Software Project Management (SPM)

Course Code: CACS407
Year/ Semester: IV/VII

Compiled by Shishir Ghimire

Credit Hours: 3hrs

Unit - 07: Software Configuration Management

Class Load : 7 Hrs

TABLE OF CONTENTS

Software Configuration Management **7 Hrs**

Concept, Requirement and Elements of SCM, Baseline, SCM Repository, Versioning and version control, SCM Process, Change Control Process. Configuration Audit and Status Reporting. Case Study: Version Control Software Tools (Git, CVS, SVN)

Concept

7.1

Software Configuration Management (SCM):

- The art of **identifying, organizing, and controlling** modifications to the software being built.
- The main goal is to **minimize mistakes**.
- It is an **umbrella activity**, applied throughout the software engineering process.
  - **Auditing** of the software configuration.
  - **Reporting** of all the changes applied to configuration.
  - Control of change.

Requirement & Elements of SQM

7.2

Requirement and Elements of SCM

1. Identification
2. Control
3. Status Accounting
4. Audit

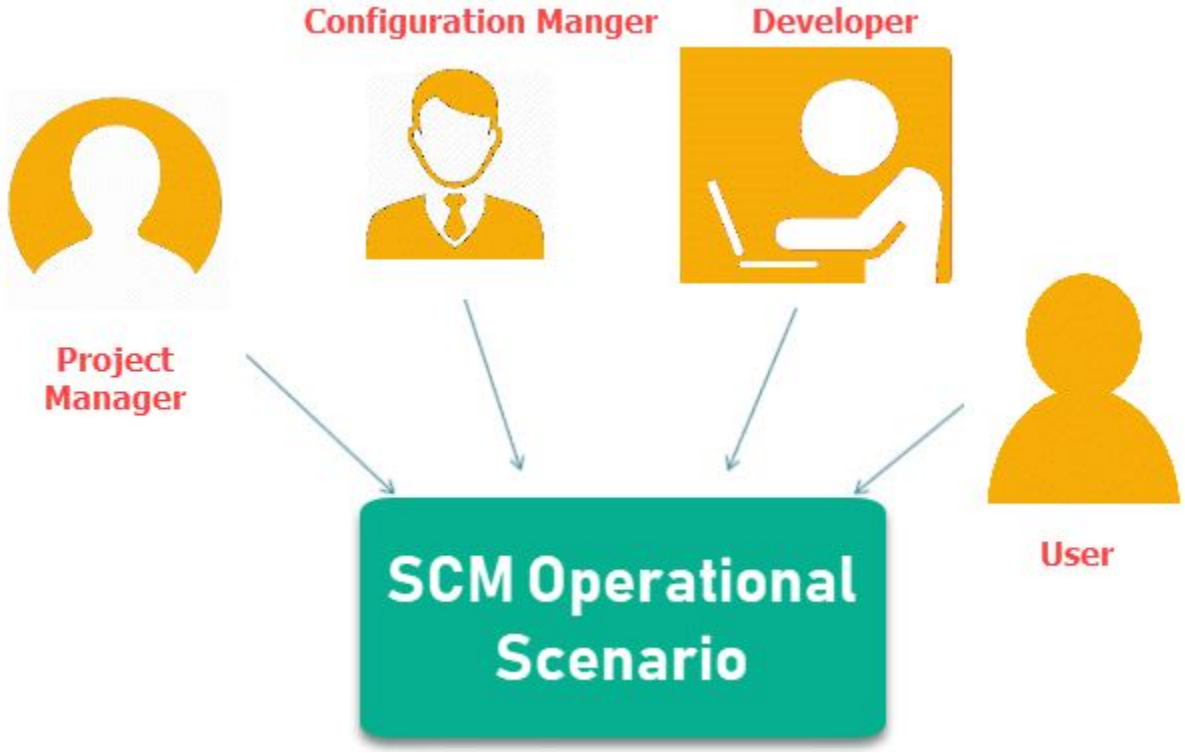
Requirement and Elements of SCM

- **Identification:**
- Each software **part is labeled** so that it can be identified.
- Furthermore, there will be different versions of the software parts as they **evolve over time**, so a version or revision number will be associated with the part.
- The key is to be able to **identify any and all artifacts** that compose a released configuration item.
- It helps to answer the following **questions**:
    - What is the component of the product?
    - What is the version of the configured item?
- Think of this as a bill of materials for all the components in your automobile. When the manufacturer realizes that there has been a problem with parking brakes purchased from a subcontractor, it needs to know all the automobile models using that version of the parking brake.

