A Journey in Software Development

An overview of methods and tools (part 3)

Mathieu Acher

Maître de Conférences

mathieu.acher@irisa.fr

UNIVERSITÉ DE RENNES 1

istic Informatique Électronique

Material

https://github.com/acherm/teaching/tree/master/PDL/

Previously

Document, refactor… Execute your tests… Debug.. Write test..

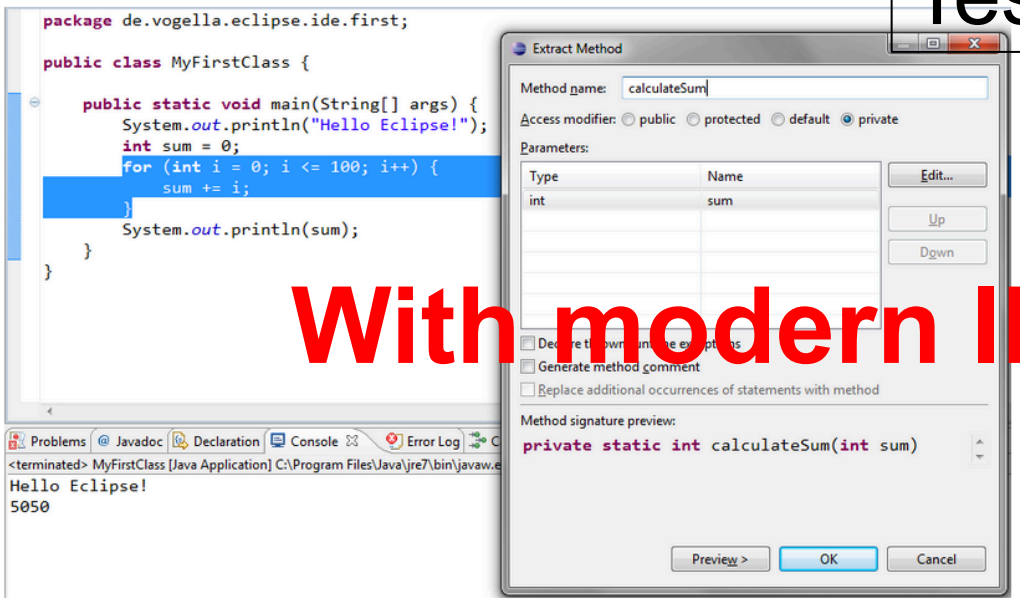
And so on!

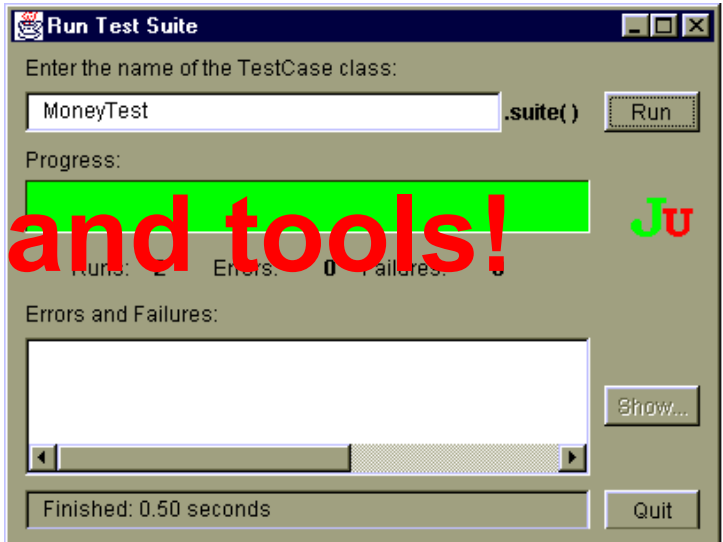
Documenting

Debugging

Refactoring

Testing





With modern IDE and tools!

PDL

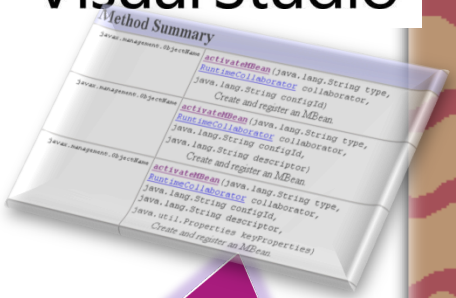
- Resources
  - refactoring.com
  - http://junit.sourceforge.net/doc/cookstour/cookstour.htm (How Junit is implemented)

