AARHUS UNIVERSITET

Software Engineering and Architecture

SCM Collaboration

AARHUS UNIVERSITET

Concepts II: Collaboration

Theory of SCM

AARHUS UNIVERSITET

Collaboration

- *What do you get if you put two developers on a code base?*
  - *Answer: Conflicts!*

> Definition: **Conflict**
>
> A conflict is a situation where the same piece of code has been changed at the same time in two or more different workspaces.

- I.e. Arne and Bente check-out version 17 of Receipt.java
- *And both edit and change the same part of the code!*

- Question: Whose edits are the correct ones to check-in?

Concurrency

> Definition: **Pessimistic concurrency**
>
> Ensure strict sequential modifications by *locking* configuration items during modification.

- You ask *permission* by the SCM system to *edit* a file
- If granted, you ‘own’/’lock’ the file
  - Anyone else will be rejected!
- Outcome: No conflicts ever occur…
- Popular in early SCM systems, like RCS

Concurrency

Definition: Optimistic concurrency

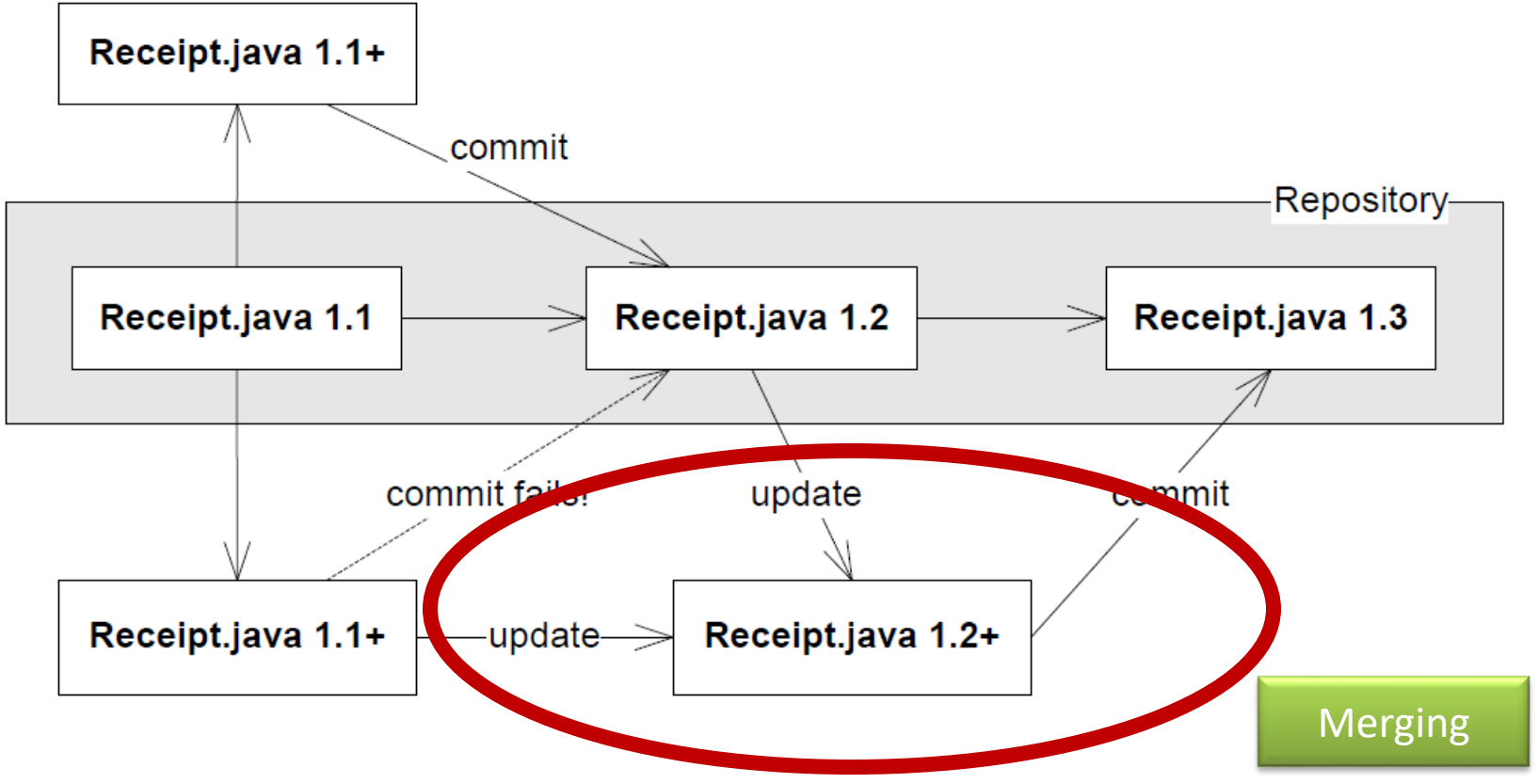
Allow for parallel modification and handle conflicts by merging.

