Software Configuration Management

Jana Kostičová, 14.3.2016

Software Configuration Management

= keeping order in all components of software product:

- Documents
- Code
- Data
- Environment
- …

Example scenarios:

- Which code version is deployed at the production environment?
- Which requirements are implemented in the latest code version?
- What is the current progress of fixing bug B-123?

Version Control

Why version control?

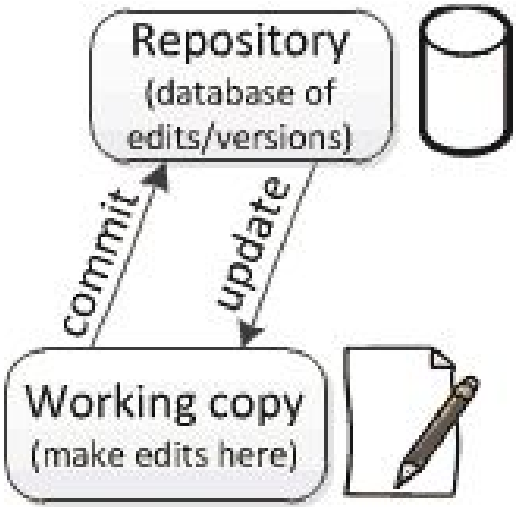
Paul and John are working on a homework as a team:

- Create python program which prints “Hello, world!” to the console

Paul’s directory contains the following files:

- Homework.py
- Homework2.py
- Homework2_fixed.py
- Homework_final.py
- Homework_final_John.py
- Homework_final_merged_John.py

How version control works



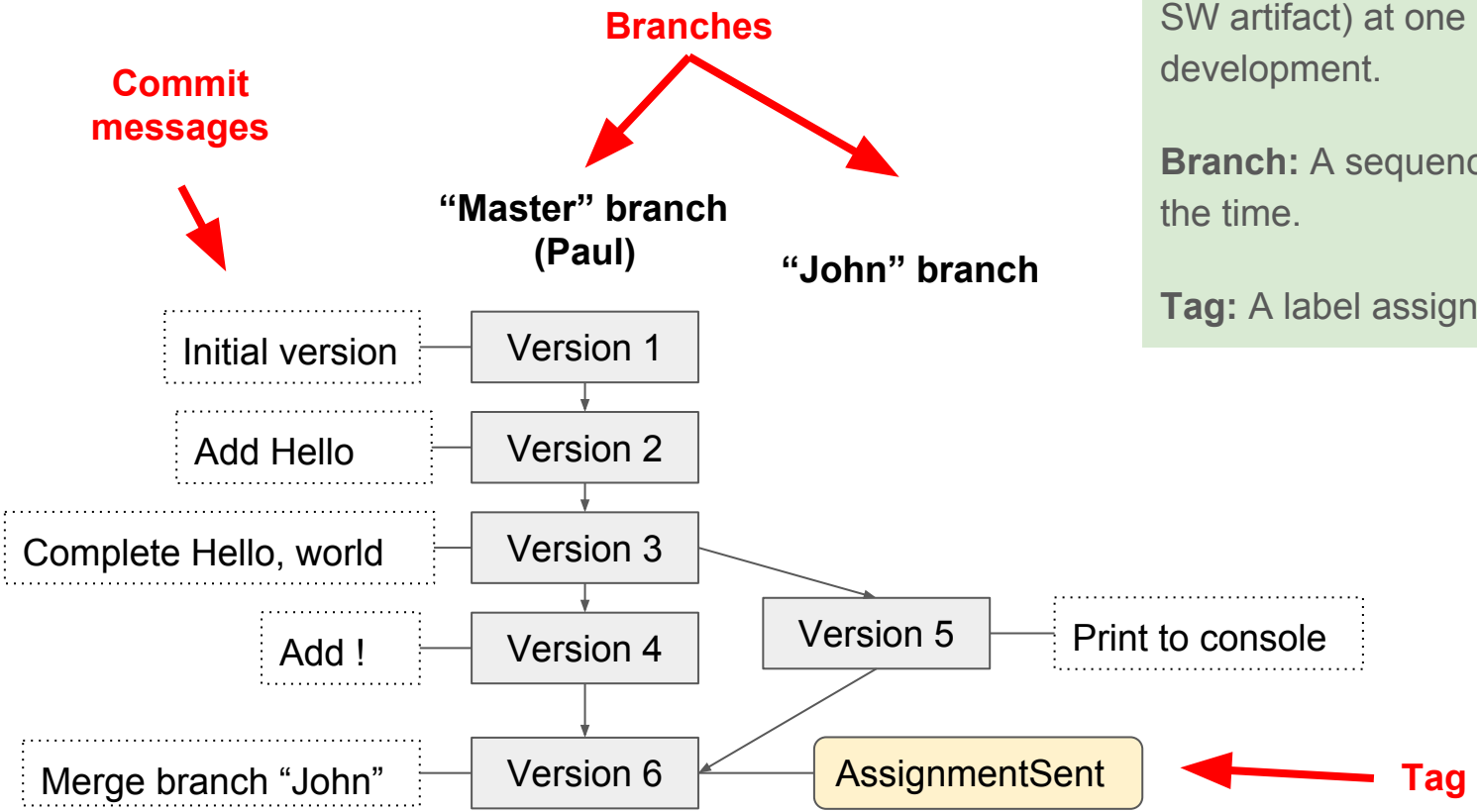
Repository: A database of versions.

