

*L3 Mention Informatique
Parcours Informatique et MIAGE*

Génie Logiciel Avancé - Advanced Software Engineering

Part IV : Version and Configuration Management

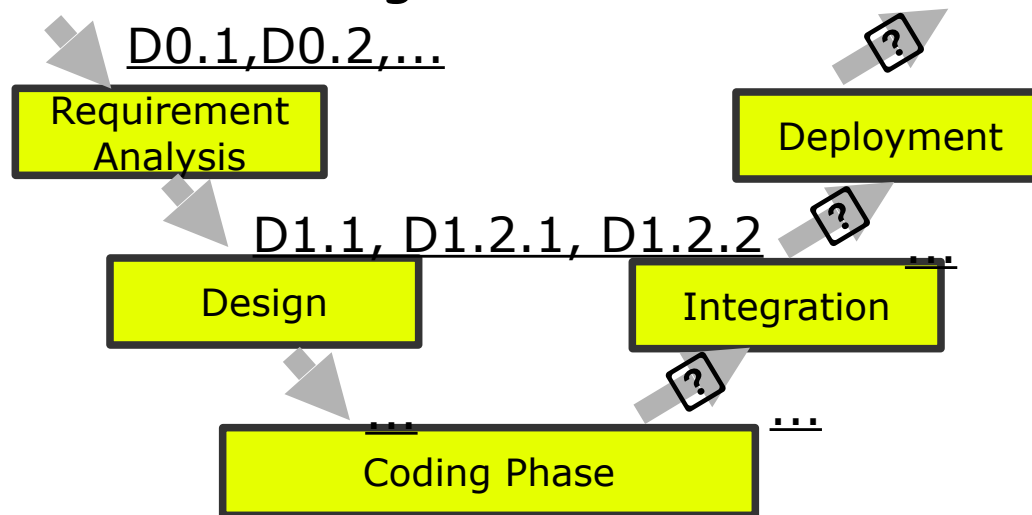
Burkhart Wolff
wolff@lri.fr

Plan of the Chapter

- ❑ Motivation: Version and Configuration Management as „the“ means for collaborative development
- ❑ Version management
 - Centralized Version Control
 - Distributed Version Control
 - Organizing Merges
- ❑ Beyond Versions: Configurations
 - Build Management
 - Advanced Configuration Management

Motivation

- Recall: SE Processes are based
 - on a large flow of documents and code
 - ... that have to be edited collaboratively
 - ... distributed consistently
 - ... while controlling access

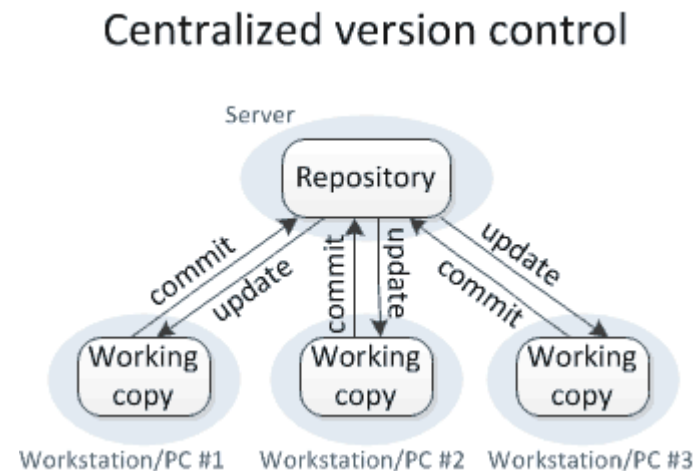


Motivation

- ❑ Important **TECHNICAL** means to support the process
 - Explicit tools for Version Management
(CVS, svn, git, mercurial, sourcedepot, ...)
 - Create and track **revisions** of files and file-trees
 - Create and track **differences** between files and file-trees
 - Access control to the various parties of process
 - Locking of documents
 - Merging of revisions
 - Quality control
 - ... actually, it is dead-useful for **everything** !!!

