SeniorConnect Project
Configuration Management Plan_v2

6/11/2015

SeniorConnect Project Team
Nanyang Technological University
50 Nanyang Avenue
Singapore 639798

Revision History

Name	Date	A*MD	Reason For Changes	Version
Li Yishan	1/11/2015	A, M	Draft configuration management plan, part 1, 2, 3	V1.1
Mao Huiqi	1/11/2015	A, M	Draft configuration management plan, part 4, 5	V1.2
Li Yishan	2/11/2015	A, M	Corrections and clarification	V1.3
Li Yishan	4/11/2014	M	Finalize configuration management plan	V2

*A - Added M - Modified D – Deleted

TABLE OF CONTENTS

1 Identification 5

1.1 Document overview 5

1.2 Abbreviations and Glossary 5

1.2.1 Abbreviations 5

1.2.2 Glossary 5

1.3 References 8

1.3.1 Project References 8

1.3.2 Standard and regulatory References 8

1.4 Conventions 8

2 Organization 8

2.1 Activities and responsibilities 9

2.1.1 Decisions process and responsibilities 10

3 Configuration identification 11

3.1 Identification rules 11

3.1.1 Identification rules of configuration items 12

3.1.2 Identification rules of SOUPs 13

3.1.3 Identification rules of documents 14

3.1.4 Identification rules of a media 16

3.2 Reference configuration identification 16

3.3 Configuration Baseline Management 16

4 Configuration control 18

4.1 Change Management 19

4.1.1 Requesting Changes 19

4.1.2 Evaluating Changes 20

4.1.3 Approving or Disapproving Changes 20

4.1.4 Implementing changes 21

4.1.5 Closing the CR 22

4.2 Interface Management 22

4.2.1 Hardware Interface 22

4.2.2 Software Interface 22

5 Configuration support activities 22

5.1 Configuration Status Accounting 22

5.1.1 Evolutions traceability 23

5.1.2 Setting up Configuration status 23

5.1.3 Configuration status reporting 23

5.1.4 Configuration status records storage 23

5.2 Configuration audits 24

5.2.1 Baseline Audit 24

5.2.2 Software Configuration In-Process Audit 24

5.2.3 Functional Configuration Audit 24
5.2.4 Physical Configuration Audit 25
5.3 Reviews 25
5.3.1 Baseline Review 25
5.3.2 Branch Review 25
5.4 Configuration management plan maintenance 25

1 Identification

This document aims to provide a completed and detailed Software Configuration Management Plan for the project.

1.1 Document overview

This document contains the software configuration management plan for the SC Project.

1.2 Abbreviations and Glossary

Definitions for configuration management terminology can be found in the publication ANSI/IEEE Std 610.12-1990 IEEE Standard Glossary of Software Engineering Terminology. The following section also defines other project-related and frequently used terminology during development of SC project.

1.2.1 Abbreviations

CCB	Change Control Board. See also SCCB.
CI	Configuration Item
CM	Configuration Management
CMDB	Configuration Management Database See also SCMDB.
CMP	Configuration Management Plan
CMS	Configuration Management System. See also SCMS.
CR	Change Request
ID	Identification
IEEE	The Institute of Electrical and Electronics Engineers
PM	Project Manager
RFC	Request for Change
SC	SeniorConnect
SCCB	Software Change Control Board
SCI	Software Configuration Item
SCM	Software Configuration Management
SCMDB	Software Configuration Management Database
SCMP	Software Configuration Management Plan
SCMS	Software Configuration Management System
SOUP	software of unknown (or uncertain) pedigree (or provenance)
SQA	Software Quality Assurance
STD	Standard
QA	Quality Assurance

1.2.2 Glossary

Audit - An independent examination of a work product or set of work products to assess compliance with specifications, standards, contractual agreements, or other criteria. [IEEE-STD-610]

Baseline - (1) A specification or product that has been formally reviewed and agreed upon, that thereafter serves as the basis for further development, and that can be changed only through formal change control procedures. (2) A configuration identification document or a set of such documents formally designated and fixed at a specific time during a configuration item's lifecycle. Baselines, plus approved changes to those baselines, constitute the current configuration identification. For configuration management there are three baselines as follows:

a) Functional baseline. The initial approved functional configuration. b) Allocated baseline. The initial approved allocated configuration. c) Product baseline. The initial approved or conditionally approved product configuration identification. [IEEE-STD-610]

Configuration - (1) The arrangement of a computer system or component as defined by the number, nature, and interconnections of its constituent parts. (2) In configuration management, the functional and physical characteristics of hardware or software as set forth in technical documentation or achieved in a product. [IEEE-STD-610]

Configuration control - An element of configuration management, consists of the evaluation, coordination, approval or disapproval, and implementation of changes to configuration items after formal establishment of their configuration identification. [IEEE-STD-610]

Configuration control board - A group of people responsible for evaluating and approving or disapproving proposed changes to configuration items, and for ensuring implementation of the approved changes. [IEEE-STD-610]

Configuration identification - (1) An element of configuration management, consisting of selecting the configuration items for a system and recording their functional and physical characteristics in technical documentation. (2) The current approved technical documentation for a configuration item as set forth in specifications, drawings, associated lists, and documents referenced therein. [IEEE-STD-610]

Configuration item (CI) - An aggregation of hardware, software or both, that is designated for configuration management and treated as a single entity in the configuration management process. [IEEE-STD-610]

Configuration management (CM) - A discipline applying technical and administrative direction and surveillance to identify and document the functional and physical characteristics of a configuration item, control changes to those characteristics, record and report change processing and implementation status, and verify compliance with specified requirements. [IEEE-STD-610]

Configuration status accounting - An element of configuration management, consisting of the recording and reporting of information needed to manage a configuration effectively. This information includes a listing of the approved configuration identification, the status of proposed changes to the configuration, and the implementation status of approved changes. [IEEE-STD-610]

