Software Configuration Management Plan

1. Introduction

Software Configuration Management (SCM) is a disciplined process for identifying, organizing, and controlling changes to software artifacts throughout the project lifecycle. This Software Configuration Management Plan defines the SCM activities for the development of the Real-Time Collaborative Online Integrated Development Environment (RCO-IDE). The objective of this document is to ensure integrity, traceability, and controlled evolution of all software and documentation components developed as part of the academic project.

2. Purpose of Configuration Management

The purpose of configuration management is to establish a systematic approach for managing changes, maintaining consistency, and ensuring that all stakeholders work with the correct and approved versions of project artifacts. SCM helps prevent unauthorized changes, minimizes errors caused by version conflicts, and supports effective collaboration among team members.

3. Configuration Identification

Configuration identification involves selecting and uniquely identifying configuration items (CIs) that need to be controlled. The following configuration items are identified for this project:

• Source Code: Frontend (React), Backend (Node.js/Express), and configuration files.

• Documentation: SRS, Problem Statement, Risk Management Plan, SCM Plan, and other academic documents.

• Database Schemas: MongoDB schema definitions and migration files.

• Build and Deployment Files: Dockerfiles, environment configuration files, and scripts.

• Test Artifacts: Test cases, test data, and test reports.

Each configuration item will follow a versioning scheme using semantic versioning (e.g., v1.0, v1.1, v2.0) and will be stored in a centralized version control repository.

4. Configuration Control

Configuration control ensures that all changes to configuration items are formally proposed, reviewed, approved, and implemented. The change control process includes the following steps:

• Change Request Submission: Any modification request must be documented with a description, reason, and impact analysis.

• Change Evaluation: The project team reviews the request for technical feasibility and academic relevance.

• Approval Authority: The project guide or team lead approves or rejects change requests.

• Implementation: Approved changes are implemented in a controlled manner.

• Verification: Changes are tested and verified before final acceptance.

This process ensures accountability and maintains project stability.

5. Configuration Status Accounting

Configuration status accounting involves recording and reporting the status of configuration items throughout the project lifecycle. This includes:

• Tracking current versions of all configuration items.

• Maintaining a history of changes, including author, date, and description.

• Generating periodic reports on configuration status for review.

Version control tools are used to automatically log changes and provide traceability between versions.

6. Configuration Audits

Configuration audits are conducted to verify that configuration items conform to their documented requirements and approved baselines. The two main types of audits performed are:

• Functional Configuration Audit (FCA): Ensures that the software meets specified functional requirements.

• Physical Configuration Audit (PCA): Verifies that all configuration items are complete, correctly versioned, and properly documented.

Audits are conducted at key milestones to ensure compliance with project standards.

7. Tools and Responsibilities

The project uses Git-based version control systems for managing source code and documentation. The project manager or team lead is responsible for SCM enforcement, while all team members are responsible for following SCM procedures, committing changes properly, and maintaining documentation consistency.

8. Conclusion

An effective Software Configuration Management Plan is essential for maintaining control over project artifacts and ensuring successful project execution. By implementing structured SCM practices, the RCO-IDE project achieves better organization, improved collaboration, and reduced risk of errors, making it suitable for academic evaluation and future enhancement.