



Software Configuration Management Plan

I. Table of Contents

I. TABLE OF CONTENTS.....	1
1.0 INTRODUCTION.....	2
1.1 SCOPE AND INTENT OF SCM ACTIVITIES.....	2
1.2 SCM ORGANIZATIONAL ROLE.....	2
2.0 SCM TASKS.....	3
2.1 IDENTIFICATION.....	3
2.1.1 Description.....	3
2.1.2 Works products and documentation.....	5
2.2 CONFIGURATION CONTROL.....	6
2.2.1 Description.....	6
2.3 VERSION CONTROL.....	6
2.3.1 Description.....	6
2.3.2 Increasing Version Number.....	6
2.3.3 Work Products and Documentation.....	8
2.4 CONFIGURATION STATUS ACCOUNTING (CSA).....	8
2.4.1 Description.....	8
2.4.2 Work products and documentation.....	8
3.0 SOFTWARE QUALITY ASSURANCE OVERVIEW.....	9
SCOPE AND INTENT OF SQA ACTIVITIES.....	9
REVIEWS AND AUDITS.....	9
3.1 Generic Review Guidelines.....	10
3.2 Formal Technical Reviews.....	11
3.3 SQA Audits.....	12
3.4 PROBLEM REPORTING AND CORRECTIVE ACTION/FOLLOW-UP.....	13
3.4.1 Reporting Mechanisms.....	13
3.4.2 Responsibilities.....	13
3.4.3 Data Collection and Valuation.....	13



1.0 Introduction

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 to make sure that we report the change to others.

1.1 Scope and Intent of SCM Activities

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 are 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 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.

1.2 SCM organizational role

Since we have rather small software development team, each member of the team will accept responsibility for software configuration management. This is necessary since there are only three members in the team. If one of the member reports changes remaining two members have to take up a job of authorizing change and to ensure that change is properly implemented. This will reduce or eliminate confusion between the team members regarding changes with 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 client's side informed of all the changes for acceptance. The changes that do not really affect user's knowledge of the software will be presented to a selected member on the client'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 customer uses the software, then those changes will be discussed with the entire



client team. Once a client has decided to go with the change then and only then will changes be implemented. We will also extensively report or document all the changes so that client will have access to it after the software is packed and delivered.

2.0 SCM Tasks

In this section we will try to detail all-important SCM tasks and will assign responsibilities for each. All of the SCM tasks will be performed by three members of the software development team members. We will try to keep one-person from the client's team informed of all the changes that do not affect users. All the changes that affect the use of the software will be discussed with entire team on the client's team during the meetings.

2.1 Identification

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

2.1.1 Description

- **Identify change**
If during the software development phase a team member suggests a change in the software then we need to have the team work 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 not afford to have one member of software development team think of a change or and 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 permission of other members. We will be using the change request report form to suggest changes in the software. Below is the link to the page that contains Change Request Report generator and the picture of it.

<http://www.engin.umd.umich.edu/CIS/course.des/cis375/projects/crr/request.html>



The screenshot shows a web browser window titled "Change Request Form: main page - Microsoft Internet Explorer". The address bar shows a URL. The page content is a form with three main sections:

- 1. SCI Identification**: Includes fields for Version/Control #, Document Name, and Page Number(s).
- 2. Contact Information**: Includes fields for First Name, Last Name, Project Manager (dropdown), Title, Others, Phone, Location, and Email.
- 3. Level of Request**: Includes radio buttons for Cosmetic Change, Modify Function, and Add New Function, and a dropdown for Request Priority (set to Medium).

At the bottom of the form are buttons for "Continue to Step 2 >>>" and "Start Over". A "Help" sidebar on the right provides instructions for each section:

- SCI Identification**:
 - Version/Control #**: The version/control # of the software you tested. Check at "about this" at software or at "about this" at the documentation that come along with the software.
 - Document Name** - Optional: Name of the Document if it's attached along with the software. Included version number, publish code (find at the back of the page).
 - Page Number(s)** - Optional: Also included the page number(s) if you define the changes in the code book or the documentation.
- Contact Information**:
 - Name**: Please enter your correct name in order for the programmer to contact you quickly. Company IDs are not allowed.
 - Title**: Select the title provided, if no match, please use "other", and state your title at the box at the right side.
 - Other**: If you select your title as "other", please fill in this box of your contact title.
 - Phone**: i.e., 1234567890. (max 10 digits). Please include the area code. The phone # that we can reach you during the office hour. Do not use office phone, or personal cell phone.