Document - (1) A medium, and the information recorded on it, which generally has permanence and can be read by a person or a machine. Examples in software engineering include project plans, specifications, test plans, user manuals. (2) To create a document as in (1). (3) To add comments to a computer program. [IEEE-STD-610]

Documentation - (1) A collection of documents on a given subject. (2) any written or pictorial information describing, defining, specifying, reporting, or certifying activities, requirements, procedures, or results. (3) The process of generating or revising a document. (4) The management of documents, including identification, acquisition, processing, storage, and dissemination. [IEEE-STD-610]

FCA - A Functional Configuration Audit (FCA) examines the functional characteristics of the configured product and verifies that the product has met the requirements specified in its Functional Baseline documentation approved at the Preliminary Design Review (PDR) and Critical Design Review (CDR). It has to do more with systems engineering and program management that official auditing. The FCA is a review of the configuration item's test and analysis data to validate the intended function meets the system performance specification. The FCA is normally performed prior to Low-Rate Initial Production (LRIP) and prior to or in conjunction with a Physical Configuration Audit (PCA). A successful FCA typically demonstrates that Engineering and Manufacturing Development product is sufficiently mature for entrance into LRIP. A FCA may also be conducted concurrently with the System Verification Review (SVR).

Hardware - Physical equipment used to process, store, or transmit computer programs or data. [IEEE-STD-610]

Interface - (1) A shared boundary across which information is passed. (2) A hardware or software component that connects two or more other components for the purpose of passing information from one to the other. (3) To connect two or more components for the purpose of passing information from one to the other. (4) To serve as a connecting or connected component as in (2). [IEEE-STD-610]

Revision - Each one of the different instances in time of a configuration item throughout its lifecycle. A revision implies a change in either the content or the attributes of a configuration item (e.g.: deleted, branched, etc.). Revisions of a configuration item are usually identified with consecutive incremental integer numbers. In opposition to versioning, revisions may not have any special meaning.

Software - Computer programs, procedures, and possibly associated documentation and data pertaining to the operation of a computer system. [IEEE-STD-610]

Software life cycle - The period of time that begins when a software product is conceived and ends when the software is no longer available for use. The software life cycle typically includes a concept phase, requirements phase, design phase, implementation phase, test phase, installation and checkout phase, and sometimes, retirement phase. [IEEE-STD-610]

SOUP – The term stands for Software of Unknown (or uncertain) Pedigree (or provenance), often used in the context of safety-critical and safety-involved systems such as medical software. SOUP is software that has not been developed with a known software development process or methodology, or which has unknown or no safety-related properties. [wikipedia]

System - A collection of components organized to accomplish a specific function or set of functions. [IEEE-STD-610]

Software Version ID - All Software Configuration Items should contain explicit and immutable version IDs that can be used to identify the exact version of each configuration item. Releases should also have an explicit and immutable version ID. NOTE: Software Configuration Audits make use of Version IDs to confirm that the correct, approved software configuration items are being used.

Traceability - (1) The degree to which a relationship can be established between two or more products of the development process, especially products having a predecessor- successor or master-subordinate relationship to one another; for example, the degree to which the requirements and design of a given software component match. (2) The degree to which each element in a software development product establishes its reason for existing; for example, the degree to which each element in a bubble chart references the requirement that it satisfies. [IEEE-STD-610]

Versioning - Versioning is a software configuration management function that identifies a specific revision of one or more configuration items for a specific purpose, such as a release of the software product to an external group or the identification of a specific baseline.

1.3 References

1.3.1 Project References

#	Document Title
[R1]	Software Release Management Plan
[R2]	Software Change Management Plan

1.3.2 Standard and regulatory References

#	Document Identifier (E-ISBN)	Document Title
[STD1]	978-0-7381-0391-4	ANSI/IEEE Std 610.12-1990 IEEE Standard Glossary of Software Engineering Terminology
[STD2]	978-0-7381-7232-3	IEEE Std 828TM-2012 IEEE Standard for Configuration Management in Systems and Software Engineering
[STD3]	N.A	SEM-0302 Software Configuration Management Plan

1.4 Conventions

Conventions for this CMP in the following section are adapted from the publication IEEE Std 828TM-2012 IEEE Standard for Configuration Management in Systems and Software Engineering.

CM shall ensure a consistent naming convention is designed to enable each CI to be uniquely named. CM shall ensure that the naming convention takes into account the need to accommodate multiple versions of items that may arise from the creation of a number of different baselines.

CM shall ensure that the naming scheme is extensible and capable of accommodating configuration items originating from third party suppliers.

NOTE1—This activity includes such items as reference implementations, hardware prototypes, and COTS, subcontracted, and supporting software.

NOTE 2—The design of the naming system should consider efficient storage, retrieval, tracking, reproduction, and distribution of CIs.

2 Organization

The organization identifies the major activities and corresponding responsibilities for SCM planning and management. It also describes the roles of individuals and groups that are involved in the SCM process.

For this SC project, there is no separated SCM team for SW project because of the limited team size. All SCM processes will be under the supervision and enforcement by the SW SQA team and the Project Manager.

The software configuration organization includes the personnel for software change management and release management.

QA Manager performing as a change manager takes charge of supervising the change control procedure and Release Manager is responsible for the release procedure, while Project Manager oversees both procedures and participates in major decision-making processes. Developers and engineers take part in both procedures and assist relevant tasks.

Software configuration management occurs when a change is requested to apply to a baselined item. After a configuration item becomes baseline, any changes or modifications made to it need to go through the change control process. Every member inside the project team, except for the developers, can identify and submit a change request.