Requirement and Elements of SCM

- **Control:**
  - Control is the **process of managing changes** to the configuration items. It includes the activities required to implement and enforce the process for controlling changes.
- **Status Accounting:**
  - Status accounting is the **recording and reporting** of the status of configuration items.
- **Audit:**
  - Audit is an **independent review** of configuration items to ensure that they conform to their specifications and that the specified process has been followed.

Participants/Roles of SCM Process:



Participants of SCM:

- **1. Configuration Manager**
  - Configuration Manager is the head who is responsible for identifying **configuration items.**
  - CM ensures team follows the SCM process.
  - He/She needs to **approve or reject** change requests
- **2. Developer**
  - The developer needs to change the code as per **standard development activities** or change requests. He is responsible for maintaining configuration of code.
  - The developer should check the changes and resolves conflicts.
- **3. Auditor**
  - The auditor is responsible for **SCM audits and reviews**.
  - Need to ensure the consistency and completeness of release.

Participants of SCM:

- **4. Project Manager:**
  - Ensure that the product is developed **within a certain time frame**.
  - **Monitors the progress** of development and recognizes issues in the SCM process.
  - **Generate reports** about the status of the software system
  - Make sure that **processes and policies are followed** for creating, changing, and testing
- **5. User**
  - The end user should understand the key SCM terms to ensure he has the **latest version** of the software

Baseline

7.3

Baseline:

- In software configuration management (SCM), a baseline is a **formally approved and documented version of a software product at a specific point in time.** Think of it as a snapshot that serves as a **reference point** for tracking changes and ensuring project stability.
- A baseline is a **formally accepted version** of a software configuration item.
- It is **designated and fixed at a specific time** while conducting the SCM process. It can only be changed through formal change control procedures.

Baseline:

- **Activities during this process:**
  - **Facilitate** construction of **various versions** of an application
  - **Defining** and determining **mechanisms** for managing various versions of these work products
  - The functional baseline **corresponds** to the reviewed system requirements
  - Widely used baselines include functional, developmental, and product baselines
- In simple words, baseline means **ready for release**.
- A baseline is a **milestone** in the development of software that marked the delivery of one or more software configuration items.

Baseline:

- **IEEE** define baseline as: "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 procedure"
- Common Baselines:
  - Software Engineering /Specification
  - Requirement Analysis
  - Software design
  - Coding
  - Testing
  - Release

Baseline > Benefits :

- **Reduced errors:** Tracking changes against a baseline helps identify and prevent regressions or unintended modifications.
- **Improved visibility:** Everyone involved in the project can clearly understand the current state of the software and any planned changes.
- **Streamlined change management:** Formalized change control processes based on baselines help ensure controlled and documented modifications.
- **Simplified quality assurance:** Testing efforts can focus on changes made since the last baseline to ensure quality is maintained.

SCM Repository

7.4

SCM Repository:

- An SCM (Software Configuration Management) repository is a **central location where developers store and manage all files and data** related to a software project, including source code, documentation, and configuration files. The repository **tracks different versions** of files, provides version control features, and ensures consistency and integrity throughout the development process. It serves as a **secure and organized hub** for managing and controlling the evolution of software projects.
- The repository performs or precipitates the following **functions**:
  - Data integrity
  - Information sharing
  - Tool integration
  - Data integration
  - Methodology enforcement
  - Document standardization

