Software Engineering
2UCCE501

Module 4

Module 4

System Implementation, Configuration Management & Risk Management

4.1 Packages & Interfaces: Distinguishing between classes versus interfaces. Exposing class & package interfaces.

4.2 Mapping Model to code, Mapping object models to Database schema.

4.3 Component & Deployment Diagrams: Describing dependencies.

4.4 Managing & Controlling Changes: Managing & Controlling versions.

4.5 Categories of Risks. Nature of risks, Types of risks, Risk identification, Risk assessment, Risk Planning and control, Risk Management, Evaluating risk to schedule, PERT technique.

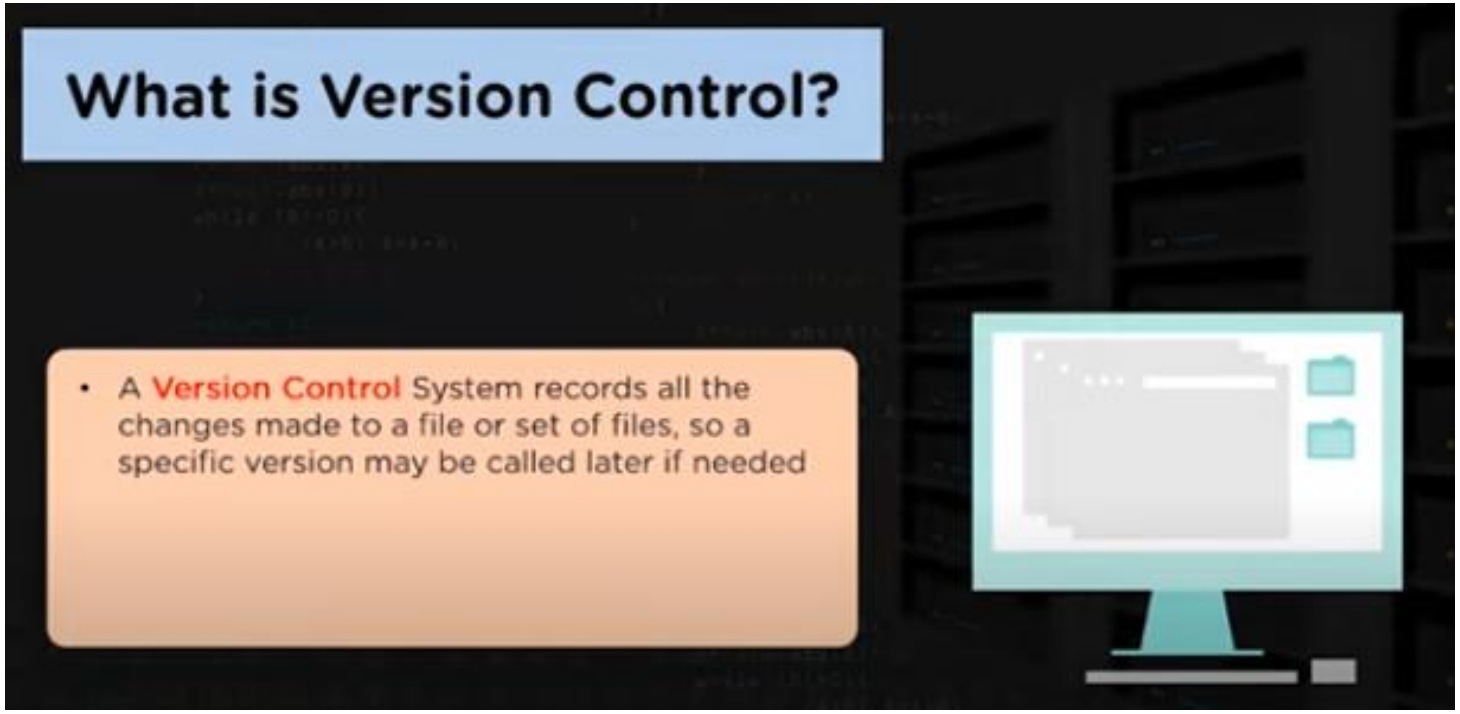
Packages & Interfaces

- **Packages**:
- Organize related classes and interfaces into a single unit.
- Declared using package keyword
- Does not support multiple inheritance.
- Group classes and interfaces based on functionality
- Syntax: package com.software.mypackage;