The QA team, here performing as the SCM team, should collaborate with the developers to evaluate the change request and report the inspection result to the PM. With reference to the inspection materials, the PM should either approve or reject the change request. For significant changes, the company management may also participate in the decision-making process to achieve an optimal result. If the change request is approved, the front-end and back-end engineers should implement the change, conduct testing, integration and validation under the supervision of the developer lead. When a successfully validation result is reached, the process should be handed over the release manager to prepare for future release.

The Project Manager should also control and manage the release processes. The related major responsibilities are: drafting the release plan, identifying and tagging correct version of the configurations items to be included in a release, and overseeing the release preparation and implementation process.

At the same time, the QA team is responsible for the quality control of the SCM processes. Besides, the QA team should conduct planned SCM process reviews and audits.

Moreover, the PM need to review SCM management process in the periodic Management Review.

The detailed activities and responsibilities are explained in part 2.1.

2.1 Activities and responsibilities

The activities and responsibilities are explained in the following table.

Table 1 Responsibility Matrix-1

Activities when setting up the project	PM	QA Manager	QA Engineer	Lead Developer	Engineers	Release Manager
Identify the configuration items		X	X			
Install the bug repository tool and set up the database		X		X		
Install the software configuration repository tool and set up the database		X		X		
Manage and structure the reference space		X				
Define the configuration processes		X	X			

Activities during the project lifecycle						
Export components for modification, test or delivery	X	X				
Set under control validated components		X	X			
Create version, write version delivery document	X	X	X	X	X	X
Approve reference configurations	X	X				
Verify version to be delivered and authorize deliveries	X	X				
Backup spaces		X	X			
Do configuration audits			X			
Inspect configuration records		X	X			
Archive reference version		X	X			
Management activities						
Manage versions and archives		X	X			
Manage configuration records		X	X			
Produce reports and statistics	X	X	X			
Manage reference space and its access control list		X				
Manage spaces backup and archive media		X	X			
Manage quality reports		X	X			

2.1.1 Decisions process and responsibilities

The responsibilities during reviews, audits and approvals are explained in the following table group.

Table 2 Responsibility Matrix Group-2

At the end of an activity of the project

Activities	PM	QA Manager	QA Engineer	Lead Developer	Engineers	Release Manager
Do a configuration freeze		X				

Activities	PM	QA Manager	QA Engineer	Lead Developer	Engineers	Release Manager
Present a configuration state of the components impacted by the activity		X	X			
Present a documentation state of the components impacted by the activity		X	X			

During a configuration management process audit:

Activities	PM	QA Manager	QA Engineer	Lead Developer	Engineers	Release Manager
Do the configuration management process audit	X					
Present the records of the configuration management process		X	X			
Present the quality records of the configuration management process		X				
Present the records of the documentation management process		X	X			

3 Configuration identification

3.1 Identification rules

Configuration Item Identifiers are used to label all of the CIs that make up a particular grouping such as an application release, a project development phase or documentation changes.

This identification scheme preserves all of the files that are used to create each release and exactly which versions of those files were used. This scheme works for the application installations and then for subsequent upgrades.

Identifiers are used to label the documentation deliverables in a project. For instance, at the end of the system design stage, all of the approved deliverables will be labeled and preserved for future reference. After the completion of the project, many of the deliverables will need to be updated to reflect changes to the application. Those deliverables are assigned identification labels so that their current state can be identified and preserved for future reference.

The identification criteria in the following section are adapted from Software Configuration Management Plan Template by Department Of Information Technology, State of Michigan.

3.1.1 Identification rules of configuration items

3.1.1.1 Identification of a configuration item

The following table shows how identifiers are assigned to files and baselines.

Examples 1 shows sample baseline release configuration identification labeling schemes, which may be explained immediately in the following section.

Table 3 CI Identifier Examples

Configuration Items	Identification Scheme
Project Proposal Review	PP_PPR_FUN.7

Configuration Item Identifiers are used to label all of the CIs that make up a particular grouping such as an application release, review meetings and design stages. This identification scheme preserves all different versions of the files after they have become a component of a baseline.

Identifiers are used to label the documentation deliverables in a project. At the end of each life cycle stage, all of the newly approved deliverables will be labeled and preserved for future reference. After these deliverables become a component of a baseline, any changes to them will need to go through the formal change control procedures. If the CR regarding one document has been approved, the changes will be committed. The modified deliverable will be assigned with a new identifier and saved separately. All versions of a deliverable (in baseline) are bookkept with different version numbers so that each version is individually accessible, and their current state can be identified and preserved for future reference. All versions of a CI are organized into a folder named with the full name of that CI.

The following table displays the definition rules for CI identifiers. In the matrix, literal text is enclosed in quotes. The label content descriptions follow the table.

Table 4 CI Identifier Rule

Baseline	Version Label Contents						
Functional	CI Code	'_'	Sub Code	'_'	'FUN'	'.'	Revision Number …
Design					'DES'		
Batch					'BAT'		
Checkout					'CHE'		
Production					'PRO'		
Not Baseline					Date Stamp		

CI Code A project must assign, to each CI, an uppercase alphanumeric code no longer than five characters. The CI codes for the identified CIs for SW project are tabulated in the tables in the following several sections according to the types of the CI items.

Sub Code For CIs like Review Forms, which has multiple documents, sub code is used to distinguish between these different documents. If a CI does not have multiple parts, sub code is not needed. This sub code is optional if the situation is not applicable according to Table 6.

Revision Number All the CIs are given an integer revision number. The initial document is given a revision number of one. Each subsequent change to this document increases its revision number by one.

Baseline Identifiers The baseline identifiers are used to label which baseline does this document belongs to. The document will be labeled with the identifier of the base line, which it is first been identified as a component of a baseline. For example, the SRS will be labeled with the baseline identifier of 'FUN'. Though it will be a component of subsequent design and batch baselines as well, it is labeled with 'FUN' as it is first identified as a component of the functional baseline.

