Software Configuration Management Plan

1. Introduction

Software Configuration Management (SCM) is a systematic process used to identify, organize, control, and track changes to software artifacts throughout the software development life cycle. For this project, SCM plays a crucial role in ensuring that all project components—such as source code, documentation, design models, configuration files, and test artifacts—are consistently managed, versioned, and maintained.

The primary objective of configuration management in this project is to maintain the integrity, traceability, and reproducibility of all software deliverables while supporting controlled evolution of the system. As the project progresses through various phases including requirements analysis, design, development, testing, and deployment, multiple versions of artifacts are created. SCM ensures that the correct versions are available to the right stakeholders at the right time, thereby reducing errors, rework, and inconsistencies.

This Configuration Management Plan (CMP) defines the policies, procedures, roles, and tools used to manage project configurations effectively. It establishes a structured approach for handling changes, tracking revisions, and auditing configuration items. The plan also ensures compliance with recognized software engineering practices and standards.

Key SCM activities covered under this plan include:

- **Configuration Identification** – identifying and labeling all configuration items.
- **Configuration Control** – managing and approving changes to configuration items.
- **Configuration Status Accounting** – recording and reporting the status of configuration items and changes.
- **Configuration Auditing** – verifying that configuration items conform to requirements and standards.

By implementing an effective SCM process, the project aims to:

- Improve collaboration among team members
- Minimize risks caused by uncontrolled changes
- Ensure product quality and consistency
- Support maintainability and future enhancements

2. Configuration Identification

Versioning of Software and Documents

Configuration Identification involves selecting and uniquely labeling all project artifacts that require configuration management. In this project, configuration identification ensures that all

software components and documents are clearly defined, traceable, and controlled throughout the development life cycle.

Configuration Items

The following are treated as configuration items (CIs):

- Source code files and scripts
- Requirements and design documents (SRS, diagrams, flowcharts)
- Test artifacts (test plans, cases, and reports)
- Configuration files and user documentation
- Project management documents

Each configuration item is assigned a unique name and version number to avoid ambiguity and maintain consistency.

Versioning Strategy

A standard versioning format is followed:

Major.Minor.version

- **Major** version indicates significant functional or architectural changes
- **Minor** version represents feature enhancements
- **Revision** version denotes bug fixes or minor updates

Version Control

All software files are maintained in a version control system, where changes are tracked using commit history and descriptive messages. Stable versions are tagged after testing and approval.

Importance

- Maintain consistency across project artifact.
- Track changes effectively
- Reduce errors caused by incorrect versions
- Support audits and future maintenance

3. Configuration Control

Change Control Process and Approval Authority

Configuration Control is the process of **systematically managing and approving changes** to software configuration items. Its purpose is to ensure that **only authorized and necessary changes** are implemented while preserving the system's integrity and stability.

3.1 Change Control Process

All changes to configuration items such as source code, documents, or design artifacts must follow a **formal change control process**. The process begins when a change is identified due to bug fixes, requirement updates, enhancements, or improvements.

The change control process includes the following steps:

1. **Change Request Submission** – A change request (CR) is documented with details such as the reason for change, affected components, and expected impact.
2. **Impact Analysis** – The proposed change is analyzed to assess its impact on cost, schedule, functionality, performance, and quality.
3. **Review and Evaluation** – The change request is reviewed by designated authorities to determine feasibility and necessity.
4. **Approval or Rejection** – Based on the evaluation, the change is either approved, deferred, or rejected.
5. **Implementation** – Approved changes are implemented, tested, and versioned according to configuration management procedures.
6. **Update and Documentation** – Configuration records and documents are updated to reflect the approved change.

3.2 Approval Authority

Change approval authority is clearly defined to prevent unauthorized modifications. A Configuration Control Board (CCB) or designated project authority is responsible for reviewing and approving all change requests.

- Minor changes (such as bug fixes or documentation updates) may be approved by the Project Manager or Technical Lead.
- Major changes (including requirement or architectural changes) require approval from the CCB. Emergency changes are implemented on priority and documented for retrospective approval.

4. Configuration Status Accounting

Tracking and Reporting Changes

Configuration Status Accounting is the process of systematically collecting, maintaining, and communicating information about the status of configuration items and change requests throughout the software development life cycle. It plays a vital role in ensuring visibility, traceability, and control over all configuration management activities.

In this project, configuration status accounting ensures that every configuration item and change request is properly documented and traceable from initiation to closure. Each change is assigned a unique identifier and tracked along with details such as the description of change,

affected configuration items, version number, approval status, implementation date, and responsible personnel.

Status accounting provides real-time insight into the current baseline of the system by maintaining records of:

- Approved, pending, rejected, and implemented change requests
- Current and previous versions of software and documentation
- Relationships between configuration items and change requests

Regular status reports are generated and shared with project stakeholders to support project monitoring and decision-making. These reports help identify trends such as frequent changes, delayed approvals, or high-impact modifications, enabling timely corrective actions.

In this project, every change request is documented and tracked from initiation through approval, implementation, and closure. Information such as change request identification, description of change, affected configuration items, version history, approval authority, and implementation status is maintained in a centralized repository.

Status reports are periodically generated and shared with project stakeholders to provide visibility into:

- The current version and baseline of each configuration item
- Approved, pending, and rejected change requests
- Progress and impact of implemented changes

By maintaining accurate and up-to-date configuration status records, the project team can effectively monitor change activities, support audits and reviews, and ensure consistency across all software and documentation artifacts.

5. Configuration Audits

Verification of Configuration Items

Configuration Audits are systematic and formal evaluations conducted to verify that configuration items are complete, consistent, and compliant with approved requirements, standards, and project baselines. The primary purpose of configuration auditing in this project is to ensure that all configuration items have been correctly developed, documented, reviewed, and approved before being released or baselined.

Configuration audits confirm that changes to configuration items have been properly authorized, implemented, and recorded through the configuration management process. They also help identify discrepancies such as missing documents, incorrect versions, unauthorized modifications, or deviations from approved requirements.

Two major types of configuration audits are performed:

- **Functional Configuration Audit (FCA):**

  This audit verifies that the functional behavior of the system satisfies all approved functional and non-functional requirements. It ensures that the implemented features reflect the intended design and that all approved change requests have been correctly incorporated.

- **Physical Configuration Audit (PCA):**

  This audit verifies the physical completeness of configuration items, including source code, documentation, test artifacts, and configuration files. It ensures that all items are properly versioned, labeled, and stored in controlled repositories, and that documentation accurately reflects the current system state.

Configuration audits are conducted at **key milestones**, such as the completion of major development phases or prior to system release. Audit findings are documented and reviewed, and any non-conformances are recorded as corrective actions or change requests. These issues must be resolved before the configuration baseline is finalized.

Through regular and thorough configuration audits, the project ensures high quality, traceability, compliance with standards, and reliable configuration baselines, thereby supporting effective maintenance, future enhancements, and long-term system stability.