Versioning and Version Control

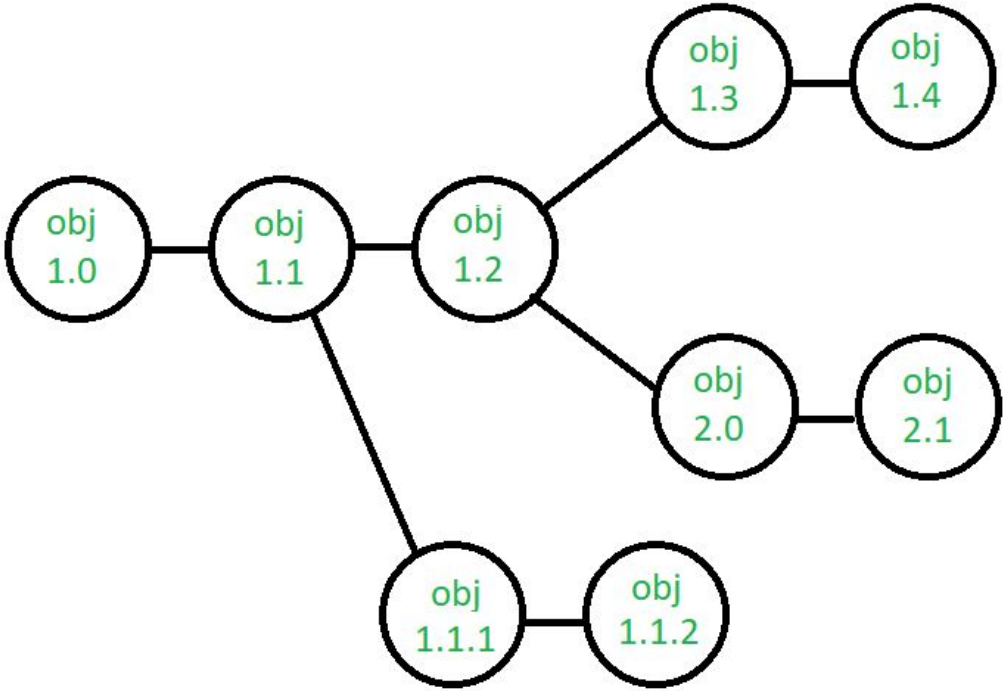
7.5

Versioning & Version Control:

❖ Versioning:

- ➢ **Definition:** The act of **assigning unique identifiers** to different versions of a software product or artifact. These **identifiers can be** simple numbers (v1.0, v2.1), letters (A, B, C), or more complex schemes.
- ➢ **Purpose:** Allows for **clear distinction** between different versions, enabling users to identify the specific version they are using and track changes over time.
- ➢ **Examples:** Software releases, documentation revisions, design prototypes.

Versioning & Version Control:



Versioning & Version Control:

❖ Version Control:

- ➢ **Definition:** A **system that tracks and manages** changes made to software files and documents. It stores different versions of files, allows users to revert to previous versions, and helps collaborate on projects by preventing conflicts.
- ➢ **Purpose:** Enables **developers to work on the same codebase** without overwriting each other's work, facilitates collaboration and code reviews, and provides a **historical record** of changes.
- ➢ **Examples:** Git, SVN, Mercurial, CVS.

Versioning & Version Control Benefits:

- **Improved Traceability:** Track changes to the codebase and understand who made what changes when.
- **Easier Collaboration:** Multiple team members can work on the same project without conflicts.
- **Rollback Capability:** Revert to previous versions if issues arise.
- **Enhanced Quality Assurance:** Test specific versions to ensure stability.
- **Improved Documentation:** Keep a detailed history of changes made to documents and code.

SCM Process

7.6

SCM Process:

- The Software Configuration Management (SCM) process involves a **set of activities and tools to control and track changes made to software throughout its lifecycle.** It ensures stability, consistency, and fulfillment of the intended purpose.