Date Stamp Some deliverables or documents, like Review Meeting Minutes, are not tied to any of the baselines. These documents or deliverables, instead of tagged with baseline identifiers, are tagged with a time stamp of the form MM-DD-YY. MM-DD-YY is the month-day-year on which the session took place. For example, for a review meeting held on Oct, 20th, 2015, its Review Meeting Minutes will receive a time stamp as 20-Oct-15.

Baseline Baseline is used to categorize each CI, and will be explained in detail in part 3.3. The baseline for each CI should refer to the following table.

Table 5 Baseline Contents

Baseline	Contents
Functional	Use Case Model Software Requirement Specification Project Plan SQAP CMMI Level 2 Process Definition SCM Plan Test Plan Risk Management Plan
Designed	ERD Diagram Conceptual Model Dynamic Models
Batch	Source Code Makefile Unit Test Suites Test cases and Scripts (Both black and white)
Checkout	Integration Test Suites
Production	System Test Suites Acceptance Test Suites User documentation and tutorials

3.1.1.2 Version number of a configuration item

The attribution of a version number is a prerequisite to any delivery of any configuration item. This number shall be incremented before a new delivery, if the product or its documentation were modified.

Every time when the modified CI has reached a stable state and is worthy of kept into record of the changes, an update revision index request should be reported to PM. After PM approved with the request, a new revision index should be applied. In other words, the revision index shall be incremented by one before the diffusion of a modified document.

3.1.2 Identification rules of SOUPs

3.1.2.1 Identification of a SOUP

The definition rules for identification of a SOUP is to take the ID used by the SOUP manufacturer. If there is no ID available, the ID should be taken referring to part 3.1.1.1. The CI code should refer to part 3.1.3.1. If no reference is applicable, use "SOUP" instead.

3.1.2.2 Version number of a SOUP

The definition rules for version number of a SOUP is to take the version number used by the SOUP manufacturer. If there is no version number available, the version number should be taken referring to part 3.1.1.2.

3.1.3 Identification rules of documents

3.1.3.1 Description of documents identifiers

The definition rules for identification of documents refer to part 3.1.1.1.

The following table shows the CI CODE for documents.

Table 6 Document CI Code

Document Identifiers	Document CI description
PPRO	Project Proposal
PP	Project Plan
CMP	Software Configuration Management Plan
CHMP	Software Change Management Plan
RMP	Risk Management Plan
CPD	CMMI level 2 Process Definition
UCD	Use Case Description
SQAP	Software Quality Assurance Plan
SRS	Software Requirements Specification
FUD	Functional Design
DMA	Design For Main
PROT	High-fi Prototype
TSTP	Test Plan
TSAR	Test Type Approach and Report (multiple)
SYD	System Design
TSTC	Test Case (multiple)
RP	Release Plan and Release Notes
MR	Maintainability Report
CR	Change Request
DTL	Defect Tracking Log (or equivalent)

CODE	C/JAVA Code (Example)
IMAGE	Graphics/Images
RMM	Review Meeting Minutes (multiple)
RF	Review Forms (multiple)
RCL	Review Checklists
AU	Audit
Sub Code	CI Parts
PPR	Project Plan Review
SRR	Software Requirement Review
SDR	Software Design Review
TRR	Test Readiness Review
POR	Post-project Review
CSR	Class Diagram & Sequence Diagram Review
CPDR	CMMI Process Definition Review
UCMR	Use Case Model Review
UCDR	Use Case Description Review
MGTR	Management Review
PROTR	Prototype Review
RPR	Release Plan Review
SQAPR	SQAP Review
TSTPR	Test Plan Review
RMPR	Risk Management Plan Review
UCMR	Use Case Model Review
SCMPR	SCM Plan Review
BR	Baseline Review

3.1.3.2 Definition and evolution of the revision index

Every time when the modified document has reached a stable state and is worthy of kept into record of the changes, an update revision index request should be reported to PM. After PM approved with the request, a new revision index should be applied. In other words, the revision index shall be incremented by one before the diffusion of a modified document.

3.1.4 Identification rules of a media

The definition rules for version number for media should refer to part 3.1.3. The CI code should refer to part 3.1.3.1. If no reference is applicable, use "MEDIA" instead.

3.1.4.1 Internal identification

The definition rules for version number for media should refer to part 3.1.3. The abbreviated name should refer to part 3.1.1.1 and 3.1.3.1. If no reference is applicable, use a self-explanatory name instead.

3.2 Reference configuration identification

Each reference configuration is defined by the combination of an identifier, its content listed in the corresponding Version Delivery Description document, and the acceptation or validation reviews associated to the building of the reference configuration.

A reference configuration is established for each design review and each test review of the project.

3.3 Configuration Baseline Management

The following baseline identification guidelines are adapted from IEEE Std. 828 Software Configuration Management and Configuration Management Plan. All SW project CIs will have at least five baselines: functional, design, batch, check-out, and production. The contents are cumulative; i.e., each subsequent baseline contains all the components of the baselines before it.

The functional baseline represents the project's requirements and processes. All documents containing project requirements or describing the project-unique development processes are labeled as functional baseline components. At a minimum, the functional baseline is embodied in the Use Case Model, Software Requirements Specification and Project Plan. A new functional baseline is established whenever a change in the project's requirements or development process occurs. They include (but are not limited to):

- Modification or addition of a project-unique development process
- Granting a deviation or waiver from the requirements
- The customer modifies a requirement
- A new feature is requested and approved

Design may proceed once the functional baseline is established. The design baseline includes the complete set of design documents. The first design baseline will support the implementation and is established when all the relevant designs have been reviewed and approved by the PM. Subsequent design baselines contain design additions or modifications which have been reviewed and approved. These changes include:

- Designs describing features for a subsequent incremental build
- Design changes in response to requirements changes
- Design modifications which evolve from implementation experience

The batch baseline contains code which is ready for integration testing in a non-real-time environment. All new development and unit testing occurs prior to establishment of a batch baseline. Source code does not come under formal CM control until it associated with a batch baseline. The batch baseline includes all the unit test suites created during new development. The first batch baseline can be any combination of unit-tested sub-systems that have undergone a code review and are ready for integration. Subsequent batch baselines are established with the addition of new unit- tested subsystems or major modifications to existing subsystems. Major modifications need regression testing using the appropriate unit and integration test suites.