- **Ensure that change is being properly implemented**
We want to have team members looking over the change. Since all the teammates will be working separately, it is possible to have made mistake in implementing the change. To make sure this doesn't happen, we want to set up times when team members will look over the change that other members have implemented and to finalize the change.
- **Document the change.**
Since change has to be documented from the time that a team member suggests change to the time change is finalized, we will end up with extensive documents. We will be using the change request form developed by Delta group to request change. To approve a change we will be using change report form, which will be submitted to change control panel (remaining member of the software development team). Link to the Software Change Report Generator is as follows.

<http://www.engin.umd.umich.edu/CIS/course.des/cis375/projects/crr/report.html>



The screenshot shows a web browser window titled "Change Report Form - Microsoft Internet Explorer". The address bar shows a URL starting with "http://www.cyberrovers.com". The main content area displays a form with four sections:

- 1. SCI Identification**: Includes fields for "Version/Control #", "Document Name", and "Page Number(s)".
- 2. Contact Information (Evaluated)**: Includes fields for "First Name", "Last Name", "Project Manager", "Title", "Others", "Phone", "Ext.", "Location", and "E-mail".
- 3. Contact Information (Request)**: Includes fields for "First Name", "Last Name", "Project Manager", "Title", "Others", "Phone", "Ext.", "Location", and "E-mail".
- 4. Evaluation**: Includes a dropdown for "Level 1: Cosmetic Change", a "Change Level" field, and a "Perceived Effects" text area. A note says: "Please use the COCOMO calculator below to help you evaluate the change." Below this is a "Perceived Effects" text area.

A "Help" window is open on the right side, listing the sections and providing instructions for each field. The help text includes:

- SCI Identification**: "the version/control # of the software you tested; check at 'about this' at software or at 'about this' at the documentation that come along with the software."
- Document Name - Optional**: "Name of the Document if it's attached along with the software."
- Page Number(s) - Optional**: "Also included the page number(s) if you define the changes in the code book or the documentation."
- Name**: "Please enter your correct name in order for the programmer to contact you quickly."
- Title**: "Select the title provided, if no match, please use 'other', and state your title at the box at the right side."
- Other**: "if you select your title as 'other', please fill in this box of your correct title."
- Phone**: "ie, 1234567890 (10 digits). Please include the area code. The phone # that we can reach you during the office hour: can be your office phone, or personal cell phone. Extension: fill out if necessary. (max 50)"
- Location**: "In the following manner: Room #, Building #, Beach #, etc."

2.1.2 Work products and documentation

- Identify change
Once the change is identified a *change request form* will be produced and will be send to all the members of the SCM team.
- Control change
After evaluator (SCM team member) got the change request form, *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.



2.2 Configuration Control

2.2.1 Description

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

- Request the change
- Software developer will evaluate the change request
- The result of the evaluation will be presented as change report
- Final decision on change will be made
- If change is approved
 1. Define constraint
 2. “Check out” items for changes
 3. Make necessary change
 4. Apply SQA activities
 5. “Check in” items
 6. Apply testing activities
 7. Rebuild the software
 8. Distribute the software

2.3 Version Control

2.3.1 Description

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.

2.3.2 Increasing Version Number

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:

<major update>.<minor update><bug fix>



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. For used for bug report is as follows.

Critique: It would be better to define specific criteria for this. For example, the bug fix number is incremented after 5 or more user visible bugs of the same type are found and corrected.

The image shows a screenshot of a Microsoft Word document titled "Bug Report Form". The document is displayed in a window titled "Bug Report Form - Microsoft Word". The document content includes the following text:

Cyber Rovers
www.cyberrovers.com

WMITS Software Configuration Management Plan (06/06/00)
Page 7

Bug Report Form

User Name: _____

Bug Number: _____

Bug Description:

Priority: Choose One

High Priority: _____

Medium Priority: _____

Low Priority: _____

Where:

Cause:

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.



Critique: It would be better to define specific criteria for this. For example, the version number is incremented after 10 or more user visible changes of the same type are found and corrected.