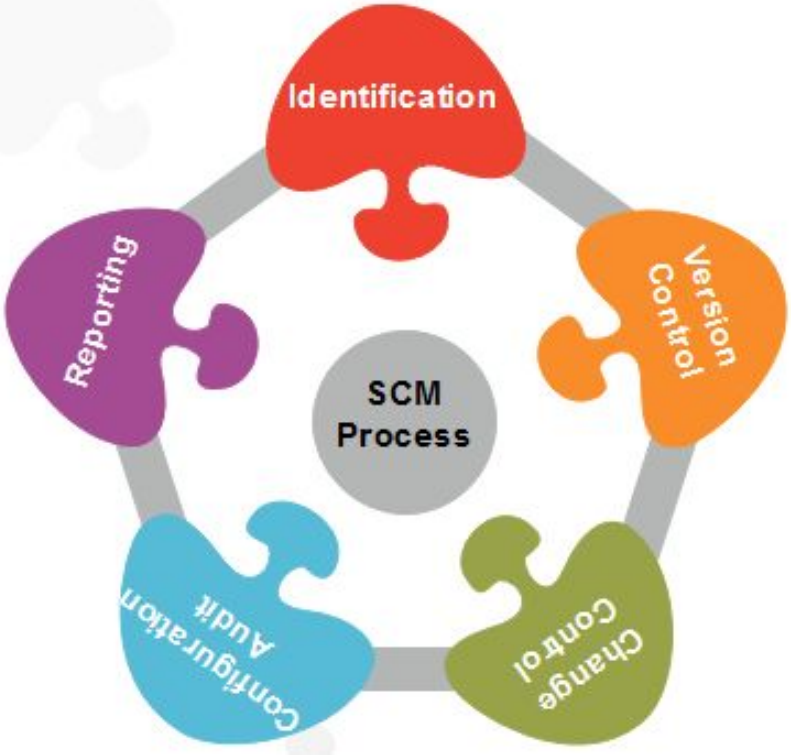
The **stages involved in SCM**:

- **Identification:**
  - **Identify** all elements that need to be controlled, including source code, documentation, test cases, configurations, etc.
  - **Define** clear naming conventions and versioning schemes for each element.
- **Version Control:**
  - **Implement** a version control system like Git or SVN to store different versions of each element and track changes made.
  - **Enforce** controlled access and change management procedures to prevent conflicts and maintain traceability.

SCM Process:

- **Change Management:**
  - Establish a **formal process** for proposing, reviewing, and approving changes to controlled elements.
  - This process ensures changes are necessary, well-defined, and don't negatively impact other parts of the software.
- **Configuration Management:**
  - **Define and manage** different configurations of the software for different environments (development, testing, production, etc.).
  - **Ensure** consistency and compatibility between different configurations.
- **Status Reporting and Auditing:**
  - Regularly **track the status** of changes and configurations to maintain visibility and control.
  - **Conduct audits** to verify compliance with established procedures and identify potential issues.