The checkout baseline is established when a batch baseline is ready for testing and acceptance in the real-time environment (a.k.a. system testing). It includes all the integration test suites used in verifying the batch baselines. There is no one-to-one correspondence between batch and checkout baselines. The Project Manager determines whether a batch baseline is promoted to a checkout baseline. Checkout baselines may also be established when minor modifications are made to source code attached to a checkout or production baseline.

A production baseline is created after a checkout baseline undergoes a functional configuration audit (FCA). Further information of FCA can be referred to Part 1.2.2. The FCA embodies the process of accepting the checkout baseline for production. There is no one-to-one correspondence between checkout and production baselines. Production baselines strictly represent software configurations intended to run an experiment. The first production baseline is the first incremental build required to run the first experiment. The production baseline includes all the real-time tests suites used in the validation of the checkout baseline and any required documentation.

The following table shows the CIs that are included in this SCM.

Table 7 Configuration Baseline Items

Configuration Items	Identifier Form	Responsible for placing item under control	When item is put under control
Project Proposal	PPRO	Project Manager/ Artist	Initiation & Planning Stage Exit
Project Plan	PP	Project Manager	Initiation & Planning Stage Exit
Software Configuration Management Plan	SCMP	QA Manager/QA Engineer	Initiation & Planning Stage Exit
Risk Management Plan	RMP	QA Engineer	Initiation & Planning Stage Exit
CMMI level 2 Process Definition	CPD	QA Manager	Initiation & Planning Stage Exit
Use Case Description	UCD	QA Manager	Initiation & Planning Stage Exit
Software Quality Assurance Plan	SQAP	QA Engineer	Initiation & Planning Stage Exit
Project Plan Review Checklist	RCL-PPR	QA Manager/QA Engineer	Initiation & Planning Stage Exit
Configuration Management Log	CML	QA Manager/ Project Manager	Initiation & Planning Stage Exit
Software Requirements Specification	SRS	Lead Developer	Requirements Stage Exit
Software Requirement Review Checklist	RCL-SRR	QA Manager/QA Engineer	Requirements Stage Exit

Functional Design	FUD	Lead Developer/ Artist	Functional Design Stage Exit
High-fi Prototype	PROT	Artist	System Design Stage Exit
Test Plan	TSTP	QA Engineer	System Design Stage Exit
Test Type Approach and Report (multiple)	TSAR	QA Manager/QA Engineer	System Design Stage Exit
System Design	SYD	Lead Developer/ Artist	System Design Stage Exit
Test Case (multiple)	TSTC	QA Manager/QA Engineer	System Design Stage Exit
Software Design Review Checklist	RCL-SDR	QA Manager/QA Engineer	System Design Stage Exit
Release Plan and Release Notes	RP	Project Manager	Implementation Stage Exit
Change Request	CR	QA Manager/ Project Manager	Construction Stage Exit
Defect Tracking Log (or equivalent)	DTL	QA Manager/QA Engineer	All Stages
C/JAVA Code (Example)	CODE	Lead Developer/ Artist	Initial unit test
Graphics/Images	IMAGE	Author of them	Initial unit test
Test Readiness Review Checklist	RCL-TRR	QA Manager/QA Engineer	Test Stage Exit
Maintainability Report	MR	Leader Developer/Artist	Test Stage Exit
Post-project Review Checklist	RCL-POR	QA Manager/QA Engineer	Post Project
Review Meeting Minutes (multiple)	RMM	QA Manager/QA Engineer	All Stages
Review Forms (multiple)	RF	QA Manager/QA Engineer	All Stages
Audit Forms	AU	QA Manager/QA Engineer	All Stages

4 Configuration control

In this section, the process and methodology used for configuration control are described, which includes change management and interface management. For change management detailed process and relevant forms, a separate document of SeniorConnect Project Change Management Plan (SCM Plan). can be referred.

4.1 Change Management

The configuration control activities request, evaluate, approve/reject, and implement changes to CI baselines.

Any member of SC project team except for developers can submit a CR to the QA Manager (here acts as CMM). The QA Manager will then analyze the request, using the current baseline documents by himself or assign the analysis task to the QA Engineer. The QA Manager will consult the request with the lead developer and the release manager if applicable. He will base his decision on how severely the change will impact the entire system. Once his analysis is completed, he will then submit the CR together with the analysis result to Change Control Board (CCB) for further decision. CCB will make the decision to approve or disapprove. If the CR is approved, the PM will assign this change to the lead developer who will further assign this task to appropriate developers. The assignee will implement the changes. The QA team should record the CR, its result, track the CR and conduct regular status accounting and audits. The State-Transition Diagram for Change Request Statuses can be referred at Appendix D of SeniorConnect Project Change Management Plan (SCM Plan).

Change requests (CRs) can be submitted against functional, design, checkout, and production baselines. Batch baselines are considered works-in-progress and forcing change authorization would only hinder the development process. The QA Manager will ensure that complete records are received at each stage. Each CR will contain a status field that indicates it current progression through the lifecycle. The statuses with descriptions are listed in the table below.