If a major functionality has been added to the product that greatly increase 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 major version number. The product will be labeled as version 2.

2.3.3 Work Products and Documentation

A single document titled **Version Revisions History** will be used to document all the version revisions. An online bug report and tracking system will also be used to monitor and document all the bug fixes and enhancement requests.

2.4 Configuration Status Accounting (CSA)

We will be using three different ways to communicate with the team members and to inform others that changes may concern.

2.4.1 Description

Three ways or the three tools that we will be using to communicate with other members or the people associated with software development.

- **Online help desk:**
This tool will be used to help us communicate with clients. Client can submit bug report or enhancement report through online help desk. In reply we can post the progress report to each of the items that have been submitted to the help desk.
- **Change request report:**
We will have two different forms that we will use as tools to request a change or to report a change to the SCM team. These documents will be in html format and we will be able to send them through web.
- **Verbal communication:**
Since our software development team is small and all the team members are in constant touch with each other it would be better to communicate verbally.

2.4.2 Work products and documentation



- **Online help desk**
- **Change request report generator**
- **Emails**
- **Test errors will be documented electronically**
- **All suggestion made during peer review will be noted**

3.0 Software Quality Assurance Overview

For the description of the methods for software quality assurance please refer to Software Quality Assurance Plan (SQA)

SQA will focus on the management issues and the process specific activities that enable a software organization to ensure that it does “the right things at the right time in the right way.” SQA plan provides a road map for instituting software quality assurance.

Comment: See my comments relative to this section in the SQA Plan.

Scope and Intent of SQA Activities

The objectives of SQA are:

- A quality management approach
- Effective software engineering technology (methods and tools)
- Formal technical reviews that are applied throughout the software process
- A multi testing strategy is draw
- Control of software documentation and the changes made to it
- A procedure to assure compliance with software development standards when applicable
- Measurement and reporting mechanisms

Reviews and Audits

A formal technical review (FTR) is a software quality assurance activity that is performed by software engineers. The objectives of the FTR are:

- (1) to uncover errors in function, logic, or implementation for any representation of the software;
- (2) To verify that the software under review meets its requirements;
- (3) To ensure that the software has been represented according to predefined standards;
- (4) To achieve software that is developed in a uniform manner;
- (5) To make projects more manageable.



3.1 Generic Review Guidelines

ISO 9001 is the quality assurance standard that applies to software engineering. The following are the 20 requirements delineated by them. And we are try to follow them as our quality assurance plan.

- Management Responsibility
- Quality System
- Contract Review
- Design Control
- Document and Data Control
- Purchasing
- Control of Customer Supplied Product
- Product Identification and Trace Ability
- Process Control
- Inspection and Testing
- Control of Inspection, Measuring, and Test Equipment
- Inspection and Test Status
- Control of Nonconforming Product
- Corrective and Preventive Action
- Handling, Storage, Packaging, Preservation, and Delivery
- Control of Quality Audits
- Training
- Servicing
- Statistical Techniques

3.1.1 Conduction a Review

There are two kinds of reviews we'll do, review cases with the client, and review cases with other teammates.

For the changes that will affect the clients' performance when they use the software, we have to consult them first. But before take the cases to the client, the entire team member has to agree with the change. And keep a good record of the project before and after changes.

3.1.2 Roles and Responsibilities

As stated in 1.2, SQA Organizational Role, the rule of each team member will be very confusing since we have a relatively small team.

- Conceptual & Advanced Interface Development: Ray Yu
- User Interface Design & Development/Trainer: John Deng
- Editor/Tester/Maintenance: Bhavin Patel

3.1.3 Review Work Product

For each period (weekly, in our case), we'll generate a work report from each member. In the work report, we will state each member's work for the past week, problems that encounter, problem that can't be solved, any cautions to remind. This report will be extremely helpful when comes to documentation and writing the help menu.



When we meet with our client, we will have a minute for the record. In the minute, we will have the questions that we make before we go see the client, answered got from the clients, clients' request, opinions on the current work, self reminder, schedule for next meeting, etc.

3.2 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 block box testing method. And for each week, when the team set down come to a meeting, we will ask the team mates to do a inspection on the interface, then hook up the other's work, do a walkthroughs of all the interfaces.

