AARHUS UNIVERSITET

Software Engineering and Architecture

SCM Versioning

AARHUS UNIVERSITET

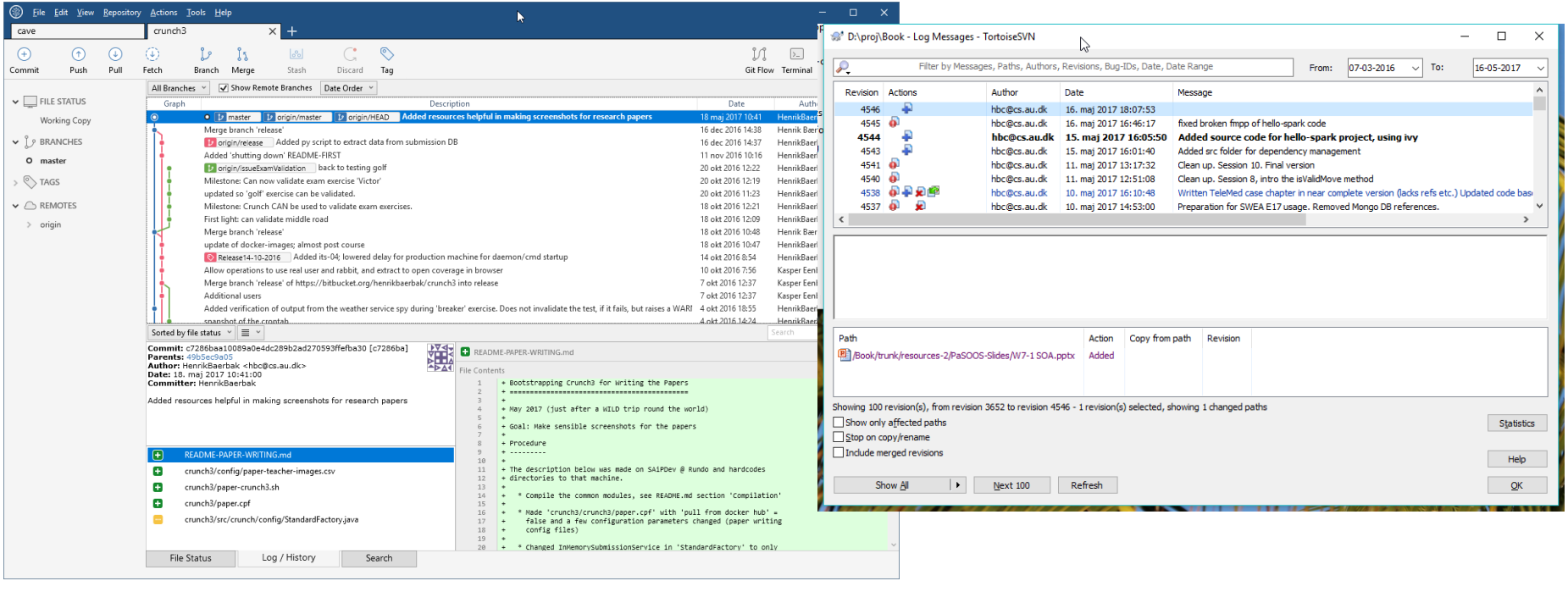
Concepts I: Versioning

Theory of SCM

SCM

Definition: **Software configuration management**

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



AARHUS UNIVERSITET

Configuration

Read: File

Definition: Configuration item

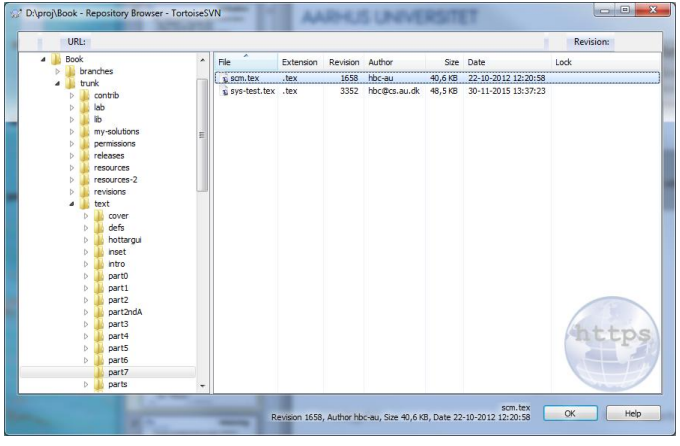
A configuration item is the atomic building block in a SCM system. That is, the SCM system views a configuration item as a whole without any further substructure. A configuration item is identified by a name.

Read: Folder

Definition: Configuration

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

- Think *Composite Pattern*
- *Git and SubVersion uses*
  - *File = Configuration Item*
  - *Folder = Configuration*