Status	Description
Approved	The CCB decided to implement the request and allocated it to a specific future build or product release. The CCB Chair has assigned a Modifier.
Canceled	The Originator or CCB decided to cancel an approved change. Details should be provided with canceller's signature when a CR is cancelled.
Change Made	The Modifier has completed implementing the requested change.
Closed	The change made has been verified (if required), the modified work products have been installed, and the request is now completed.
Evaluated	The Evaluator has performed an impact analysis of the request.
Rejected	The CCB decided not to implement the requested change.
Submitted	The Originator has submitted a new issue to the change control system.
Verified	The Verifier has confirmed that the modifications in affected work products were made correctly.

4.1.1 Requesting Changes

Project change requests (CRs) may solicit requirements modifications, requirements waivers, functional enhancements, design modifications, additional documentation, or defect corrections. CRs may be submitted by stakeholders of SC project team. This section specifies the procedures for requesting a change to a baselined SCI and the information to be documented for the request. As a minimum, the information recorded for a proposed change shall contain the following:

- The name(s) and version(s) of the SCIs where the change is desired
- Date of request
- Indication of degree of impact to user and indication of probable frequency of occurrence of such impact
- The need for the change (e.g., correct a defect, modify functionality, add new functionality)
- Description of the requested change
- If a defect is being reported: description of steps needed to replicate the occurrence of the defect.
- Additional information, such as urgency, priority or classification, may be included to clarify the significance of the request and to assist in its analysis and evaluation.

The CR form template can be found in Appendix A. All CRs will be submitted electronically to a defect tracking software product. The requester will need to fill in all the required fields in the form before submitting it.

Once a CR is received by the QA Manager, the QA Manager (here acts as CMM) assigns a CR number to the request. The naming convention of a CR should follow the specification in Section 3.1 of this SCM Plan. A number after the CI code will uniquely identify the CR. The CR ID is incremental. A CR achieves the **'Submitted'** state when it receives this number. Other information, such as status and disposition, shall be recorded for change tracking as well. From this point forward, the submitter will be notified during the weekly meetings of all status changes to the CR. It shall take no longer than two weeks to address a submitted CR.

4.1.2 Evaluating Changes

The analysis of a change is required to determine the impact of the proposed change and the procedures for reviewing the results of the analysis. A CR reaches the **'Reviewed'** state when the QA team (here act as CMM) actively evaluates it. Changes should be evaluated according to their effect on the deliverable and their users and impact on project resources.

The QA team will formally meet at least once two weeks to review all outstanding CRs. The QA Manager will divide the outstanding CRs between himself and the QA Engineer for review. The assignment guarantees that at least one person will examine the CR; but more than one person can examine the same CR.

The reviewer will analyze the CR form focusing on its impact to project cost or schedule and uncover any technical risks. If the CR identifies a defect, the reviewer will verify the reproducibility of the defect. The reviewer will record his findings on Part B of the CR form and mark the CR as **'Reviewed'**. Any additional costs must be approved by the PM.

At the formal meeting, each CR will be addressed as the agenda. The assigned reviewer will make a recommendation; then the floor will be open for debate. The PM decides when to end the debate; he then will announce his decision on the CR (section 5.3 of this plan). The CRs that do not reach a decision in this meeting will be discussed again during the next meeting.

4.1.3 Approving or Disapproving Changes

A Configuration Control Board (CCB) may be an individual or a group. For SC project, the PM, as an individual, will take the responsibility of the CCB. The PM will make decisions regarding all the change requests submitted.

The PM will render a decision on a CR when he feels he has sufficient information. Part

C of the CR form (Appendix A) records this decision. The PM can make one of four decisions:

'Approve', 'Reject', 'Defer', or 'Duplication'.

The result of each decision follows:

Approve The CCB Chairman now has another choice which is to defer the CR. The CCB Chairman may contact the submitter for suggested modifications to the CR form before approving a request. The submitter is required to indicate that he concurs with the modifications. For this purpose, the submitter is required to sign the CR form on the required signature field.

Reject The PM must record a reason for the rejection in Part C of the CR form.

Defer The PM can approve a CR and then defer it. The PM must state his reason for deferring the CR on the CR form (part C). A deferred CR has the lowest priority; it is performed on a time-available basis only. They contain noncritical changes that enhance usability of the simulation or fix very minor annoyances. A deferred CR can be revived under these conditions:

a) A developer volunteers to implement the CR and the PM concurs.

b) The PM assigns the CR as a training exercise for a new employee.

c) The project elevates it to a required change.

Once activated, the deferred CR will be assigned to an assignee. Any deferred CRs remaining when the project is closed are immediately marked **'Rejected'**. The PM can use 'project closed' as the reason for rejection.

Duplication It has been determined that the requested change is a duplicate of another change request. Two forms of duplication exist: identical and functional. Identical duplication indicates that the change request is a near copy of a previous change request. Functional duplication means that the change request represents a different symptom of the same defect or a different description on a new feature or enhancement reported in a previous change request. The change request is then marked as 'Duplicate'. The submitter is added to the notification list of the "original" CR. A pointer to the original CR will be added to the duplicate.

4.1.4 Implementing changes

The developer is granted change authorization when he receives an approved CR. The programmer creates a "developer baseline" which contains the CR implementation. The developer baseline will be based on the baseline against which the CR was logged (usually a checkout baseline). Requirements changes will result in a new functional baseline.

Likewise, design changes result in a new design baseline. Source code changes will result only in a new batch or checkout baseline; only small changes should move directly to a checkout baseline.

When the programmer has tested his changes and is satisfied with the results, the developer will fill out part B of the Change Request Validation Form. It describes the actions taken to make the change and the actual impact on cost, schedule, and resources.

The names and versions of the modified CI components (if any) shall be recorded with the Validation Form. The developer will then contact the QA team to schedule a review.

The review process will be documented in other parts of Change Request Validation Form. It shall record down the associated CR, review date, review notes, and review result (P/NP). Once the change passes the review, the change request is ready for closure.