Concepts of Centralized Version Control

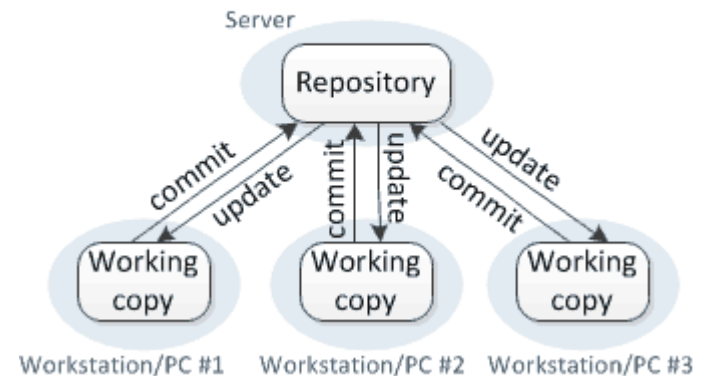
- ❑ Working copies (in user space)
- ❑ Repository (on the server-side)
- ❑ update: syncing with the repository
- ❑ commit: creating a new revision of a document (involves new registration, inclusion in documents, consistency checks)
- ❑ operations lock, checkout, import, ...



Concepts of Centralized Version Control

- ❑ First widely used system: *CVS*
- ❑ Nowadays in use : *svn*
- ❑ In connection with a gui-client: useable for end-users ...
... for **everything** ...
- ❑

Centralized version control



<https://subversion.apache.org/>



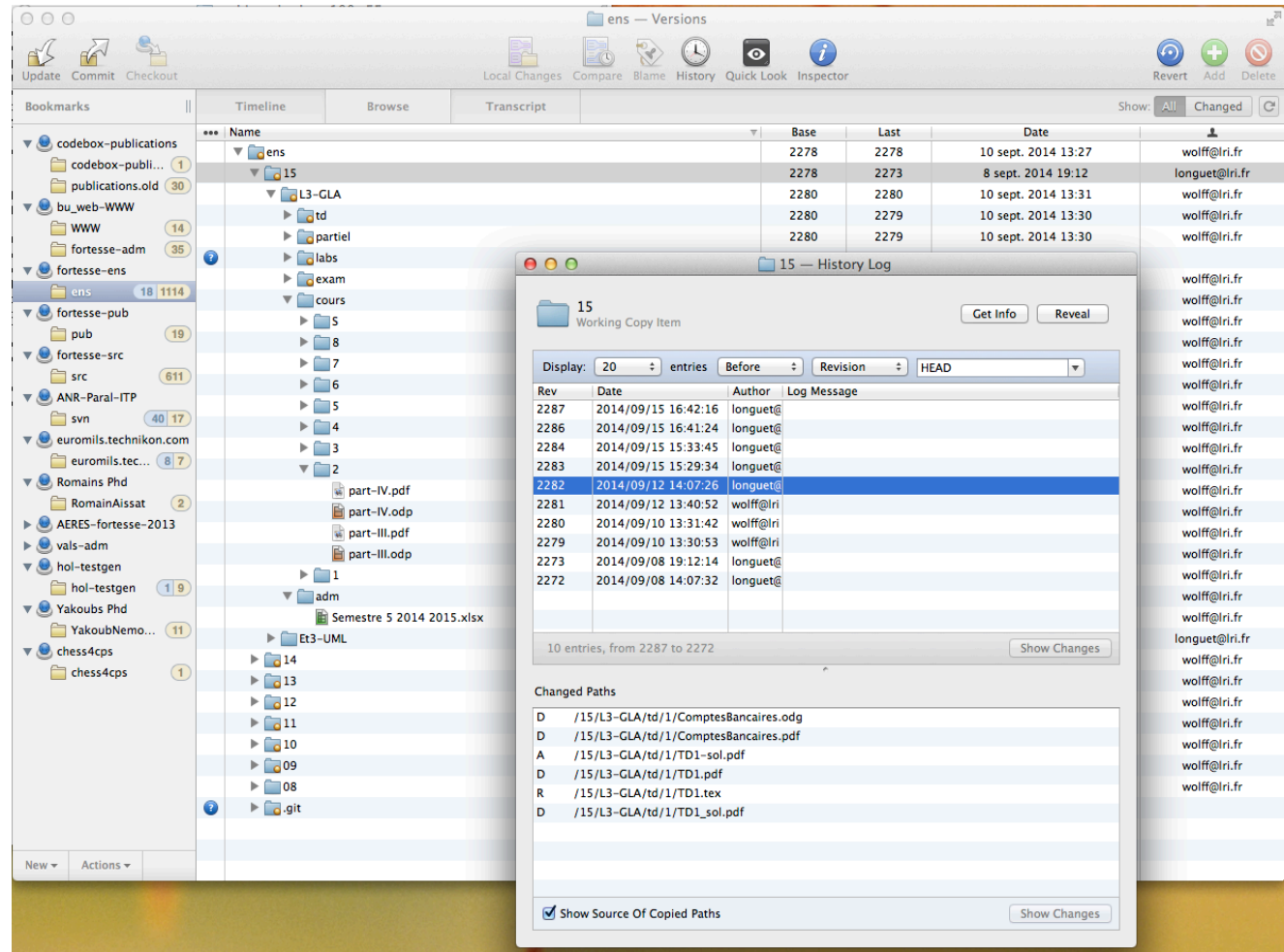
... for everything ...

my working copy for this course material

...