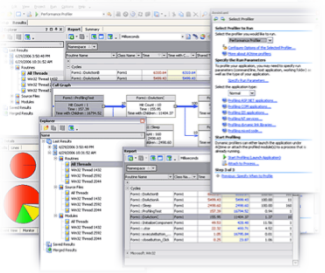
- Good news
  - 20th december for C1

Versioning in a nutshell

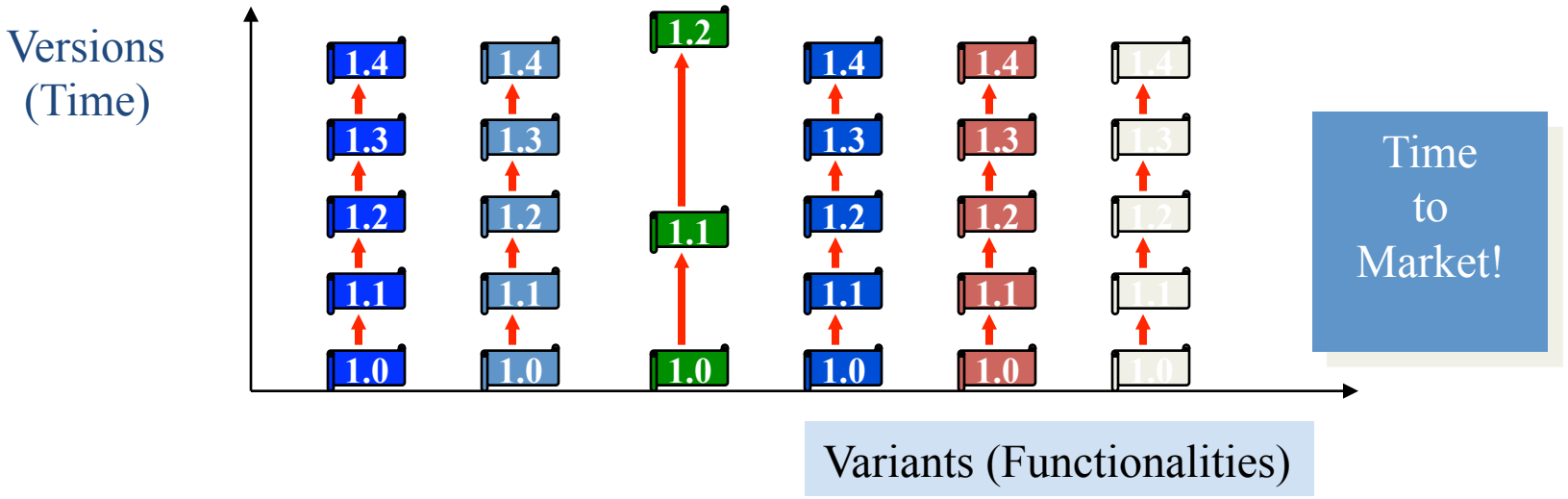

4°

Software Engineering





Software Engineering (back)



The 3 Dimensions of Software Configuration Management

[Estublier et al. 95]

- The Revision dimension
  - Evolution over time
- The Concurrent Activities dimension
  - Many developers are authorized to modify the same configuration item
- The Variant dimension
  - Handle environmental differences

- Even with the help of sophisticated tools, the complexity might be daunting
  - Try to simplify it by reifying the variants of an OO system