4.1.5 Closing the CR

A rejected CR is considered immediately closed and no further action is taken on it. The closure procedures for an approved CR begin when the change passed the CR review conducted by the QA team. Requirements and design changes are merged directly into the appropriate baseline. Source code changes are folded into a new batch baseline unless the PM rules that the change may be used to create a new checkout baseline. Once the change is merged, the QA team will tag the form and CR as closed. The QA Manager will announce the new baseline to all project participants in an e-mail which includes a CR summary. This informs all project developers, who are using the batch or checkout baselines for testing, that a new change has been added.

4.2 Interface Management

The project development interacts with both hardware and software interfaces. The purpose of interface control is to coordinate changes of CIs because of changes in the external environments

4.2.1 Hardware Interface

Android devices might be customized on different hardware device vendor and therefore, for hardware device from different hand phone vendors, program source code might be modified to cater to unique requirements of the hardware. However, in our SC project, the framework "Ionic" that we use can help us take care of the hardware interface to run on different devices.

4.2.2 Software Interface

Updates and bug fixes of operating system (Android and iOS in our case) would happen quite frequently, and the updates in operating system may result in change of UI style and usage of system API. This interface is out of control of development team, and the project on the specific platform will be affected. In this case, a CR should be properly raised to address this change, and normal process of SCM should be followed.

5 Configuration support activities

5.1 Configuration Status Accounting

Status of each CI should always be properly kept track of. Upon establishment of baseline, each change should have a corresponding CR, indicating necessary information for initiation of the change. When CR is approved, the change should not be made on the baseline copy, but should be made on a duplicate of the baseline copy. Only when the change is verified can the change be merged to baseline copy. This is the time when a change is formally recognized.

Digital format should always be used for recording content of changes, and paper recording should be avoided in any stage of status accounting. Centralized information storage should be used to book keep configuration change data. For each CI that involves changes, the following information, complementing CR of the change, should be recorded in the system and in the document where necessary.

- The person who made the change.
- The date the change was made.
- The reason for the change in a brief format.
- The corresponding CR for which this change was made.
- Detailed changes made to the CI.

Some different rules apply to some specific CIs.

- Source code: for each change, provided the affected components/subsystems in the changes made to CI.

- Documents: for each change, update the change history in the document for easy reference.

Read access to CIs, CRs, and status and changes of all CIs should be open to all team members. However, write access to configuration system will only be given when CCB approves and verifies the change.

5.1.1 Evolutions traceability

The traceability of modifications of items given their types:

- Document: The modification sheet number identifies the origin of the modification. The modified paragraphs in the document are identified, if possible, by revision marks. For this project, Wiki site is used to record versions of documents.
- Source file: The software configuration management tool records, for each source file or group of source files, a comment where is described the modification. For this project, SVN is used to control source code by recording changes and reasons between versions.
- Other Configuration item: The Version Delivery Description of the configuration item identifies the modification sheet included in the current version.

The modification sheet describes the modifications done to the components with enough precision to identify the modified parts.

5.1.2 Setting up Configuration status

The software configuration manager sets up the state of all versions and of each configuration article with:

- The label,
- The version number,
- The creation date of the Version Delivery Description (VDD) written by software configuration manager. For this project, QA manager acts as the software configuration manager.

5.1.3 Configuration status reporting

Status reporting is to summarize and convey the status of all CIs, giving the status of current project configuration. The purpose of this report is to give QA team and PM an overview of current project status, and to determine the proper time for a release.

Given current system in use, the status report will be generated using built-in information in SVN, change history in each file, and related information in CR for which the change is made. Therefore, this report consists of automated effort and manual effort.

The content of the report consists of the following:

- Versions of all CIs for last baseline
- CIs involved since last baseline
- Changes made since last baseline, and person who made each change
- CRs for which changes have been made since last baseline, and description of CRs
- CRs that have been changed/voided since last baseline
- CRs that remain open, and description of CRs

Status Report should be generated each week on Friday after 7pm. All status report must be archived for future reference.

5.1.4 Configuration status records storage

The configuration status records are stored in a configuration folder, which contains:

- The requests sorted by record number,

- The software documents,
- The VDD's,
- The configuration states sorted chronologically.

5.2 Configuration audits

Configuration processes should be audited to ensure the quality assurance. This section describes how peer review audits and formal audits are made to assess the compliance with the CM Plan. The audit includes baseline audit, functional configuration audit, software configuration audit.

5.2.1 Baseline Audit

If a baseline is reviewed and approved, the baseline should be updated and release notes should be generated. Baseline Audit includes:

- an examination of specific actions performed against the baseline
- identification of individuals involved in any action
- an evaluation of change within the baseline
- (re-)certification for approval
- Accounting
- metric collection
- comparison to another baseline

5.2.2 Software Configuration In-Process Audit

SCM In-process audit can be performed any time during project development, at each external release, and at least once per month. The purpose of this audit is to verify that

SCM plan, including but not limited to CI identification, CI changes, CI status accounting, and CM auditing, has been properly followed by the project team. This audit will be done by the QA team by assessing the existing content in SCM system and status of CIs.

Any findings from this audit should be properly recorded in the audit form in **Appendix**

B. The findings should be conveyed to SCM team and PM for improvement in future.

5.2.3 Functional Configuration Audit

FCA happened at the stage where a working/release baseline is transformed into production baseline, meaning the time when an external release is to be made. The purpose of FCA is to ensure all functional requirements and non-functional requirements are satisfied with the current status of CIs.

The FCA should be conducted with the following setting:

- FCA is conducted by SCM manager, and under the vision of PM
- The input of FCA should be ready. The input consists of requirements (both functional and non-functional) and test methods. Ensure each requirement is covered by at least one test.
- The first FCA should cover all requirements and therefore will take some time. Subsequent FCAs will focus on changes that have been made from last FCA.
- FCA should always be conducted before PCA if it's foreseeable that some changes will be made to CIs after FCA.