3.2.1 Description of review *Walkthroughs*

3.2.1.1 Description and Focus of Review

This review mainly focus on the integrations of the parts that we design (such as interfaces, form, database.) We will ask other team members to do the walkthroughs with the presence of coder.

3.2.2 Description of Review *Inspection*

3.2.1.1 Description and Focus of Review

This review is mainly focus on the correctness of the parts that we designed. Usually allow the other two team-members to do a private test, without the designer's interrupt. This idea is try to allow other team-members bring out the test cases.

3.2.1 System Specification Review

System specification usually changed after each weekly meeting, and after each meeting with our client. As for this moment, most of the system designs have been settle down. And since we are actually doing a "bug fixing" originally, so the system specification is pretty much there. Even we have expended the project into a higher level, the basic mapping of the project is still the same. For more information, please see the document titled "System Specification".

3.2.2 Software Project Plan Review

The purpose of Software Project Plan is over look of the whole project. For more information, please see the document titled "Software Project Plan".

3.2.3 RMMM Review

RMMM, Risk Mitigation, Monitoring & Management, is use to prevent, monitor, and manage the risk. For more information, please see the document titled "RMMM".

3.2.4 Requirements Reviews (Models, Specification)

Software Requirements stated the data requirement, specifications. For more information, please see the document titled “Requirement Plan”.

3.2.5 Data Design Review

The Data Design document is about the data flow between each form (interface), and forms to the database. For Architectural Design, please see the document titled “Architectural Design”.

3.2.6 Architectural Design Review

The Architectural Design document is about the whole project design, layout, and data flow. For Architectural Design, please see the document titled “Architectural Design”.

3.2.7 Interface (GUI)

Up on the request of the client, we will redesign the interfaces from the previous version. We have 15 interfaces so far, and maybe more will be added later on. They all are Visual Basic front-end.

3.2.8 Component Design Review

Besides the 15 interfaces, we also doing a palm pilot integration, and online help desk for a period of time. We also doing some database re-engineering for the client’s database to make it more efficient and suit our project need.

3.2.9 Code Review

3.2.10 Test Specification Review

3.2.11 Change Control Reviews and Audits

3.3 SQA Audits

- Team members will have a weekly report on their individual performance for the past week. Any problems, question regardless on the performance of other team members will also noted there.
- Members will write part of the help menu that related to their design parts. And they also share between members.



- Any changes that will affect the project will consult with other team members before doing any changes. These are the changes that is minor or require little code change, but still different from the original architectural design.

3.4 Problem Reporting and Corrective Action/Follow-up

This section will describe problem reporting mechanisms that occur as a consequence of the FTRs that are conducted and the means for corrective action and follow-up.

3.4.1 Reporting Mechanisms

We will use the “tiny tool” that we designed on CIS 375 to do problem reporting, it is browser-based program, which is very convenience for the team members. Site name is <http://www.cyberrovers.com/crr>. It not only used by the team members to report problems to other team members, we can also train the clients to use it during the testing phase. When they submit the form, they will receive a copy of the request. The copy also carbon copy it to the DEQ manager, Lawrence Aubuchon. The request will send to all three members of the cyber rovers’ team.

An online help desk will also put online for the client ask any question regard on the operation of the software. This service will be provide during the training period, or up on the request of the clients.

3.4.2 Responsibilities

Each members of the team will be responsible for something. But we are helping each other out. Since we use the ego-less model, so we don’t select a team leader. But since Ray has a lot real world experience on software development and has great knowledge on the project, so he’s the spirit leader for the team. But as far as decision making, no changes will be make unless all three members agree on it.

- Conceptual & Advanced Interface Development: Ray Yu
- User Interface Design & Development/Trainer: John Deng
- Editor/Tester/Maintenance: Bhavin Patel

3.4.3 Data Collection and Valuation

To properly conduct software quality assurance, data about the software engineering process should be collected, evaluated, and disseminated. Statistical SQA helps to improve the quality of the project and the software process itself.

During each meeting, we will present a client a prototype of the project; ask the clients’ opinion on the design, present different option for them to choice in many cases. Even during the two weeks that between two meetings, if we try to keep the client have the latest and 1st hand knowledge on the process of the project, we occasionally posted the project online, so letting them exam the process. Or let them download a prototype version, let them run it in the real world enlivenments, and then ask for feedback.

