and ensuring that change is properly implemented. This will reduce or eliminate confusion between the team members regarding changes to the software. Since all the members participate in the SCM, the need to interact change with other software engineering teams is eliminated. We will also keep all the members on the customer’s side informed of all the changes for acceptance. The changes that do not affect the user’s knowledge of the software will be presented to a selected member on the customer’s side. These changes will be noted in a specific section so that we can refer back to them to know what the original plan was and why the changes were made. If the changes are made or suggested so that they will affect the way the customer uses the software, then those changes will be discussed with the project manager. Once a customer has decided to go with the change then and only then will change be implemented. We will also extensively report or document all the changes so that customers will have access to it after the software is packed and delivered.

1. **Identification & Approval:**

In this section, we will describe the way software configuration items will be identified for the software configuration management plan.

* Identify change

If during the software development phase, a team member suggests a change in the software then we need to have the teamwork on the suggestion and to figure out if the change is necessary and is justified.

* Approve change

We want to be able to have control over any change within the software. We can’t afford to have one member of the software development team think of a change or implement it without telling any other member of the team. This can create huge technical problems for the software. We want to develop rules so that no member of the team will think of and implement change without the permission of other members than the project manager. We will be using the change request report form to suggest changes in the software.

1. **Work products & Documentation:**

* Identify change

Once the change is identified, a change request form will be produced and sent to all the SCM team members.

* Control change

After the evaluator (SCM team member) got the change request form, a change report form will be generated.

* Ensure that change is being properly implemented.
* Document the change.

Once the change is approved we will document the change in the library. And we will change the software version number if it is necessary.

1. **Configuration Control:**

Changes will be controlled by using human procedures and automated tools. Here are the steps, which will be taken to control change.

· Request the change

· Software developer will evaluate the change request

· The result of the evaluation will be presented as a change report

· Final decision on change will be made

· If change is approved

1. Define constraint

2. “Check out” items for changes

3. Make the necessary change

4. Apply SQA activities

5. “Check-in” items

6. Apply testing activities

7. Rebuilt the software

8. Distribute the software

1. **Version Control:**

As a result of changes, the version number of various modules will be increased accordingly. We will be using a universal version number system for all modules. We will also have a final version of the entire product. Major documentation will also have version numbers, such as User Manual or Design Specification.

When a change request is filed, a change report will be created. After the change is finalized, it will be documented in the library. We will be using a decimal point version number system for baselines dates as follow:

160422\_release\_XX

160422\_intermediate\_XX

**Bug Fix**:

If enough bug fixes have been done on the product/module, the bug fix portion of the version number will be increased. The number of user-visible bug fixes will also affect when the bug fix number is increased. The more visible bug fixes have been made, the closer the bug fix number will need to be increased.

**Minor Update:**

If functionality is added to the product/module that will increase the user-friendliness/performance but does not change the way a function/interface work, the minor update number may be increased. Several of these changes will warrant a version number change. Again, visible changes (interface) will cause the version number to increase sooner.

If a major functionality has been added to the product that greatly increases the user’s experience or greatly improves the program performance, a minor version update will be issued immediately.

**Major Update**:

We do not foresee any change in the major version numbers. The product will be labeled as version 2.

1. **Reviews and Audits:**
   1. **Review Guidelines**

* Software project plan review
* Confirm project needed environment
* Confirm the same folder structure
* Confirm access rights
* Confirm management tools
* Check baseline internal and external audits
  1. **Formal Technical Reviews**

Here are the FTR that we will conduct during the software process: - Walkthroughs - Inspections After each form (interface) we design, we’ll do a test on the interface using the block box testing method. And for each week, when the team set down comes to a meeting, we will ask the teammates to inspect the interface, then hook up the other’s work, and do a walkthrough of all the interfaces.

Software Requirements stated the data requirement and specifications. Risk Mitigation, Monitoring & Management, is used to prevent, monitor, and manage the risk.

The Data Design document is about the data flow between each form (interface), and forms in the database.

Code Review.

Test specification review.

Control reviews and audits.

1. **Problem Reporting and Corrective Actions:**

Data about the project process should be collected, evaluated, and disseminated. During each release meeting, we will present a customer a prototype of the project; ask the customer’s opinion on the design, and present different options for them to choose from in many cases. Even during the two weeks between two meetings, if we try to keep the customer have the latest and 1st hand knowledge on the process of the project, we occasionally posted the project online, letting them examine the process. Or let them download a prototype version, let them run it in the real-world enlivenments, and then ask for feedback.

1. **Naming Convention:**

**Documents:**

Each file is named after the data it is describing or abbreviation of the document it is describing example:

1. Project Management Plan.docx
2. SRS.docx (system requirements specifications)

**REQ:**

Every requirement is based on the format BS – REQ - <abbreviation of the feature it is describing>-#number.

Example:

1. BS – REQ – CR – 01.

**Design:**

Sequence diagram: every file is named after the functionality it is describing with ‘\_’ instead of any space and the number of it is order example:

1. Admin\_Edit\_Client\_06

Wireframe: every file should be named after the function it is describing and containing the initials WF describing it is a wireframe. Example:

1. Client Registration\_WF\_02

**Code:**

Front-end: each file should be describing the functionality it is performing with extension .aspx or .css

Back-end: each file should be describing the functionality it is performing with extension .aspx.cs

Example:

1. ClientRegistration.aspx
2. ClientRegistration.css
3. ClientRegistration.aspx.cs