Working copy: A personal copy of all the files in the project - here the user does his/her work.

Commit: An action of incorporating changes into the repository. Also a set of changes incorporated into the repository by single commit action.

Commit message: A descriptive text message attached to the commit.

Update: An action of updating working copy according to the content of the repository. In case a conflict appears, it is resolved automatically or by human interaction.



Version (revision) : Code (or other SW artifact) at one point in its development.

Branch: A sequence of versions in the time.

Tag: A label assigned to a version.

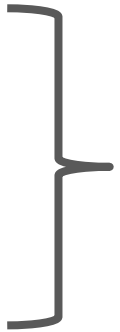
Version control

1. Manages concurrent changes to a single project

-> Team collaboration
- Multiple developers modify the same file

-> Using multiple computers to work on a project
- Workstation at work vs. laptop at home

-> Working on several product variations simultaneously
- Feature/bug branches
- Release branches
- Customization branches (target environments, customer,..)



Valuable also for one-man projects

Features:
- Atomic commits, branching, merge
- Conflict detection and resolution
- Locking - useful for some file types (e.g., binary format)

Version control

2. Keeps historical versions of the project

-> Revert back to a previous (stable) version in case a serious problem is found

-> Get information about previous versions

- Timestamp, author, list of changes,...

-> Keep additional information with versions

- Tags, commit message

Version control systems

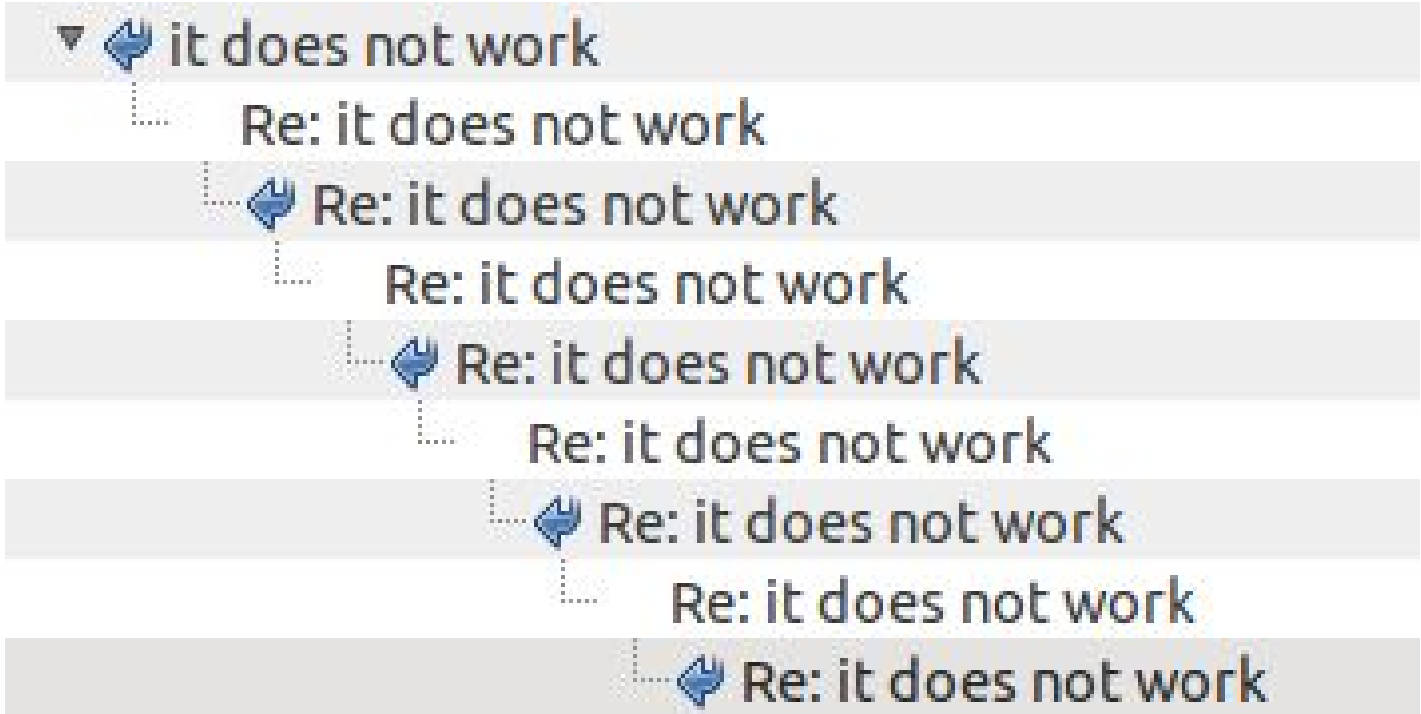
- Git, Mercurial, SVN, CVS, ...
- But also: Office Software (MS Office, Google Docs,..), SharePoint, Wiki,..