- Both may modify any file in their workspace
- *First* to check-in does so as usual
- *Second* to check-in are required to *merge* in case of *merge-conflicts*
  - *Review all places where Arne and Bente has made changes in the same code lines*

Definition: Merge

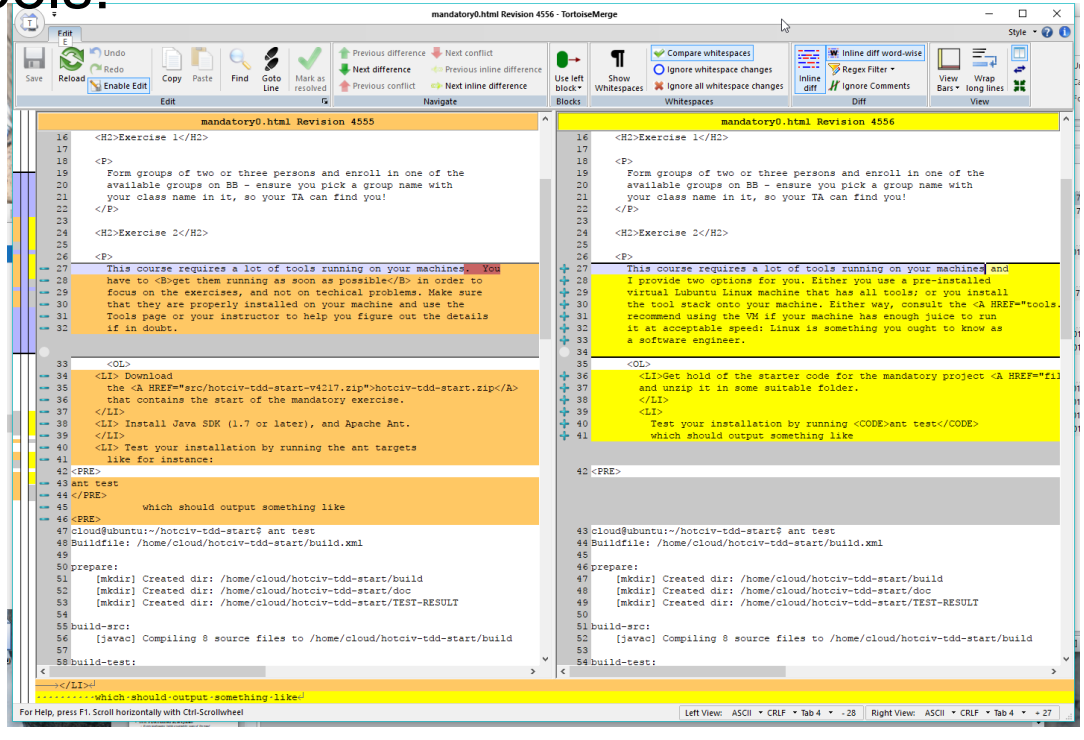
A merge is a operation where the sum of changes since the last common ancestor in the version graph is included in a configuration item or configuration.

