



Software Engineering and Architecture

SCM Versioning



Concepts I: Versioning

Theory of SCM

Definition: Software configuration management

Software configuration management (SCM) is the process of controlling the evolution of a software system.

The image shows a Windows desktop with two windows open. The left window is titled 'cave' and shows a file tree for 'crunch3'. It includes a 'FILE STATUS' sidebar, a 'BRANCHES' sidebar (with 'master' selected), and a 'REVISIONS' sidebar (with 'origin' selected). The main pane displays a commit history for the 'master' branch, with the most recent commit being 'Added resources helpful in making screenshots for research papers' (revision 4546). The right window is titled 'D:\proj\Book - Log Messages - TortoiseSVN' and shows a list of log messages. The log messages are as follows:

Revision	Actions	Author	Date	Message
4546		hbc@cs.au.dk	16. maj 2017 18:07:53	fixed broken fnpp of hello-spark code
4545		hbc@cs.au.dk	16. maj 2017 16:46:17	Added source code for hello-spark project, using ivy
4544		hbc@cs.au.dk	15. maj 2017 16:01:40	Added src folder for dependency management
4543		hbc@cs.au.dk	11. maj 2017 13:17:32	Clean up, Session 10, Final version
4541		hbc@cs.au.dk	11. maj 2017 12:51:08	Clean up, Session 8, intro the isValidMove method
4540		hbc@cs.au.dk	10. maj 2017 16:10:48	Written TeleMed case chapter in near complete version (lacks refs etc.) Updated code basis
4538		hbc@cs.au.dk	10. maj 2017 14:53:00	Preparation for SWEA E17 usage. Removed Mongo DB references.
4537		hbc@cs.au.dk	10. maj 2017 14:53:00	

Below the log messages, there is a table showing a single path entry: 'D:\Book\trunk\resources-2\PaSOOS-Slides\W7-1 SOA.pptx' with action 'Added'.

Configuration

Definition: Configuration item

A configuration item is the atomic building block in a SCM system. That is, the SCM system views a configuration item as a whole without any further substructure. A configuration item is identified by a name.

Read: File

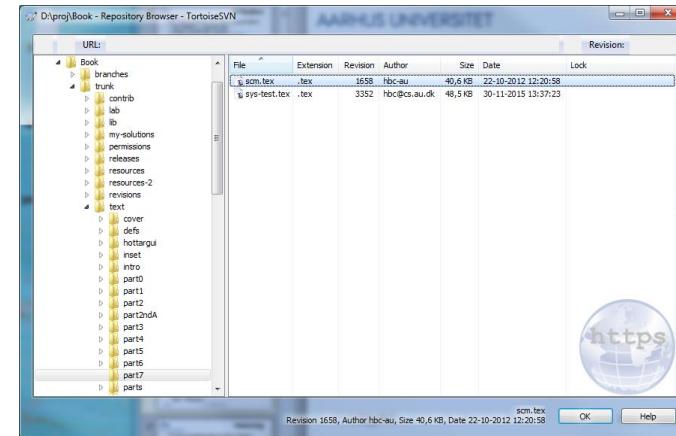
Definition: Configuration

A configuration is a named hierarchical structure that aggregates configuration items and configurations.

Read: Folder

- Think *Composite Pattern*

- *Git and SubVersion uses*
 - *File = Configuration Item*
 - *Folder = Configuration*



- The main purpose is to track evolution: *time focus*
- Another term for SCM is *Version Control*