Version - management is not just for code

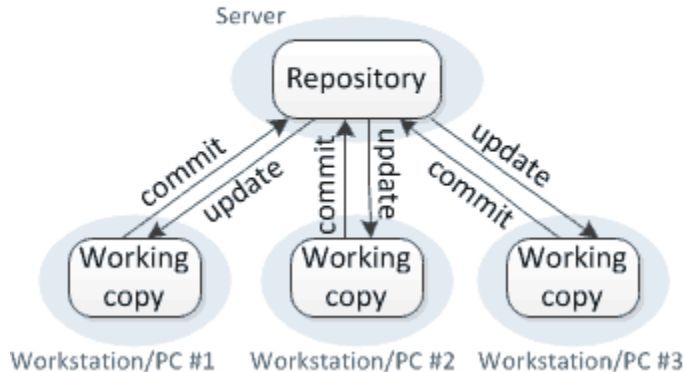
...



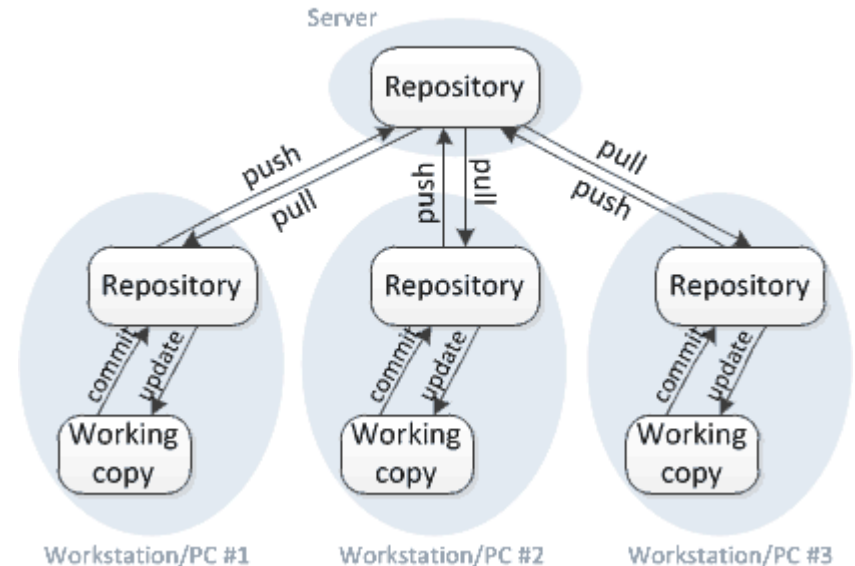
One step further:
Distributed Version Control

□ Hierarchy of repositories:

Centralized version control



Distributed version control



- one more sync-level, but more precise history in practice... since everybody can check in locally
- hierarchy strictly speaking not necessary