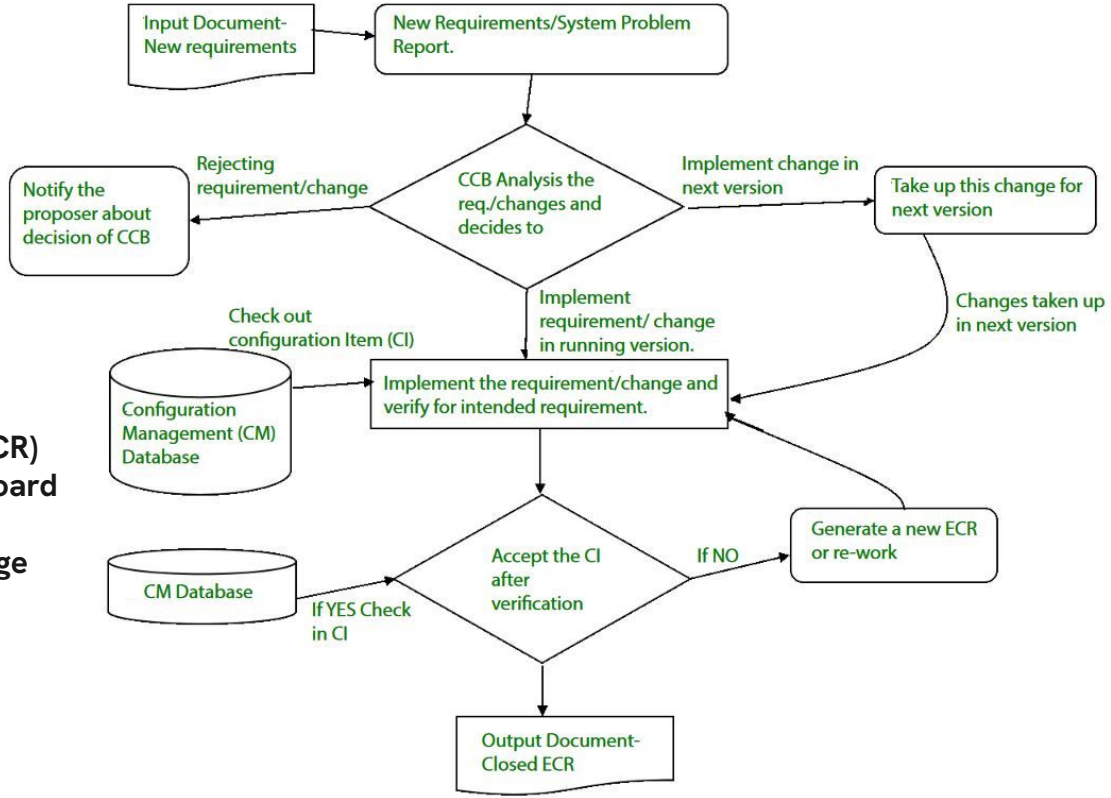
SCM Process:


Identification
Version Control
Change Control
Configuration Audit
Reporting
SCM Process

Change Control Process

7.7

Change Control Process:



- change request (CR)
- change control board (CCB)
- engineering change Request (ECR)

Change Control Process (CCP):

- Change Control Process (CCP) is a **formalized, structured approach** used to manage and approve changes made to any product, project, or service, including software. The **document ensures that** proposed changes are **evaluated** for impact, risks, and **potential benefits** before being implemented.
- **Types of Change Requests:**
  - **Changes beyond your control:**
    - Weather events
    - Team attrition
    - Employee sickness
  - **Changes that are specifically requested:**
    - Adding features to software applications
    - Updating network equipment
    - Installing system patches

Change Control Process > Key Steps :

- **Initiation:**
  - A change request is proposed, outlining the desired change, rationale, and potential impact.
- **Evaluation:**
  - The request is reviewed by a designated team to assess its feasibility, risks, and benefits. This might involve technical analysis, cost estimation, and impact on other parts of the system.
- **Approval:**
  - Based on the evaluation, the request is approved, rejected, or modified. Decisions are documented and communicated to all stakeholders.
- **Implementation:**
  - If approved, the change is implemented according to a defined plan, which may involve testing, documentation updates, and training.
- **Review and Closure:**
  - The effectiveness of the implemented change is reviewed, lessons learned are documented, and the change is formally closed.

Change Control Process > Benefits :

- **Reduced risks:** CCP helps identify and mitigate potential negative consequences of changes.
- **Improved quality:** Ensures changes are well-defined, tested, and meet quality standards.
- **Increased transparency:** Keeps stakeholders informed of proposed and implemented changes.
- **Better cost control:** Allows for careful evaluation of cost implications before approval.
- **Reduced rework:** Minimizes the need to undo poorly planned or implemented changes.

Change Control Procedures in Software Project Management:

- **Step-1:Document the Change Request:**
  - When a change request occurs, the first step is to categorize and record it. This is required even if the change is ultimately not pursued.
- **Step-2: Conduct a Formal Change Evaluation:**
  - The project team will meet and formally evaluate the change. Key questions to ask include:
    - Is the change justified?
    - What are the risks and benefits of making the change?
    - What are the risks of not making the change?
    - Do the risks outweigh the benefits?
    - If the change is accepted, the next steps are assigned to a dedicated development team.