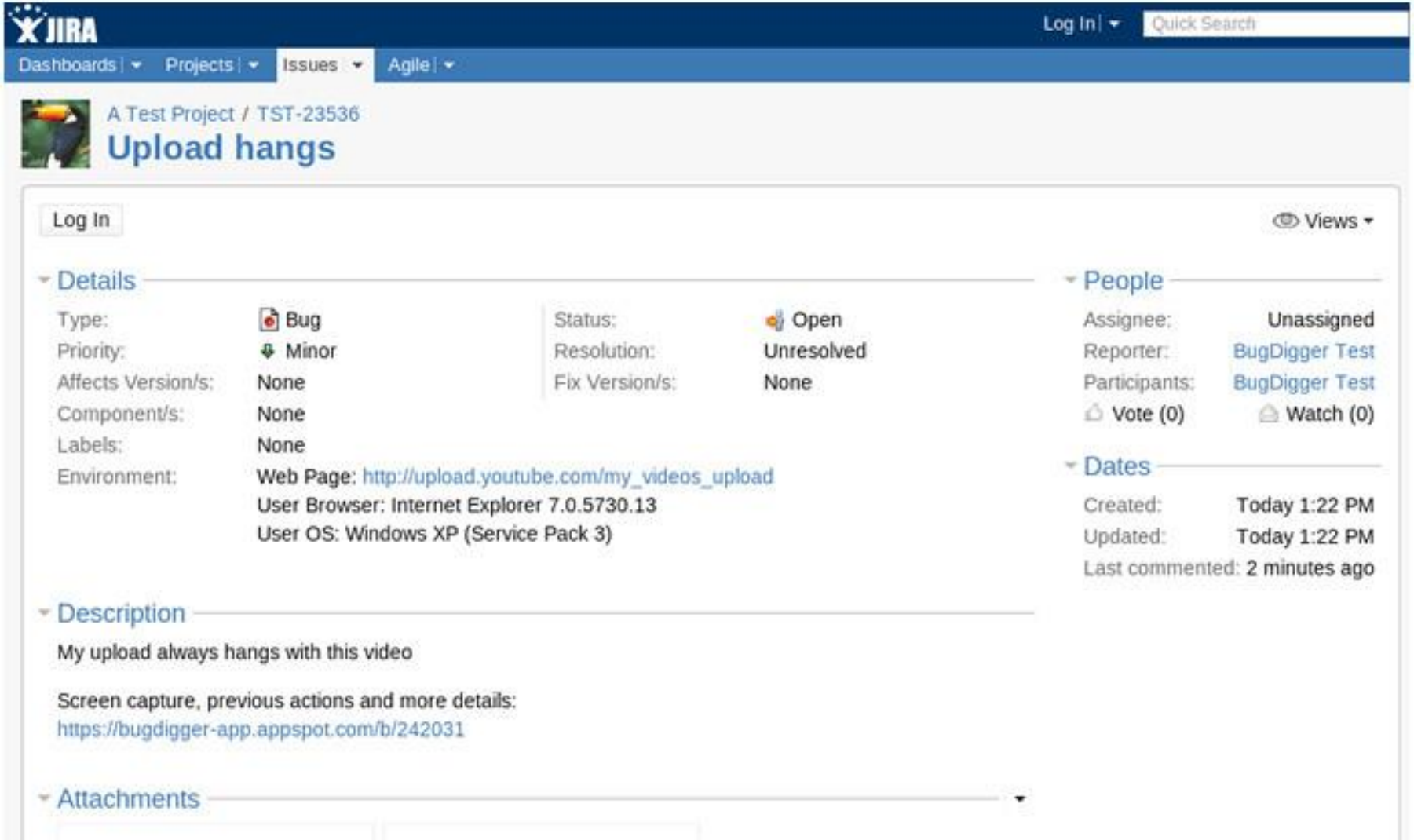
-> Not just for source code management!

  - Also for documentation, data,.. any other part of the software product

Issue tracking

Why issue tracking?




JIRA
Log In
Quick Search
Dashboards
Projects
Issues
Agile
A Test Project / TST-23536
Upload hangs
Log In
Views
Details
Type: Bug
Priority: Minor
Affects Version/s: None
Component/s: None
Labels: None
Environment: Web Page: http://upload.youtube.com/my_videos_upload
User Browser: Internet Explorer 7.0.5730.13
User OS: Windows XP (Service Pack 3)
Status: Open
Resolution: Unresolved
Fix Version/s: None
People
Assignee: Unassigned
Reporter: BugDigger Test
Participants: BugDigger Test
Vote (0)
Watch (0)
Dates
Created: Today 1:22 PM
Updated: Today 1:22 PM
Last commented: 2 minutes ago
Description
My upload always hangs with this video
Screen capture, previous actions and more details:
https://bugdigger-app.appspot.com/b/242031
Attachments

Issue tracking

1. **Keeping track of issues in one place in a uniform structure**

-> Unique identification of each issue

- Issue key

-> Monitoring issue status

- Open, in progress, resolved, closed, reopened, ...

-> Keeping additional information on issues

- Environment, component, priority, fix version, progress, ...