Merging


Receipt.java 1.1+
commit
Repository
Receipt.java 1.1
Receipt.java 1.2
Receipt.java 1.3
commit fails!
update
commit
Receipt.java 1.1+
update
Receipt.java 1.2+
Merging

Merging is a Demanding Task!

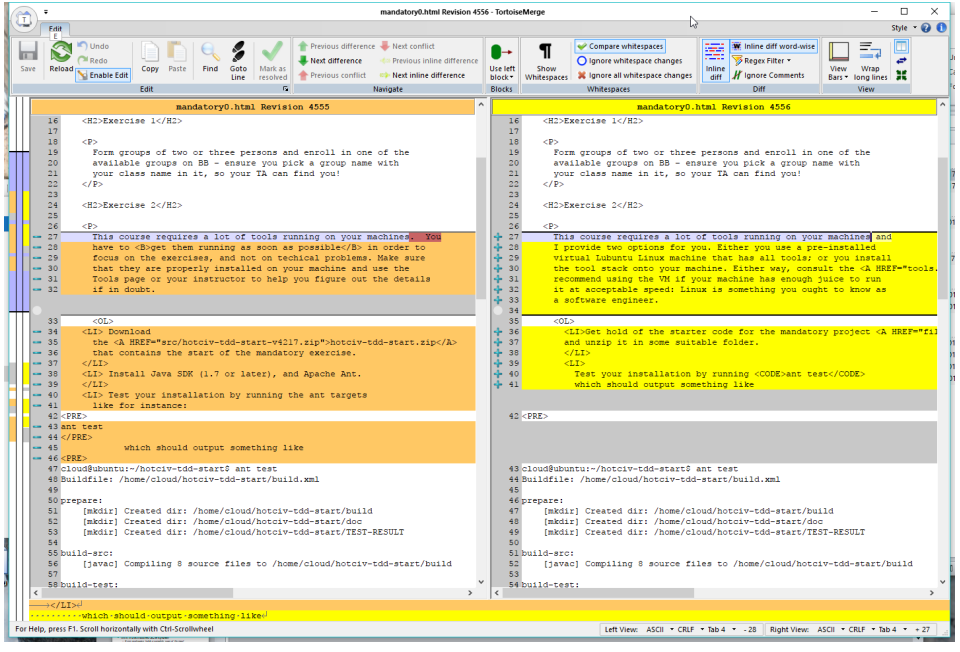
- In case of many **merge-conflicts**, you need help
- Use graphical merge tools!
  - Built-in
    - Tortoise SVN
    - SourceTree
    - …
  - WinMerge
- WarStory
  - One man **fulltime** to handle merge in local company!



Keep Merge-Conflicts Small

- WarStory
  - One man **fulltime** to handle merge in local company!
- Exercise: How to avoid?

- By *taking small steps!*
- Check-in often, merge often !!!
  - Hourly, not monthly!!!



Exercise

- SCM systems must handle *all aspects* of an application
  - Source code, build scripts, image files, sound files, configuration files…
- Which types of files are best handled using
  - Pessimistic concurrency?
  - Optimistic concurrency?

AARHUS UNIVERSITET

So - Status

- OK – so what have we got?
  - Collaboration on configurations by either optimistic or pessimistic concurrency – through locking items or through a merge phase

- Basically **collaboration management**
  - The team can work together without overwriting code made by others, and are warned when there are conflicting edits

- *Note: SCM systems can only detect syntactic conflicts!*

Thus – SCM:

Definition: **SCM system**

A SCM system is a tool set that defines

1. A central repository that stores versions of entities.
2. A schema for how to setup multiple, individual, workspaces.
3. A commit and a check-out operation that transfer copies of versions between the repository and a workspace.
4. A schema for handling/defining version identities for configuration items and configurations.
5. A schema for collaboration/concurrent access to versions.