Course: Software Configuration Management

Week 6: Software Configuration Identification

Lecturer: Yimer Amedie (MSc.)

Addis Ababa Science and Technology University, Ethiopia

October, 2025

Contents

Figure 1: Concepts of SCM
(Source: OpenAI, 2025)

- Introduction
- Configuration Items (CIs) and Baselines
- Naming, Documentation, and Traceability
- Software Libraries and Their Roles
- Benefits, Challenges, and Risks
- Summary

Learning Outcomes

After completing this lesson, you will be able to:

- Define software configuration identification and explain its role in SCM.
- Identify and classify different types of Configuration Items (CIs).
- Explain the purpose of baselines, naming conventions, and software libraries.
- Demonstrate how traceability supports change control.
- Evaluate challenges, risks, and best practices to improve CI processes in real-world projects

Introduction

- Configuration Identification
  - First step in SCM process execution
  - Process of identifying and defining system components to be managed
  - Defines what will be controlled
  - Establishes unique identities for each item under SCM control
  - Enables baselines
  - Foundation of configuration management discipline

Purpose of Identification

- Prevents confusion in large projects
- Supports team collaboration
- Facilitates impact analysis
- Provides accountability
  - Define what constitutes the configuration baseline
  - Facilitate accurate change control and reporting
  - Ensure traceability and reproducibility

Elements of Configuration Identification

- Identification of Configuration Items (CIs)
- Establishment of baselines
- Naming and numbering conventions
- Version and revision control
- Documentation and traceability mechanisms

Configuration Items (CIs)

- The building blocks of configuration identification
  - A uniquely identifiable component of a system managed through SCM
  - Types


Software


Documents


Test Data


Libraries and tools

Configuration Items (CIs)

