SHELVD

Software Configuration Management Plan

Version 1.2

27/3/2024

TABLE OF CONTENTS

1 Identification

1.1 Document overview

This document contains the software configuration management plan of software shelvd.

1.2 Abbreviations and Glossary

1.2.1 Abbreviations

- SCM: Software Configuration Manager
- QA: Quality Assurance
- SRS: Software requirement specification
- VDD: version description document
- CM: Configuration Management
- CI: Configuration Item
- SOUP: Software of Unknown Provenance

1.2.2 Glossary

- Branch: a line of development that exists independently of another line, yet still shares a common history, and can be merged in the future. Often used as in concept of feature branching, where each independent branch works on a feature, before merging to the main branch after completion.
- Version: state of a configuration item at a well-defined point in time
- Variant: versions that coexist

1.3 References

1.3.1 Project References

#	Document Identifier	Document Title
R1	SRS	Software Requirements Specification

1.3.2 Standard and regulatory References

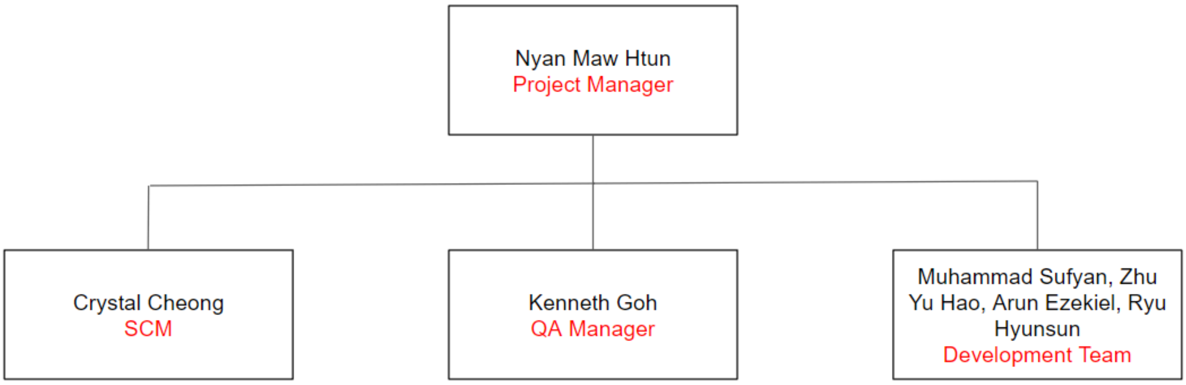
#	Document Identifier	Document Title
STD1	SCM1	IEEE 828-2012 - IEEE Standard for Configuration Management in Systems and Software Engineering
STD2	SSCM1	IEEE 828 Standard for Software Configuration Management Plans

1.4 Conventions

Typeface	Usage	Example
Bold	Emphasis, headers, titles	**Software Configuration Management**
Italics	Minor emphasis, file names, figures, tables	*Figure 1:*

2 Organization

The software configuration is managed by members of the project, with specific tools. Responsibilities are shared between:



2.1 Activities and responsibilities

Activities when setting up the project	**Person responsible**
Identify the configuration items	SCM
Install the bug repository tool and set up the database	SCM
Install the software configuration repository tool and set up the database	SCM
Manage and structure the reference space	SCM
Define the configuration processes	SCM

Activities during the project lifecycle	**Person responsible**
Export components for modification, test or delivery	SCM
Set under control validated components	SCM
Create version, write version delivery document	SCM
Approve reference configurations	Project manager
Verify version to be delivered and authorise deliveries	Project manager
Backup spaces	SCM
Do configuration audits	QA Manager
Inspect configuration records	QA Manager

Activities during the project lifecycle	Person responsible
Archive reference version	SCM

Management activities	Person responsible
Manage versions and archives	SCM
Manage configuration records	SCM
Produce reports and statistics	SCM
Manage reference space and its access control list	SCM
Manage spaces backup and archive media	SCM
Manage quality reports	QA Manager

2.1.1 Decisions process and responsibilities

Responsibilities during reviews, audits and approvals are listed below:

At the end of an activity of the project

Activities	Person Responsible
Do a configuration freeze	SCM
Present a configuration state of the components impacted by the activity	SCM
Present a documentation state of the components impacted by the activity	SCM

During a configuration management process audit:

Activities	Person Responsible
Do the configuration management process audit	Project Manager
Present the records of the configuration management process	SCM
Present the quality records of the configuration management process	Quality Manager
Present the records of the documentation management process	SCM

3 Configuration identification

Configuration identification is the process of:

- Systematically defining and cataloging all configuration items (CI) within a project
- Documenting unique identifiers, versions, and descriptions for each CI
- Recording relationships between different CIs
- Creating a comprehensive inventory for manageability and traceability
- Facilitating effective configuration management, change control, and status accordingly
- Providing a foundation for transparent and overseen changes throughout the SDLC

3.1 Identification rules

3.1.1 Identification rules of configuration items

3.1.1.1 Identification of a configuration item

The identification of configuration item is:

- XXX_Vm.n
    - where: "Vm.n a" is the version of the configuration item.
- Code
    - Source Code
- Documentation
    - Project Proposal
    - SRS
    - Quality Plan
    - Project Plan
    - Risk Management Plan
    - Test Plan
- System Designs
    - Use case models
    - Conceptual models
    - Software architecture
    - Software prototype
    - Unit Test
    - Coverage Test
- Specification
    - Project requirements specification
    - Unit Test

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.

The following are the rules for defining a version number:

- The form of the version number is MAJOR.MINOR.PATCH. Every time there is a change, the version needs to be increased.

- MAJOR releases:
  - Significant modification to business logic
  - influence the storing of personal data
  - major function removal
  - A new API that modifies the user interface is added.
- MINOR releases:
  - Modifications to current functionality without altering the user interface
- PATCH:
  - Compatibility issues
  - Grammatical issues
  - Improvement of performance

3.1.2 Identification rules of SOUPs

3.1.2.1 Identification of a SOUP

Take the SOUP provider's ID. As a result, developers will find mapping easier.

3.1.2.2 Version number of a SOUP

We will obtain the version number of a SOUP from the SOUP provider, just like we would with SOUP identification.

3.1.3 Identification rules of documents

3.1.3.1 Description of documents identifiers

The identification of documents is described below:
XXX_<document type>_<document number>_<revision index>
where:
" document type " is:

- Foo for FOO documents,
- BAR for bar documents

" document number " is a incremental number, with a separate list for each document type,
" revision index " designates the approved iteration of the document. The revision index is V1 for the first iteration, V2 for the second and so on.

3.1.3.2 Definition and evolution of the revision index

The attribution of a revision index is a prerequisite to any delivery of a document or file. This index shall be incremented before the diffusion of a modified document. The guidelines for defining a revision index are the same as those outlined in "Version number of a configuration item" (3.1.1.2).

3.1.4 Identification rules of a media

3.1.4.1 Internal identification

The identification of a media is described below:
<configuration item identification >/<media>/<volume>
where:

- "media" is the media number,
- "volume" is an incremental number to distinguish the media if the delivery contains more than one media.

3.2 Reference configuration identification

Each reference configuration is defined by:

- An identifier,
- Its content listed in the corresponding Version Delivery Description document,
- 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

Configuration Baseline Management describes how baselines are established, defined, and controlled over the course of a project.

- Functional Baseline (FBL): High-level description of the system's fundamental needs, including functionality and performance.
- Allocated Baseline (ABL): Describes the distribution and composition of the system, defining the performance of each component at the initial design phase.
- Product Baseline (PBL): Consists of all necessary testing documentation for both functional and physical aspects.
- Developmental Baseline: Indicates how development artifacts are changing over time.

4 Configuration control

The majority of features will not need to be altered uness a bug is discovered or performance is subpar.