-> Grouping communication related to the issue

Issue: A system improvement or task that requires some work to be done such as new requirement, bug, enhancement, ..

Issue tracking

2. Managing issue life cycle

-> Distribution and assignment of issues to people in charge

-> Modification of issue attributes in accordance with defined workflow

- Status, assignee, resolution type,..

Issue tracking systems

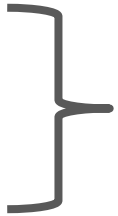
- Bugzilla, JIRA, Redmine, GitHub issues,...

-> Not just for development phase!

- Also for requirements analysis, design, testing, project management, .. any task within the project

Methods & tools summary

- Version control
  - Manages source code, documentation, data,...
- Issue tracking
  - Manages issues - new requirements, bugs, ...



It is also important to manage references between commits and issues
Example rule:
“Always reference resolved issues in commit message” or “Always reference corresponding commits from the issue”

- Build tools
  - Manage the build process
- Dependency management
  - Manages external dependencies (see scenario later)
- … other

-> SCM is not only about tools!
Key to success is to define appropriate rules (and follow them)

Software Configuration Management

Terminology

Software product:

- A complete set of computer programs, associated procedures, documentation, or data.

(Software) configuration item:

- Any identifiable part of a software product. A single unique entity for SCM.
- Examples:
  - Requirements
  - Design specification
  - Source code and executable code
  - Test specification, data, and records
  - User documentation
  - Library and supporting software
  - Bug reports
  - Environment configuration

Terminology

Software configuration management (SCM):

A discipline whose purpose is

- to identify the status of configuration items and
- to control changes throughout their life cycle.

SCM goals

- **Configuration identification**
  - Identification of all configuration items
  - Identification of the software product
- **Configuration control**
  - Ensuring a controlled change process which is correct with respect to the SW product.
- **Configuration status accounting**
  - Recording and reporting all the necessary information on the status of the configuration.
- **Configuration auditing**
  - Ensuring that configurations contain all their intended parts and are sound wit
    specifying documents, including requirements, architectural specifications and user manuals.

Scenarios & example solutions

-> Which version of code is deployed at dev/test/prod environment?

- Use tags in Version control

-> Which new requirements are implemented in this version of the source code?

- Collect issue references from commits between last release and the current version. Filter issues with type “New requirement”.

-> What is the current progress of fixing bug B-123?

- Look for the corresponding issue and check its progress.

-> What bugs have been fixed in release 1.2?

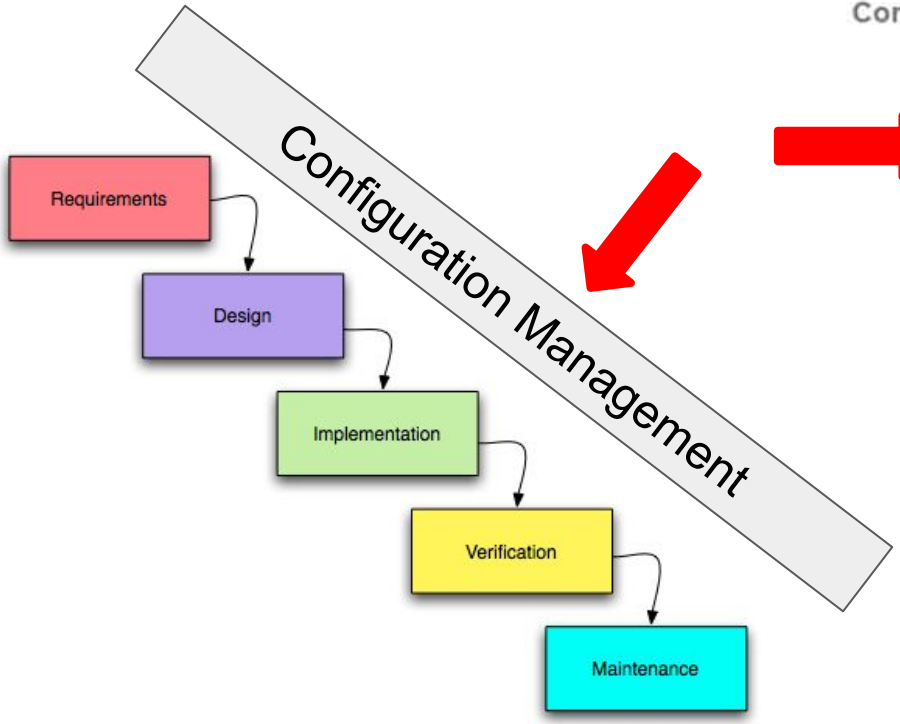
- Filter issues having “Status” set to “Resolved/Closed” and “Fixed version” set to “1.2”