Versioning of source code

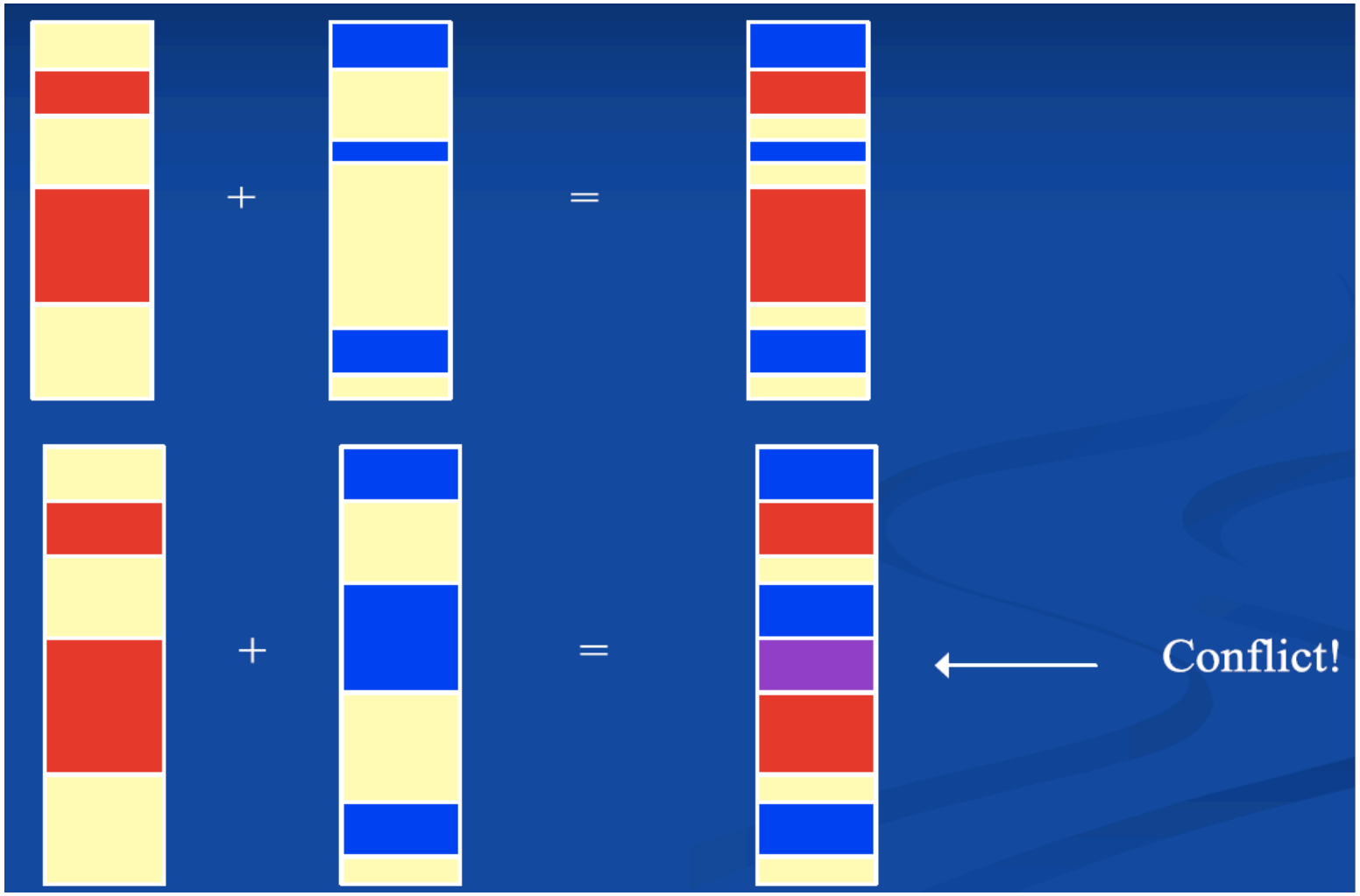
- **Collaborative** software engineering
- To master
  - software development by very large developer teams
  - parallel implementations (experiments, vendors)
- Goals
  - Increase productivity of developers and software robustness
  - Low-down development costs
- Manage software system configuration
  - to control software systems evolution
  - evolution tracking (time-machine)
  - issue and bug tracking

Versioning : What for?

- **History** of versions
  - back to an older version in case of errors
- **Alternative** versions (branching)
  - different design/implementations (maybe experimentals) for the same module
- **Collaborative** access by many developers
  - audit modification history
    - how many commits by X ?
    - when most of the commits are done?
    - ...