- Interfaces:
- Define a contract for classes to implement
- Declared using interface keyword.
- Define method signatures that classes implementing the interface must provide implementations.
- Interface methods are implicitly public and abstract.
- Supports multiple inheritance as a class can implement multiple interfaces

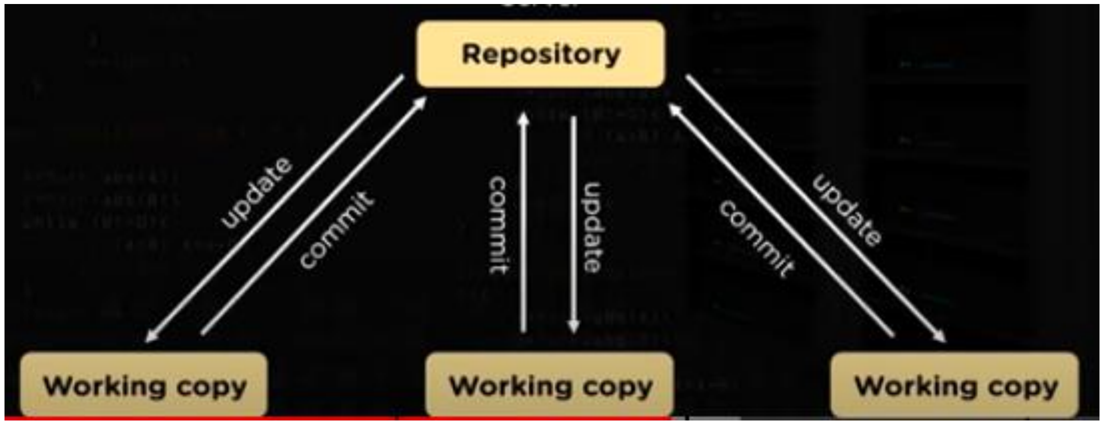
Version Control

- Challenges in distributed work environment:
  - Collaboration
  - Restoring previous version
  - What and where change happen
  - Backup
- A version control system is a kind of software that helps the developer team to efficiently communicate and manage(track) all the changes that have been made to the source code along with the information like who made and what changes have been made.



What is Version Control?

- A Version Control System records all the changes made to a file or set of files, so a specific version may be called later if needed



Version Control

- Version control **combines procedures and tools to manage different versions of configuration objects** that are created during the software process.

- A version control system implements four major capabilities:

(1) a **project database** that stores all relevant configuration objects

(2) a **version management capability** that **stores all versions** of a configuration object.

(3) a make facility that enables **construct a specific version of the software.**

(4) version control and change control systems often implement an **issues tracking (also called bug tracking)** capability

Version Control

- A number of version control systems **establish a change set**—a collection of all changes that are required to create a specific version of the software.
- **named change sets can be identified** for an application or system.
- construct a version of the software by specifying the change sets **(by name)**

Version Control

- Modeling approach for building new versions contains:

1. Template for building version
2. Construction rules
3. Verification rules

Change Control

- It is a systematic approach to manage all changes made to the product.
- Too much change control and we create problems.
- large software project, uncontrolled change rapidly leads to chaos(confusion).
- For **large projects** change control combines **human procedures and automated tools** to provide a mechanism for the control of change.

Change Control

- engineering change order (ECO).

- **elements of change management:**

1. **Access Control**

- Access control governs which software engineers have the authority to access and modify a particular configuration object.