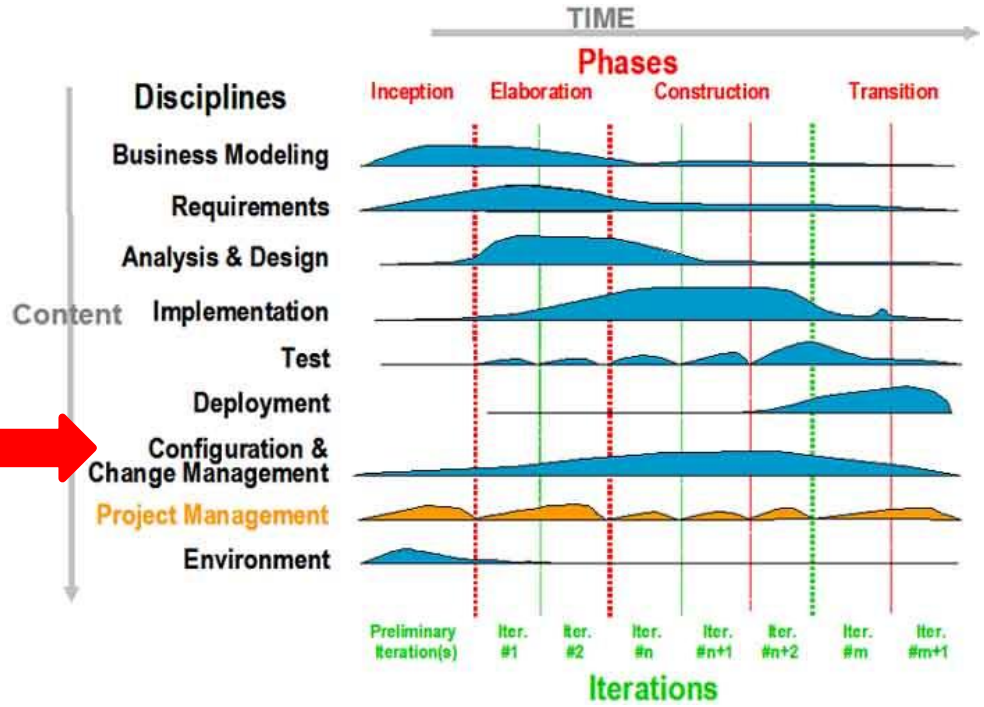
Scenarios & example solutions

-> Which external libraries are used in this version of code?

- The code uses external library - file “mysuperlib.jar”.
- We find out that version 3.1 of mysuperlib.jar has a serious vulnerability which has been fixed in version 4.0.
- **Problem:** What version of the library does our code use?
  -> Simple: Name the file e.g., mysuperlib-3.1.0.jar,
  -> Advanced: Use dependency management tool such as Maven.