4.1 Change Management

Problem Resolution:

- Changes requests are emitted by the project manager according to the problem resolution process,
- When a change request is accepted by the project manager/product manager, a branch is created in the SCM
- The branch identification is …
- Branch content contains the changes applied

Multiple Configuration:

- Changes requests of configuration files are emitted by the product manager according to the production procedure
- When a change request is accepted by the project manager/product manager, a branch is created in the SCM
- The branch identification is …
- Branch content contains the changes applied

4.2 Evolutions control of SOUP items

1. Initial Assessment and Baseline Establishment
    a. Perform a thorough assessment of each SOUP item at the beginning of the project to ascertain its necessity, dependability, and support status.

   b. For every SOUP item, establish a baseline version. Stable, popular, and well-supported versions are the best choices.
2. Change Control Process
   a. Establish a change control procedure for SOUP item updates. This entails evaluating the update's effects, putting it through a controlled environment test, and recording the modification.
   b. The project's configuration management team must approve changes to SOUP items to make sure that updates don't jeopardize the integrity of the software or add needless risks.
3. Constant Observation
   a. During the course of the project, keep a close eye on the progress of SOUP items' development and support. This involves keeping tabs on security flaws, known problems, and new releases.
   b. Maintain contact with open-source communities or SOUP vendors to learn about important updates or changes that could impact the project.
4. Flexible Freezing Policy
   a. Although halting SOUP items at the start of a project can improve stability, it is acknowledged that this strategy might not always be possible because of changing project requirements or new security flaws.
   b. Adopt a flexible freezing policy that, depending on the update's possible benefits and risk assessment, permits the controlled updating of SOUP items when needed.
5. Documentation and Auditing
   a. Keep thorough records of every SOUP item used in the project, including baseline versions, change histories, and justifications for updates.
   b. To guarantee adherence to the project's configuration management guidelines and to spot chances for consolidation or optimization, regular audits of SOUP items should be carried out.

5 Configuration support activities

5.1 Configuration Status Accounting

Configuration Status Accounting (CSA) is the process of recording, storing, maintaining and reporting the status of configuration items during the software lifecycle. All software and related documentation should be tracked throughout the software life.

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.
- Source file: The software configuration management tool records, for each source file or group of source files, a comment that describes the modification.
- Configuration item: The Version Delivery Description of the article 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 SCM sets up the state of all versions and of each configuration article with:

- The label,
- The version number,
- The creation date of the VDD

5.1.3 Configuration status diffusion

The SCM and the QA manager write the VDD.

5.1.4 Configuration status records storage

The 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.

The VDD is written by the SCM, as follows:

- It serves as the main configuration control document that is used to monitor and manage software versions that are released into the field.
- The VDD identifies and describes the software version being delivered to the State, taking into account all modifications made to the program since the last VDD was released. It also provides an overview of the features and contents of the software build.
- A VDD must be used to describe each unique software release, even the first one. Every version of the software that is released at roughly the same time (to separate locations, for example) needs to have its own VDD and version number.
- The software CI product baseline includes the VDD. When the version description document is distributed, it should be sent with a cover memo that lists the major changes included in the release on one page.
- Each VDD can be uniquely identified and tracked down with the aid of the label and version.
- An executive summary of the information contained in the attached VDD can be found in the memo that is attached to it. The cover memo should refer to the treatment as a Summary of Changes.

5.2 Configuration audits

Examining and validating the CIs within shelvd is a crucial step in the configuration audit process. One of the management tools for creating a baseline is configuration reviews.

The configuration audits and reviews scheduled for shelvd are specified in the audit plan.

Similar to other configuration audits, configuration audits will be baseline audits that include functional assessments (FCA) and physical assessments (PCA). However, they will have a specific focus on the CIs and the documentation within the baseline that has been established.

For shelvd , we will conduct three primary types of configuration audits:

1. Functional Configuration Audit (FCA)

   a. The purpose of the FCA is to confirm that shelvd satisfies its functional and performance requirements through a critical evaluation. This audit evaluates whether the finished software product satisfies the predetermined requirements, paying particular attention to features associated with user interface elements, user account management, search functionality, and book cataloguing. The FCA, is planned for the conclusion of the development phase but before beta testing, entails a comprehensive review carried out in a controlled setting by the project manager, lead developer, and QA manager. They make use of the test reports and requirements specification document to find any deviations from the established standards. The main objective is to guarantee that the application satisfies all functional requirements and is free of critical bugs. Any problems found will be recorded and will be given attention for correction.