Distributed Version Control Systems

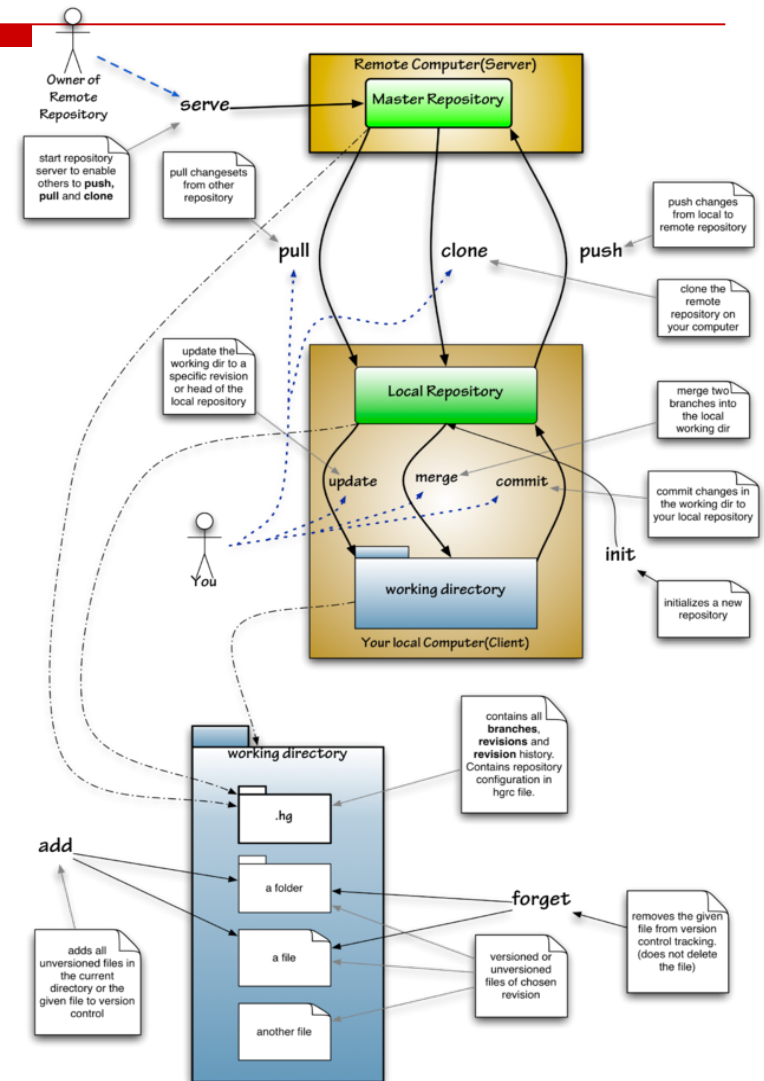
Open source:

➤ git (Linux)



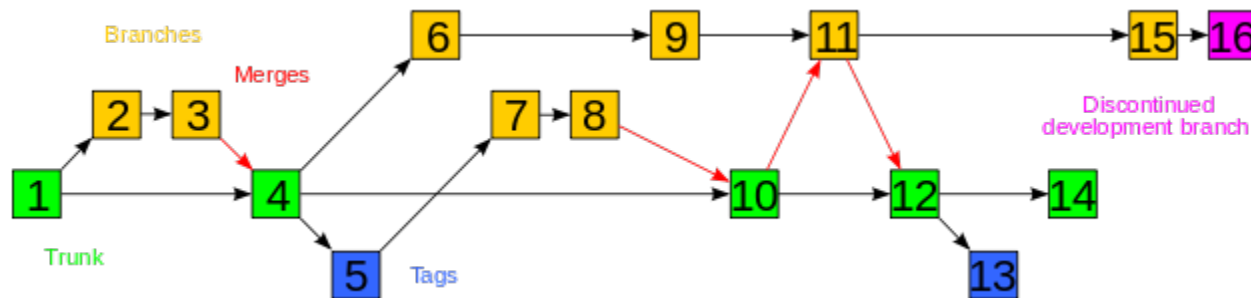
➤ mercurial (Isabelle)

<http://mercurial.selenic.com/>



Problems of a distributed development

- ❑ If not synced via explicit locks, a development looks like this:

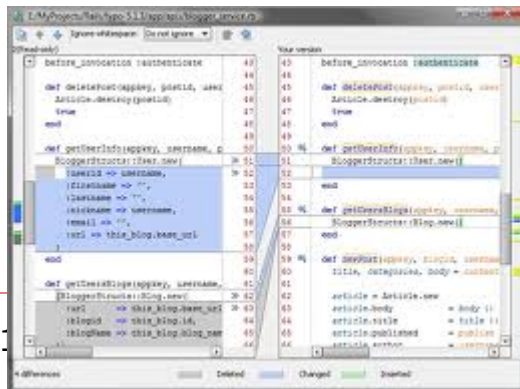
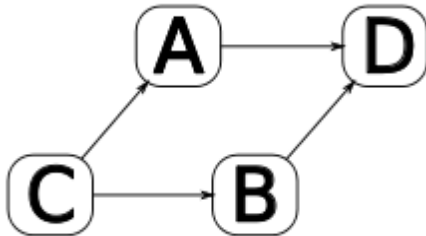


- ❑ Merging: For binary formats, only special purpose merges work (as in MS Word, for example ...)
- ❑ For textual formats :

A partial solution ...

❑ For textual formats:

A three-way merge is performed after an automated difference analysis between a file 'A' and a file 'B' while also considering the origin, or common ancestor, of both files. It is a rough merging method, but widely applicable since it only requires one common ancestor to reconstruct the changes that are to be merged.



A three-way merge
is performed after

A three-way merge
is performed after

A three-way merge
is performed after

A three-way merge
is performed after

Tools: For example xmerge, which also offer conflict resolution by hand ...

A better solution ...