Computer Software Configuration Item

Top-level software configuration item

Computer Software Components

Grouped units forming larger parts

Computer Software Units

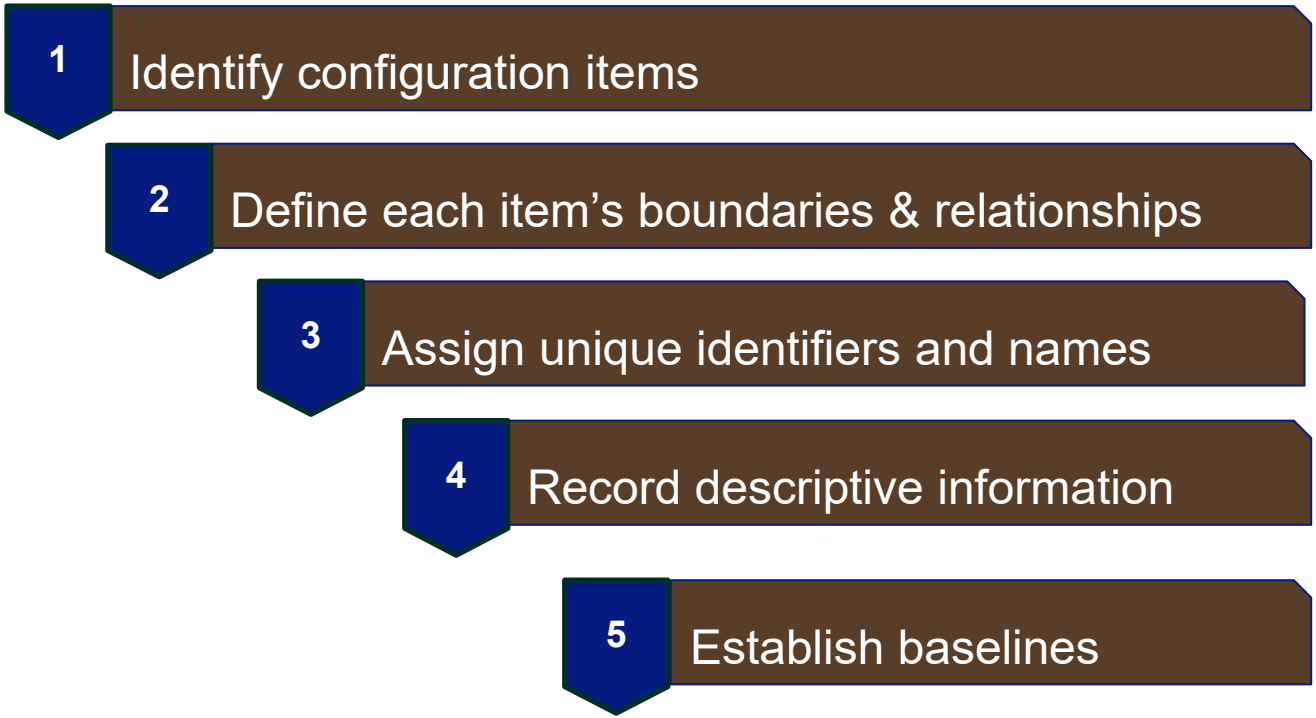
Basic software building blocks

Hierarchical structure:

- CSCI ➔System
- CSC ➔Subsystem
- CSU ➔Unit/Module

Steps in Configuration Identification

The CI process follows a structured sequence (Leon, 2025)



Impact of CI Selection

- Choosing which components to control as configuration items has major implications.
  - CI selection affects control effort, documentation, and traceability
  - Over- or under-selection leads to inefficiency or lack of control
  - Must align with project complexity and risk

Too Many vs. Too Few CIs

- **Too Many:**
  - Excessive documentation
  - Redundant control
  - Wasted resources
    - **Too Few:**
      - Missing traceability
      - Uncontrolled changes
      - Audit difficulty

Baselines and Identification

- A baseline represents an approved snapshot of a configuration item or a set of items at a specific point in time.
  - Identification ensures baseline clarity
  - Baselines used in reviews and audits
  - Foundation for change control

CI Selection Criteria and Checklist

Technical or functional significance

Reusability or integration dependency

Criticality to performance or quality

Frequency of change

Customer or contractual requirement

Configuration Items – Naming, Description and Documentation

- Unique identifiers: names, numbers, or codes
- Should reflect hierarchy and version
- Follow organizational standards

Designation – Naming

Description & Documentation

- Description includes
  - function, relationships, and version history
- Recorded in CI database or repository
- Supports audits and change tracking

Acquisition and Control of CIs

- After identifying the configuration items, the next task is to ensure their proper acquisition and control
  - Define process for acquiring
    - internally and externally developed CIs
  - Validate authenticity and configuration status before inclusion
  - Ensure supplier items conform to control procedures

Principles of Configuration Items

- Effective item identification rests on four principles
  - Unique identifiers
  - Consistent naming conventions
  - Version numbers
  - Relationships between items

Levels of Identification

Identification occurs at multiple levels of granularity.

System level

Overall solution

Subsystem level

major functional areas

Component/module level

Source code files or database schemas

Document level

Design reports, test plans, and manuals

Software Library Concept

- The Software Library is the backbone of configuration identification:
  - Central repository
  - Stores and protects CIs
  - Supports retrieval and versioning
  - Types:
    - Development
    - Controlled
    - Master

Development Library

- The Development Library serves as the working environment for project teams.
  - Workspace for developers
  - Frequent updates
  - Not yet baselined
  - May include prototypes and drafts

Controlled Library

- The Controlled Library
  - Holds baselined items
  - Limited access
  - Strict version control
  - Used for reviews and audits

Master Library

- The Master Library
  - Official project archive
  - Holds final approved versions
  - Releases and deliveries stored here
  - Immutable once published

Configuration Control

- Once items are identified and stored in libraries, the next essential activity is Configuration Control.
  - Governs changes after identification
  - Ensures discipline in modifications
  - Prevents unauthorized changes
  - Managed by Change Control Board (CCB)

The Role of the CCB

- The Change Control Board is central to configuration control.
  - Evaluates change requests
  - Balances risk, cost, and benefit
  - Maintains baseline integrity
  - Cross-functional representation

Change Request (CR)

- Change request should be considered as software projects are flexible and accommodates changes:
  - Formal proposals for modification
  - Submitted by stakeholders or team members
  - Logged in a Change Management System
  - Each CR has unique ID and status