Definition: **Version**

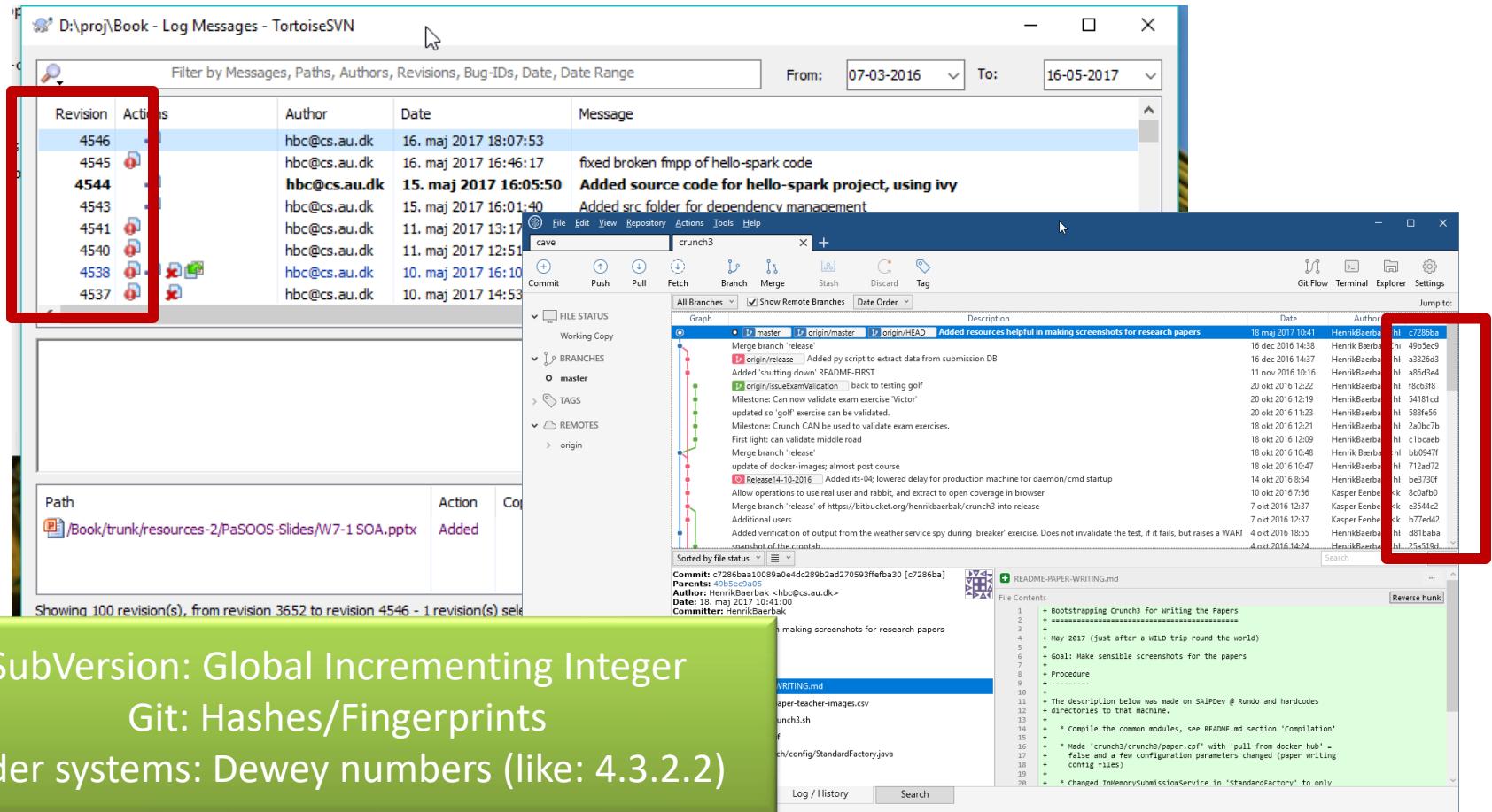
A version, v_i , represents the immutable state of a configuration item or configuration at time t_i .

- To give us a ‘handle’/‘name’ of a version we need

Definition: **Version identity**

A version is identified by a version identity, v_i , that must be unique in the SCM system.

Version Examples



The screenshot shows a desktop environment with two main windows:

- TortoiseSVN Window:** The title bar reads "D:\proj\Book - Log Messages - TortoiseSVN". It displays a log of 100 revisions from revision 3652 to 4546. The revision 4544 (15. maj 2017 16:05:50) is highlighted with a red box and has a detailed message: "fixed broken fmpp of hello-spark code" and "Added source code for hello-spark project, using ivy". The log table includes columns for Revision, Actions, Author, Date, and Message. A "Path" table below shows a file path: "D:\proj\Book\trunk\resources-2\PaSOOS-Slides\W7-1 SOA.pptx" with an "Added" action.
- Terminal Window:** The title bar reads "cave crunch3". The window contains a command-line interface with several commands and their outputs, including "Commit", "Push", "Pull", "Fetch", "Branch", "Merge", "Stash", "Discard", and "Tag". Below the commands is a "FILE STATUS" section showing a "Working Copy". To the right is a "Graph" visualization of the repository's history, showing branches like "master" and "origin" and their commits. A "REMOTE" section lists "origin". At the bottom of the terminal window is a "README-PAPER-WRITING.md" file with its content displayed.