AARHUS UNIVERSITET

Versioning

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

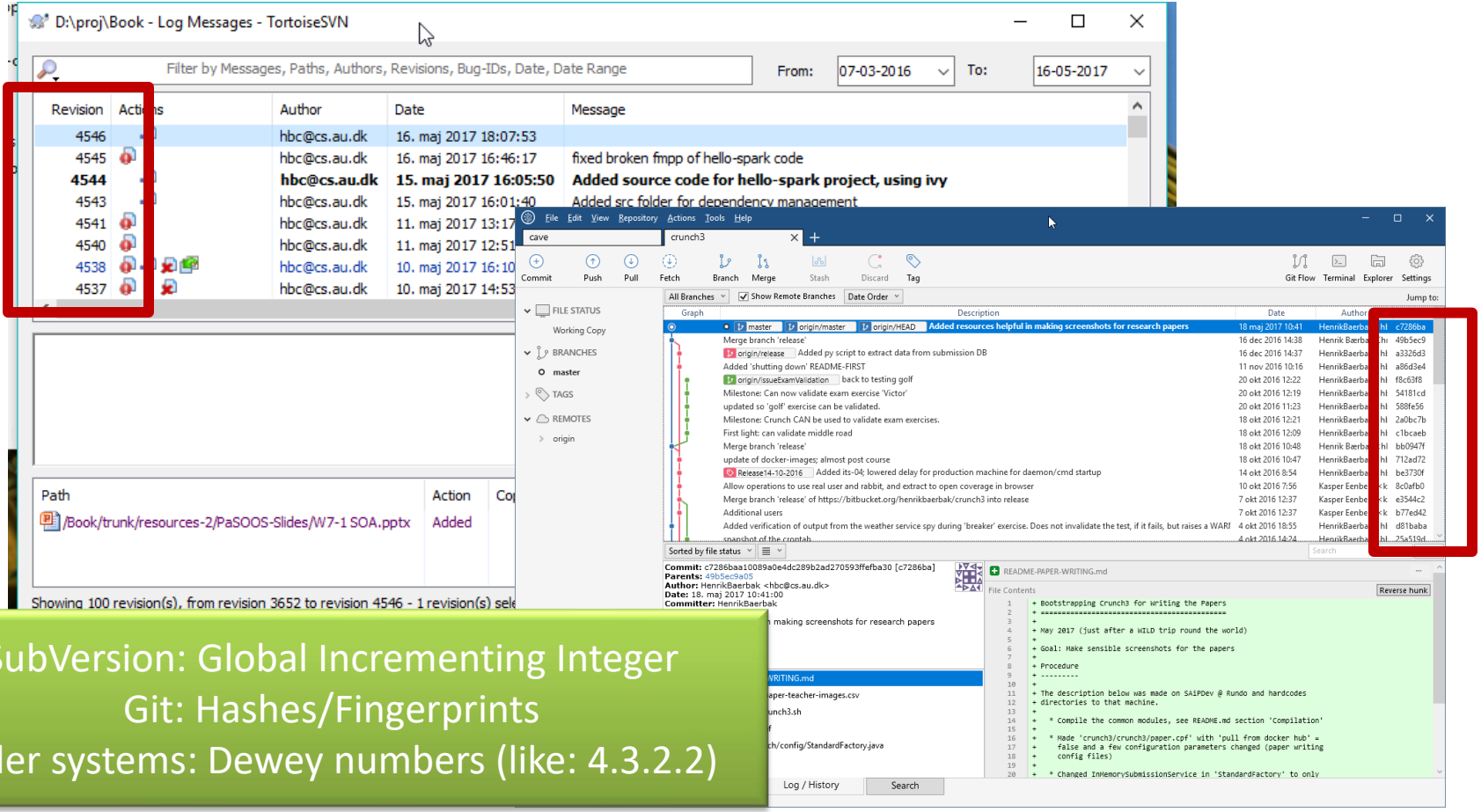
> Definition: **Version**
> A version, v_i, represents the immutable state of a configuration item or configuration at time t_i.

- 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.

AARHUS UNIVERSITET

Version Examples



SubVersion: Global Incrementing Integer

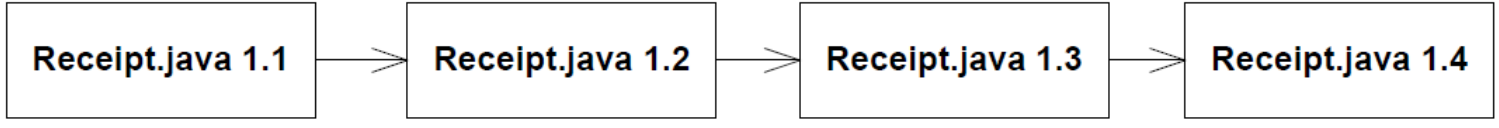
Git: Hashes/Fingerprints

Older systems: Dewey numbers (like: 4.3.2.2)

Evolution

- 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

AARHUS UNIVERSITET

Check-in/Check-out

- Operations
  - Check-in / Commit: Snapshot in time **to** repository
  - Check-out / Update: 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**

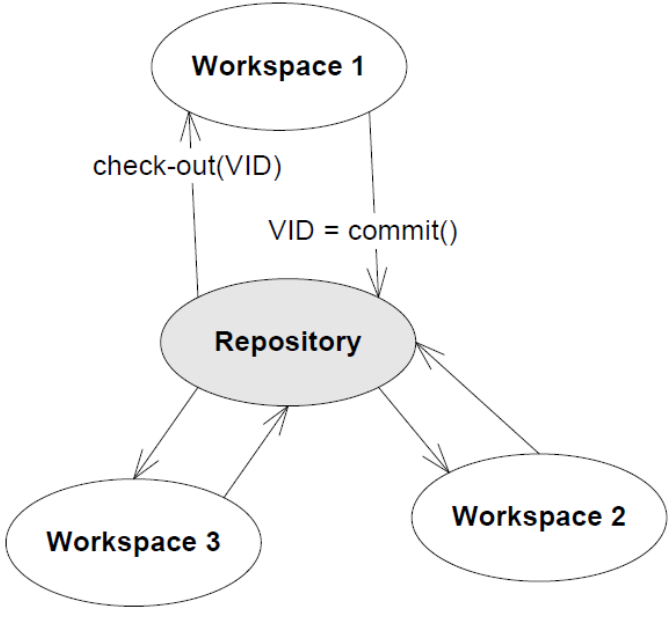
The repository is a central database, maintained and controlled by the SCM system that stores all versions of all controlled entities.

Workspace

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…



So - Status

- 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