- ❑ In a distributed development process, where a very large number of merges occurs routinely, the validation of the intermediate results becomes **crucial**.

(This limits the use of informal documents.)

- validation on any check-in
(for example: automated type-checking)
- validation for UML - documents
(self-defined consistency checkers for UML models)
- **automated static analysis and tests during systematic and periodic builds ...**
- ... more advanced methods, using proof techniques.

Build Management: A Build-Server

The screenshot shows the Jenkins web interface. At the top left is the Jenkins logo and the name 'Jenkins'. On the right, there is a search bar and the user name 'Burkhard Wolff'. Below the header, there are navigation links for 'People', 'Build History', and 'My Views'. On the left side, there are two panels: 'Build Queue' (showing 'No builds in the queue.') and 'Build Executor Status' (showing '1 Idle'). The main area displays a table of build jobs, filtered by 'Alle', 'Featherweight OCL', and 'HOL-TestGen'. The table has columns for 'S' (status), 'W' (weather icon), 'Name', 'Last Success', 'Last Failure', and 'Last Duration'. Each row represents a build job with its corresponding status icon, name, and timing information.

S	W	Name ↓	Last Success	Last Failure	Last Duration
		CPU (quick and dirty)	N/A	4 days 6 hr - #5	43 sec
		Featherweight-OCL	6 days 6 hr - #24	11 days - #18	3 min 52 sec
		Featherweight-OCL (quick and dirty)	7 hr 49 min - #12	N/A	3 min 42 sec
		HOL-IMP	2 days 5 hr - #5	19 days - #1	35 sec
		HOL-IMP - Examples	2 days 5 hr - #4	N/A	27 sec
		HOL-TestGen - add-ons - OS	3 hr 15 min - #12	10 days - #1	1 min 39 sec
		HOL-TestGen - Core	3 hr 22 min - #34	12 days - #22	31 sec
		HOL-TestGen - Examples - List	3 hr 18 min - #35	16 days - #15	1 min 4 sec
		HOL-TestGen - Examples - List Verified	3 hr 17 min - #24	18 days - #3	28 sec
		HOL-TestGen - Examples - max	3 hr 16 min - #33	9 days 3 hr - #24	55 sec
		HOL-TestGen - Examples - RBT	N/A	18 days - #3	48 sec

Build Management: A Build-Server

Jenkins search ? **Burkhard Wolff**

Jenkins > Featherweight-OCL [ENABLE AUTO REFRESH](#)

[Back to Dashboard](#)

[Status](#)

[Changes](#)

[Workspace](#)

[Build Now](#)

Project Featherweight-OCL

This Jenkins projects builds nightly the proposed Annex A for the OCL Standard 2.6 and later.

- [Workspace](#)
- [Recent Changes](#)
- [Latest Test Result \(8 failures / ±0\)](#)

Document links

- [Proof Outline](#)
- [Proof Document](#)
- [Annex A](#)

Upstream Projects

- [Featherweight-OCL \(quick and dirty\)](#)

Permalinks

- [Last build \(#24\), 6 days 8 hr ago](#)
- [Last successful build \(#24\), 6 days 8 hr ago](#)
- [Last failed build \(#18\), 11 days ago](#)
- [Last unstable build \(#24\), 6 days 8 hr ago](#)
- [Last unsuccessful build \(#24\), 6 days 8 hr ago](#)

Build History

#	Time
#24	Sep 10, 2014 2:36:00 AM
#23	Sep 6, 2014 5:44:26 PM
#22	Sep 5, 2014 6:09:33 PM
#21	Sep 5, 2014 6:29:28 AM
#20	Sep 5, 2014 12:14:47 AM
#19	Sep 4, 2014 11:39:52 PM
#18	Sep 4, 2014 6:21:38 PM
#17	Sep 4, 2014 5:12:55 PM
#16	Sep 4, 2014 1:57:50 PM
#15	Sep 4, 2014 1:54:15 PM
#14	Sep 4, 2014 12:12:01 PM
#13	Sep 4, 2014 10:08:02 AM
#12	Sep 4, 2014 9:58:40 AM
#11	Sep 3, 2014 11:08:58 PM
#10	Sep 3, 2014 10:56:24 PM
#9	Sep 3, 2014 10:37:27 PM

Test Result Trend

(just show failures) enlarge

Towards Configuration Management

- ... generalizes Version-Management

- by configuration descriptions
(including functionality environment, hardware,...)
- attempts to GENERATE a revision on the basis of meta-data over dependencies and change-sets
- works with heavy virtualization techniques nowadays (Google, Microsoft)