SDLC


Requirements
Design
Implementation
Verification
Maintenance
Configuration Management


TIME
Phases
Inception
Elaboration
Construction
Transition
Disciplines
Business Modeling
Requirements
Analysis & Design
Content
Implementation
Test
Deployment
Configuration & Change Management
Project Management
Environment
Preliminary Iteration(s)
Iter. #1
Iter. #2
Iter. #n
Iter. #n+1
Iter. #n+2
Iter. #m
Iter. #m+1
Iterations

Summary

- SCM tracks and controls changes in software configuration items
- SCM operates through the whole SW lifecycle

- Not just version control
- Not just for source code management
- Not only for development phase
- Tools: Version control, Issue tracking, Build tools, Dependency management, Document management, ..

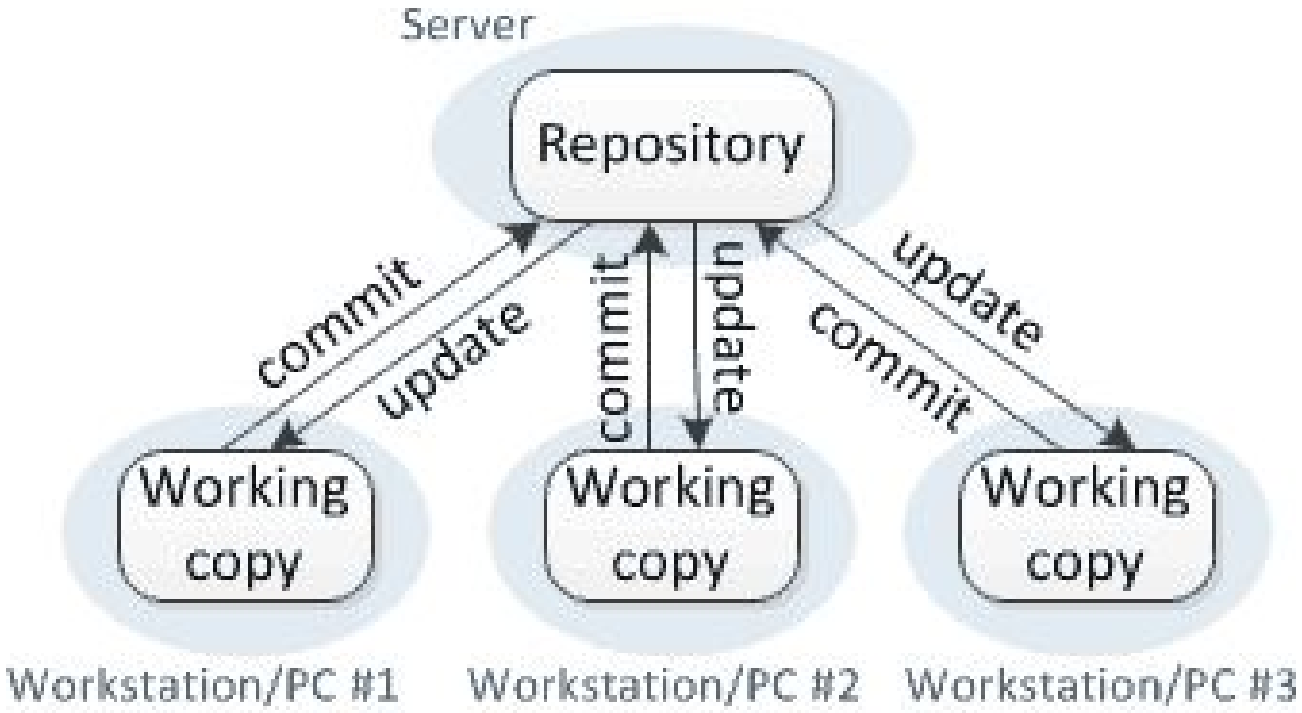
- Tools are important, but design and management of CM process are more crucial to project success

Further reading

- Ian Sommerville: Software Engineering (10th edition)
- ISO IEC 90003 2004 - quality standard for software products
- IEEE 828 standard
- Version Control with Subversion, Ben Collins-Sussman et al.

Centralized version control

Example:
SVN, CVS



Distributed version control

Example: GIT