2. Physical Configuration Audit (PCA)
   a. The PCA verifies that every CI in the shelvd —including the codebase, databases, and documentation—conforms to technical descriptions by looking over each one. This audit confirms that the app's physical parts match the technical requirements and design documentation for the project. To guarantee the consistency and integrity of the physical structure of the app, the PCA is a crucial step that is carried out prior to release, following significant development milestones. In this process, the database administrator, lead developer, and configuration manager all have important responsibilities. They carefully check the CIs against the technical documentation. Before approval, any inconsistencies are recorded, and corrective actions are assigned to guarantee total alignment with the technical documentation.
3. In-Process SCM Audits
   a. Throughout the course of a project, in-process SCM audits provide ongoing assessment, with a particular emphasis on ensuring adherence to SCM plans, policies, and procedures as well as software product requirements and quality standards. These audits evaluate the efficacy of SCM procedures and instruments, such as CI/CD pipelines and version control systems, in managing the project's CIs and guaranteeing quality control. These audits, which are regularly carried out at prearranged intervals, include the project manager, DevOps engineer, and SCM specialist. The team finds gaps in SCM practices and suggests changes by looking over build and deployment records, version control logs, and SCM plan adherence. Strict adherence to SCM policies and effective software configuration management are the goals of the recommended changes.

A specialised Configuration Audit Team made up of representatives from the development, SCM, and QA departments oversees the implementation of this configuration audit plan. Planning, organizing, and directing audits, making sure that findings are taken seriously, and keeping thorough audit records are the responsibilities assigned to this team. Additionally, they report results to senior management, providing information on how well the project adheres to its CM plan and pointing out areas where improvements can be made. The shelvd app project is well-positioned to meet and surpass stakeholder expectations by adhering strictly to this plan, which will guarantee the highest quality standards are met.

5.3 Reviews

The creation and implementation of a Configuration Management Plan Review (CMPR) is a critical step in ensuring the effectiveness and comprehensiveness of the configuration management strategy. The review serves as an independent evaluation of the existing configuration management plan, focusing on its capacity to adequately cover the five foundational elements of CM:

- Configuration Planning
- Identification
- Status Accounting
- Change Control and Verification
- Audits

Throughout the Software Development Life Cycle (SDLC), technical reviews are essential to ensure that a high-quality product that satisfies all predetermined baselines is delivered by the project's end. The Configuration Manager is in charge of making sure that each project component is precisely identified and that any changes to the documentation are carefully noted and updated during meetings and review sessions.

During the review process, the CM manager and the assigned team members—as previously specified in the project's organizational framework—will assess the CM Plan's viability and usefulness in light of the project's particular needs and specifics. This assessment verifies that the plan is practically applicable and evaluates its adherence to regulatory requirements and specified quality management standards (referred to as STD1 and STD2, respectively).

In addition, the review team will assess if the CM Plan satisfies contractual requirements for quality management and conforms to all applicable regulatory compliance standards. The team's goal in going through this rigorous review process is to confirm that the CM Plan is both theoretically and practically sound, meaning that it can be successfully implemented to effectively manage the configuration requirements of the project. This all-encompassing strategy makes sure that the CM Plan is strong, flexible, and in line with the project's objectives, which makes it easier to deliver a product that satisfies stakeholders' expectations as well as the project's standards.

5.4 Configuration management plan maintenance

Throughout the project's life cycle, the CM plan outlined the tasks and accountability for ongoing configuration management planning. Throughout the project's life cycle, the SCM plan maintenance information will specify the tasks and roles required to guarantee ongoing configuration management planning.

The CM maintenance schedule would look something like this:

- CM plan monitoring falls under the purview of SCM.
- The CM plan will be examined every week.
- A weekly review will be held to update the CM plan.
- The project manager and SCM must approve changes.
- The project team will be notified via email of any approved changes.

At the beginning of each software project phase, the CM plan will be examined, adjusted as necessary, approved, and given to the project team.
Several maintenance mechanisms for the detailed procedures that have been documented elsewhere in appendices or references may be appropriate if those procedures were used in the construction of the CM plan.

All of the CM maintenance tasks will adhere to the Software Configuration Management Plans (SSCM1) IEEE 828 Standards.