Change Control Procedures in Software Project Management:

- **Step-3: Plan the Change:**
  - The team tasked with developing the change will create a detailed plan outlining how it will be designed and implemented.
  - Methods should be in place to define and verify success.
- **Step-4: Design Software Changes:**
  - If the requested change affects software, the team will design the new program and test it for accuracy. If it passes successfully, they will request approval and set a date for implementation.
- **Step-5: Conduct an Internal Software Review:**
  - Before the final approval and full implementation of any changes, an internal review will be conducted to ensure all aspects of the change meet the necessary standards and requirements.

Configuration Audit and Status Reporting

7.8

Configuration Audit & Status Reporting:

- **Configuration Audit:**
  - **Focus:** Examining and verifying the **current state** of a software product **against** its **documented configuration**.
  - **Activities:**
    - Identifying discrepancies or unauthorized changes.
    - Assessing the impact of any deviations.
    - Verifying consistency across environments.
    - Employing methods like physical/functional/documentation audits.
  - **Outcome:** Raw data and findings about the configuration adherence to documentation.

Configuration Audit & Status Reporting:

- **Configuration Status Reporting:**
  - **Focus:** Presenting the **findings of the audit** in a clear and concise format.
  - **Content:**
    - Summary of audit findings with impact analysis.
    - Recommendations for corrective actions.
    - Status of change requests and their potential impact.
    - Metrics on configuration stability and change control effectiveness.
  - **Outcome:** Distilled information and recommendations for stakeholders to make informed decisions.

Configuration Audit Vs Status Reporting:

Factors	Audit	Reporting
Purpose	Finds Issues	Communicate Issues.
Audience	For Internal Team	For Broader Stakeholders
Action	Gathers Data	Recommends Solution
Output	Findings as Raw Data	Processed Information

Case Study: Version Control Software Tools (Git, CVS, SVN)

7.9

Case Study: Version Control Software Tools (Git, CVS, SVN)

- This case study **compares** three commonly used version control software tools: Git, CVS, and SVN. Each tool offers unique features and benefits, suitable for different project needs and workflows.
- **Git:**
  - **Type:** Distributed version control system
  - **Key Features:**
    - Each developer has a complete copy of the repository, enabling offline work.
    - Strong focus on branching and merging, facilitating parallel development.
    - Powerful command-line interface, though with a steep learning curve for beginners.
  - **Ideal For:** Large, complex projects with active collaboration.

Case Study: Version Control Software Tools (Git, CVS, SVN)

- **CVS (Concurrent Versions System):**
  - **Type:** Centralized version control system
  - **Key Features:**
    - Single server repository, requiring constant network access.
    - Simple branching and merging capabilities.
    - User-friendly interface, suitable for beginners.
  - **Ideal For:** Smaller projects or teams with less complex branching needs.

Case Study: Version Control Software Tools (Git, CVS, SVN)

- **SVN (Apache Subversion):**
  - **Type:** Centralized version control system
  - **Key Features:**
    - More robust and scalable than CVS.
    - Improved branching and merging capabilities.
    - User-friendly interface with both command-line and GUI options.
  - **Ideal For:** Mid-sized projects needing a balance between features and ease of use.

Case Study: Version Control Software Tools (Git, CVS, SVN)

Feature	Git	CVS	SVN
System Type	Distributed	Centralized	Centralized
Repository	Complete copy on each developer's machine	Single server	Single server
Branching & Merging	Strong focus, enables parallel development	Simple branching and merging	Improved capabilities over CVS
Interface	Powerful command-line (steep learning curve)	User-friendly, easy to learn	User-friendly (command-line and GUI options)
Ideal For	Large, complex projects with active collaboration	Smaller projects or teams with less complex needs	Mid-sized projects needing balanced features
Network Access	Offline work enabled	Constant network access required	Constant network access required

ALL THE BEST!

Do you have any questions?

theciceerguy@p_m.me

9841893035

ghimires.com.np