SubVersion: Global Incrementing Integer

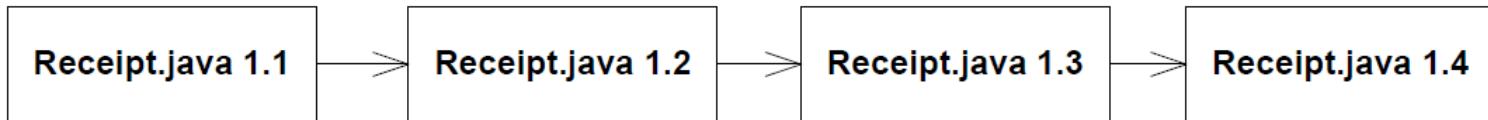
Git: Hashes/Fingerprints

Older systems: Dewey numbers (like: 4.3.2.2)

- Evolution is tracked over time

Definition: Version graph

A version graph is an oriented graph that shows the ancestor-relation between versions of an entity.



- One advantage of integers/dewey over hashes
 - Version 7 is *before* Version 8 and Version 21987
 - Version 8 is *after* Version 3

Check-in/Check-out

- Operations

- Check-in / Commit:
- Check-out / Update:

Snapshot in time **to** repository
Snapshot in time **from** repository

Definition: Commit

A commit is an operation that

1. generates a new, unique, version identity for the given entity.
2. stores a copy/snapshot of the entity under this identity.

Sigh! Git uses term
'checkout' for something
completely different!

Definition: Check-out

A check-out is an operation that, given a unique version identity, is able to retrieve an exact copy of an entity as it looked when the given version identity was formed during a commit.

*Note: Versions cannot
be deleted!*

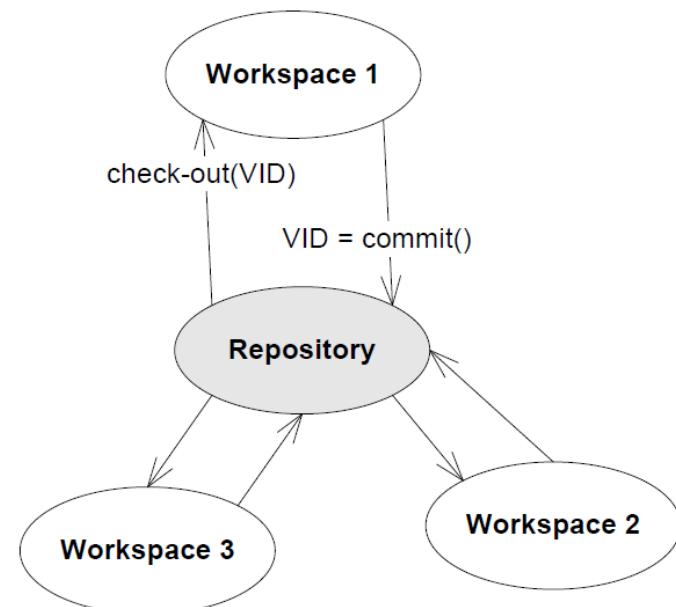
Definition: Repository

The repository is a central database, maintained and controlled by the SCM system that stores all versions of all controlled entities.

Definition: Workspace

A workspace is a local file system in which individual versions of entities can be modified and altered. Only one version of a given entity is allowed at the same time in the workspace.

- Each developer has his/her own workspace(s) in which modifications are made
- Repository is shared, collaboration is handled through it...



- OK – what have we got?
 - Versions made by commits into the repository; they form a version graph of identified versions that form a graph. This graph reflects the evolution over time.
 - I can checkout any version into my workspace for review.
- Basically **version/release management**
 - I can reproduce the exact codebase as it looked when I gave it to the customer...
 - if I
 - a) remember to commit it and
 - b) can remember the version identity