Change Control

2. Synchronization Control.

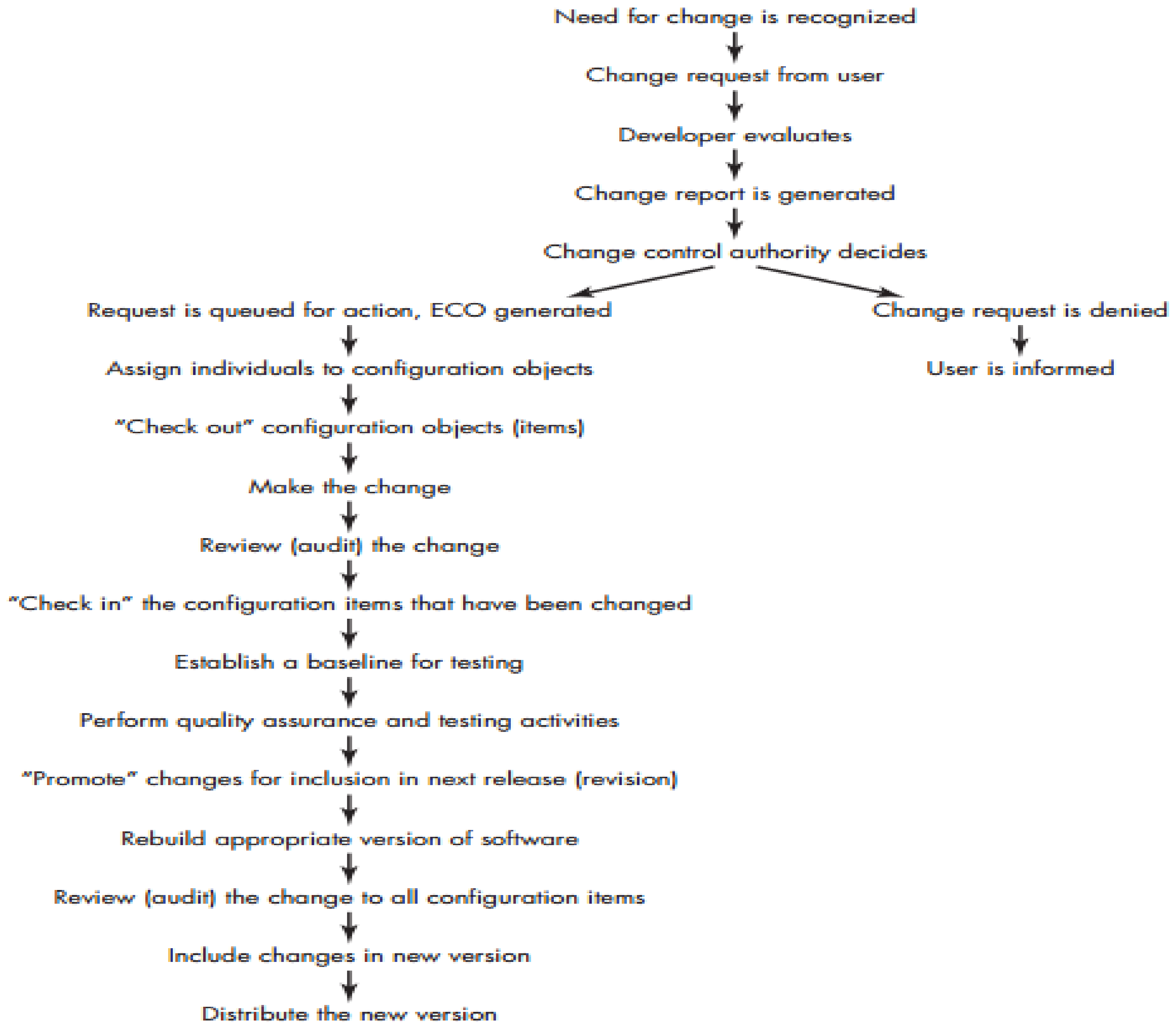
- Synchronization control helps to ensure that parallel changes, performed by two different people, don’t overwrite one another.

- **Informal Change Control**
- Developer makes changes in project

Change Control

- **project level change control**
- developer must gain approval from the project manager to makes changes.

- **formal change control**
- is instituted when the software product is released to customers


Need for change is recognized
Change request from user
Developer evaluates
Change report is generated
Change control authority decides
Request is queued for action, ECO generated
Change request is denied
User is informed
Assign individuals to configuration objects
"Check out" configuration objects (items)
Make the change
Review (audit) the change
"Check in" the configuration items that have been changed
Establish a baseline for testing
Perform quality assurance and testing activities
"Promote" changes for inclusion in next release (revision)
Rebuild appropriate version of software
Review (audit) the change to all configuration items
Include changes in new version
Distribute the new version