Any deviations from requirements discovered in FCA should be properly recorded in the review form in Appendix C. For each deviation, a CR should be raised if necessary to make sure CIs are in line with requirements.

5.2.4 Physical Configuration Audit

PCA is conducted as a complement of FCA. The purpose of PCA is to examine the actual configuration of CIs in the external release matches the content in the design documents. The input of PCA consists of all CIs and design documents.

Any deviations from requirements discovered in PCA should be properly recorded in the review form in Appendix C For each deviation, a CR should be raised if necessary to make sure CIs are in line with requirements.

5.3 Reviews

During the establishment of baselines, technical review should be conducted to provide analysis. The baseline review and branch

5.3.1 Baseline Review

When a baseline is to be established, a review should be conducted by PM, QA team and SCM team. This is to ensure the integrity of baseline and avoid problems related to defects in baseline CIs. If the baseline is approved, the baseline should be updated and release notes should be generated.

The checklist of baseline review is attached in Appendix B.

5.3.2 Branch Review

A development branch is a version that is under development, and has not yet been officially released. When a branch is to be created, a review should be conducted by PM, QA team and SCM team. This is to ensure the integrity of branch and avoid problems related to defects in branch CIs. If it is approved, the branch should be.

5.4 Configuration management plan maintenance

Implementation and adherence to SCM plan shall begin as soon as it is approved by

Project Manager, SCM team and made known to the whole project team.

The key events in the CM implementation phase are as follows:

- Establish the SCM team
- Specified other necessary and detailed CM procedures which complement this plan and obtain approval for the procedures from Project Manager
- Establish the project CR database and customize it for the CR form for the project if needed
- Establish the project repository for CIs

The abovementioned activities will create a solid environment for SCM and projects should be capable of reusing existing SCM product customizations.

Routine SCM activities:

- Status reporting: each Friday at 8pm
- SCM In-Process Audit: last day of each month if no SCM In-process audit has been conducted.

Appendix A-1 Change Request Form

SeniorConnect Project
Change Request Form

SUBMITTER - GENERAL INFORMATION (Part A)				
CR#				
Submitter Name				
Submitter Contact Number				
Submitter E-mail				
Product Name				
Product Version				
Request Title				
Brief Description of Request				
Date Submitted				
Date Required				
Priority	☐ Low	☐ Medium	☐ High	☐ Critical
Reason for Change				
Other Artifacts Impacted				
Assumptions and Notes				
Attachments or References	☐ Yes	☐ No		
	Link:			

INITIAL ANALYSIS (Part B)		
Hour Impact		
Duration Impact		
Schedule Impact		
Comments		
Recommendations		

CHANGE CONTROL BOARD – DECISION (Part C)				
Decision	☐ Approved	☐ Approved w/Conditions	☐ Rejected	☐ More Info
Decision Date				
Decision Explanation				
Conditions				

Appendix A-2 Change Request Validation Form

Change Request Validation Form

Part A: Change Request Validation Information			
Related Project Name:			
Related Change Request Number:			
Validator:		Date of Validation:	
Description of Validation Process:			
Part B: Changes Made by Previous Implementation Assignee			
	CI	Changes Made	
1			
2			
3			
4			
Part C: Detailed Validation of CIs			
	CI	Issue Description (if any)	Accept/Reject
1			
2			
3			
4			
Part D: Validation Result			
Decision of Validator on next status of related CR:			
Reason for Decision:			
Subsequent Implementation Assignee (if applicable):			

Estimated Date of Completion (if applicable):

Checked by:		**Approved by:**	
Validator	Signature: __________	Change Assignee	Signature: __________
		Project Manager	Signature: __________

Appendix B-1 Baseline Review Checklist

ID	Y/N	Aspect to check	Comments
1		All CIs for this baseline are present.	
2		All CIs have complete change history related with it.	
3		CIs are consistent in terms of versions used and referred to in other CIs.	
4		All CIs have been reviewed before and approved by Project Manager.	
5		Changes for CIs of versions to be included have been closed.	
6		Physical central repository is ready to accept the new baseline.	
7		Internal Release Notes are ready for generation.	
8		External Release Notes are ready for generation (if applicable)	
9		All other requirements for CIs and baseline release have been satisfied.	

Checked by:

QA Engineer	Signature: _________	QA Manager	Signature: _________

Approved by:

Project Manager	Signature: _________	SCM Manager	Signature: _________

Appendix B-2 CM Configuration Auditing and Review Forms

Process Audit Form			
General Information			
Auditor			
Date of Report			
Date of Audit			
Process/ Procedure Audited			
Issues			
	Classification	Issue Description	Comment
1			
2			
3			
4			
Audit Findings			
____ Process/Procedure Acceptable ____ Process/Procedure Conditionally Acceptable (Subject to satisfactory completion of action items listed below) ____ Process/Procedure Unacceptable			
Follow-up (if any)			
	Action Item	Assignee	Deadline
1			
2			

Corrective Action			
Disposition	*APPROVE*	*CANCEL*	*DEFER*

Checked by:		**Approved by:**	
SQA	Signature: __________	Project Manager	Signature: __________

Appendix C FCA & PCA Form

FCA and PCA Form				
Type of Audit: FCA / PCA				
Record				
ID	Related CI	Description of Error / Issue	Classification (**C, S, M, M'**)[2]	Recommended Action
1				
2				
3				
4				
5				

Follow-up			
ID	Action Item	Assignee	Related CR
1			
2			
Conclusion of Audit			
Does the baseline pass the audit? Yes / No (Delete where inappropriate)			

Checked by:			
Assignee	Signature: ________	QA Manager	Signature: ________
Approved by:			
Project Manager	Signature: ________	Project Manager	Signature: ________