Role of Traceability in Identification

- Traceability is one of the most important outcomes of effective configuration identification.
  - Links requirements to design, code, tests
  - Supports impact analysis
  - Helps during audits
  - Increases confidence in system integrity

Benefits of Software Configuration Identification

- Software configuration identification delivers tangible benefits to projects of all sizes.
  - Reduces confusion
  - Improves collaboration
  - Enables control and auditing
  - Supports compliance and quality

Common Challenges in Identification

- While software configuration identification is essential, it is not without challenges.
  - Too many items tracked
  - Poor naming conventions
  - Inconsistent versioning
  - Lack of ownership

Best Practices for Item Identification

- To overcome challenges, organizations should adopt best practices in configuration identification.
  - Define clear criteria for CIs
  - Establish consistent naming rules
  - Automate version control
  - Assign ownership to items

Tools Supporting Identification

- Modern tools play a critical role in simplifying configuration identification.
  - Version control systems (Git, SVN)
  - Change management tools (Jira, ServiceNow)
  - Automated build and release tools
  - Integrated SCM platforms

Case Study: Identification in a Banking System

- Consider a case study of a large banking system.
  - CIs:
    - Requirements, APIs, UI modules, test scripts
  - Software library use
  - Change request flow
  - Benefits realized

Integration with Other SCM Activities

- Software configuration identification does not stand alone; it integrates closely with other SCM activities.
  - Planning defines identification rules
  - Status reporting tracks CI evolution
  - Audits verify identification accuracy
  - Tools enforce consistency

Risks of Poor Identification

- When configuration identification is poorly executed, the consequences can be severe.
  - Lost or duplicated work
  - Inability to trace changes
  - Compliance failures
  - Increased project costs

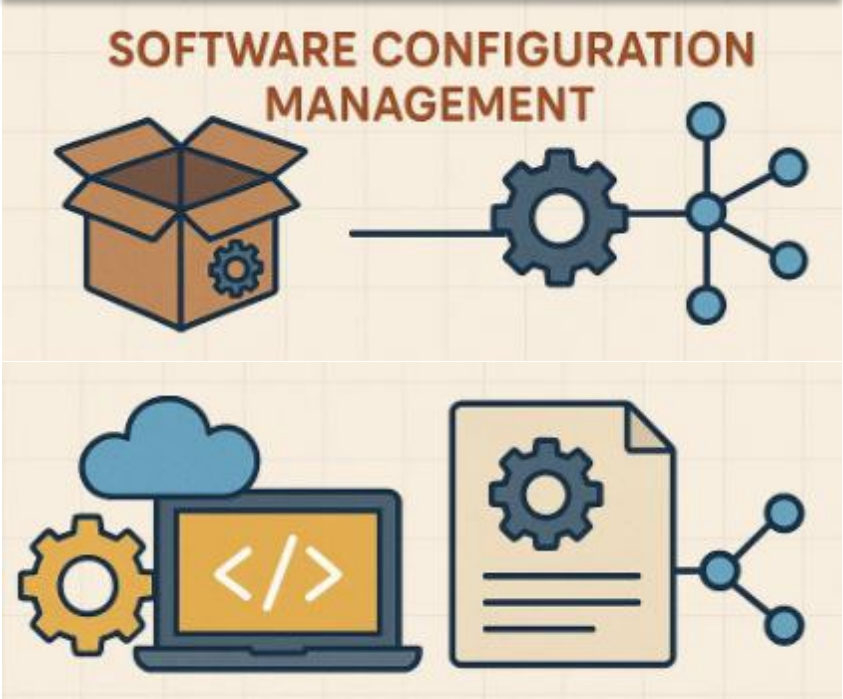
Summary

- Software configuration identification is the foundation of SCM
  - Identification is the foundation of SCM
  - Libraries safeguard artifacts
  - Control and CRs ensure discipline
  - Implementation maintains consistency
  - Supports audits and compliance

References

1. Leon, A. (2015). Software Configuration Management Handbook (3rd ed.). Norwood: Artechhouse. Retrieved September 4, 2025.
2. OpenAI. (2025, September 4). SCM history and concepts [AI-generated image]. ChatGPT (Sora). https://chat.openai.com/

Thank You!



NEXT LECTURE

Software Configuration Identification – Change Management