# Configuration Management

## 🔹 Definition

**# Source**: **Chatgpt (GPT-5-Standard) at [8/18/2025]**

**Configuration Management** is a **process of systematically organizing, controlling, and tracking changes** in software, hardware, documents, test artifacts, and other project deliverables throughout the software development lifecycle (SDLC).

It ensures that:

* The **right version** of the right component is used.
* All changes are **controlled, recorded, and traceable**.
* Teams work with the **most up-to-date and approved versions**.

## **🎯 Objectives of Configuration Management**

**# Source**: **Chatgpt (GPT-5-Standard) at [8/18/2025]**

1. **Version Control** → Maintain different versions of code, documents, and test artifacts.
2. **Change Control** → Ensure changes are properly reviewed, approved, and documented.
3. **Consistency** → Keep test environments and builds stable and reproducible.
4. **Traceability** → Track changes from requirements → design → code → test cases.
5. **Collaboration** → Enable teams to work on shared resources without conflicts.

## **📦 Key Elements of Configuration Management**

**# Source**: **Chatgpt (GPT-5-Standard) at [8/18/2025]**

1. **Configuration Items (CIs)**
   * The entities under configuration control (e.g., requirements, code files, test cases, test data, environments, documentation).
2. **Configuration Identification**
   * Defining which items will be controlled and assigning unique IDs.
3. **Configuration Control**
   * Managing changes through a **Change Control Board (CCB)** or approval process.
4. **Configuration Status Accounting**
   * Recording and reporting the status of configuration items (e.g., version, who changed it, when).
5. **Configuration Audits**
   * Verifying that items are correct, consistent, and follow defined standards.

## 🔧 Examples of Configuration Management Tools

**# Source**: **Chatgpt (GPT-5-Standard) at [8/18/2025]**

* **Version Control Systems**: Git, GitHub, GitLab, Bitbucket, SVN.
* **Build & Release Tools**: Jenkins, Maven, Gradle.
* **Test Artifact Management**: HP ALM, Jira, TestRail.
* **Environment & Infrastructure**: Docker, Kubernetes, Ansible.

## **✅ Why is CM Important in Testing?**

**# Source**: **Chatgpt (GPT-5-Standard) at [8/18/2025]**

* Keeps **test cases, test data, and test scripts** consistent.
* Prevents **version conflicts** (e.g., testing old code while developers deploy new code).
* Ensures **traceability** from requirements → test cases → defects.
* Supports **regression testing** by maintaining old builds.

## 📊 Example

**# Source**: **Chatgpt (GPT-5-Standard) at [8/18/2025]**

Imagine you are testing an **E-commerce website**:

* **Configuration Items**: Requirement docs, source code, test cases, test data, database scripts, test environment.
* CM ensures:  
  + You test the correct **build version**.
  + You use the correct **set of test cases** mapped to requirements.
  + Any change in requirements or test scripts is **tracked and approved**.

👉 **In short:** **Configuration Management** = The discipline of **tracking, controlling, and maintaining consistency** of software, documents, test assets, and environments → ensuring stability, quality, and traceability across the project.