Concurrency management

with a simple picture



Concurrency control

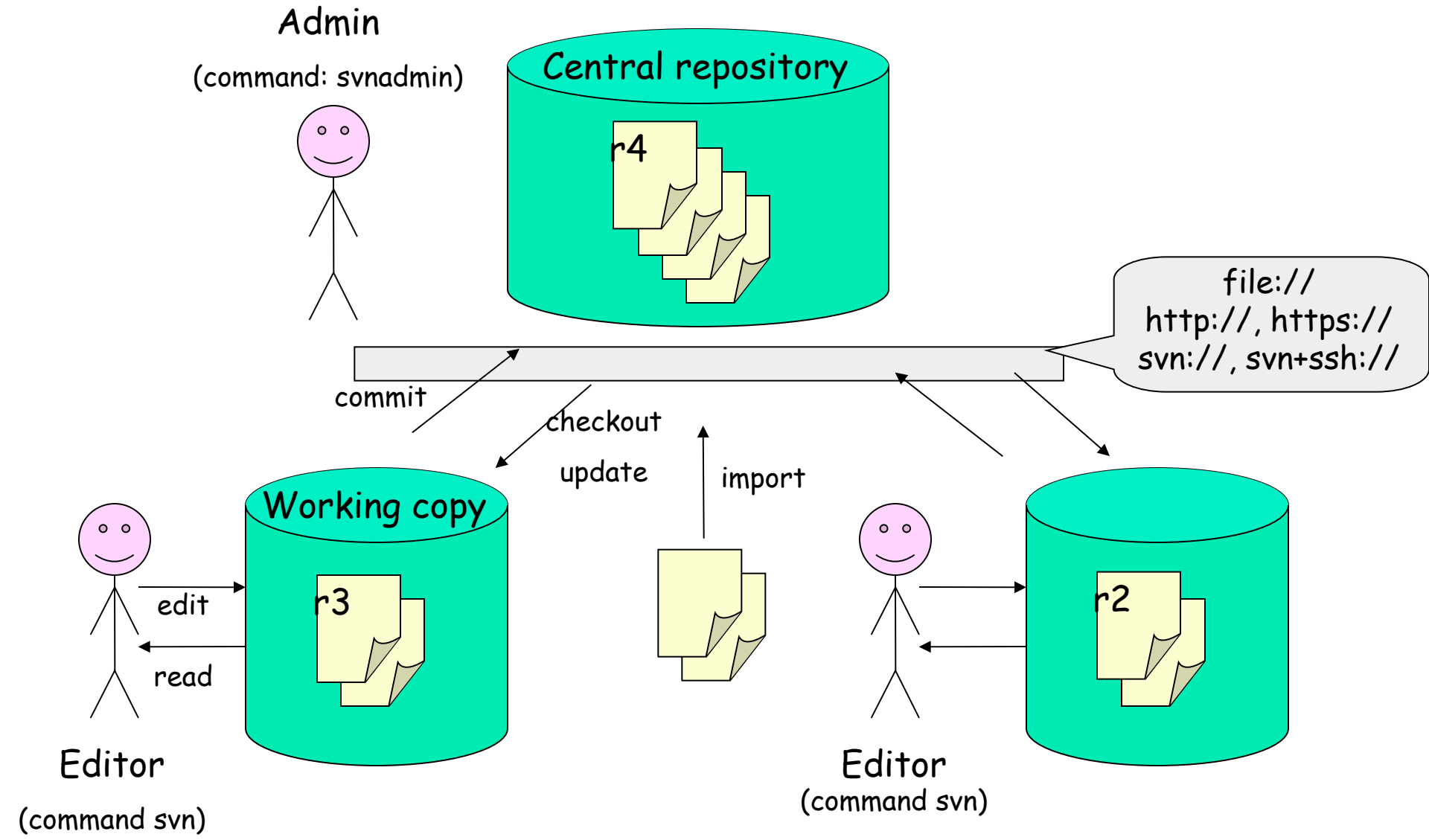
- Doing nothing!
- Lock-Modify-Unlock (Pessimistic)
  - SCCS, RCS
  - Decrease productivity
- Copy-Modify-Merge (Optimistic)
  - Conflicts resolution when concurrent modifications (which are actually rare)
    - Merge, Selection, …
  - CVS, SVN : Client level resolution
- Policy-based
  - Merging and validation process for each code contribution

Concept of Version

- Trunk
  - main development
- Branches
  - Alternatives to trunk
    - Different design/implementation (experimental), vendor-specific
- Revisions
  - Sequence of versions
- Tags
  - Symbolic references to revisions (Tiger, LongHorn, …)
    - Represent a public release (R), a milestone (M)
- Branch merging

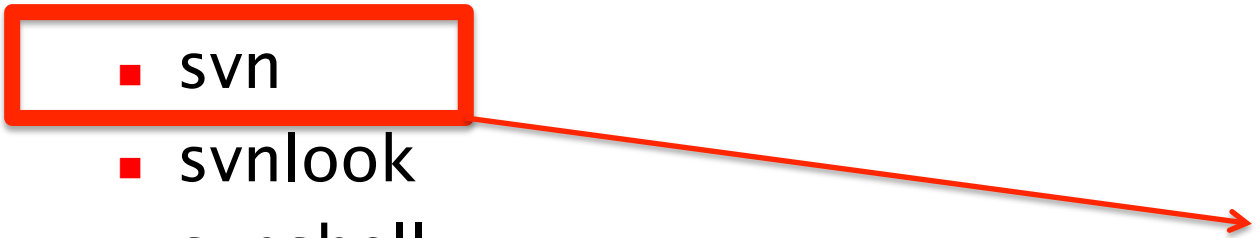
SVN

SubVersion: Architecture



Main commands

- Editor
  - svn
  - svnlook
  - svnshell
- System Administrator
  - svnadmin
  - svndumpfilter
  - svnlook
  - svnshell
- Server
  - svnserve, mod_dav_svn for Apache HTTPD



« User » focus

Git


GIT

≠


SUBVERSION

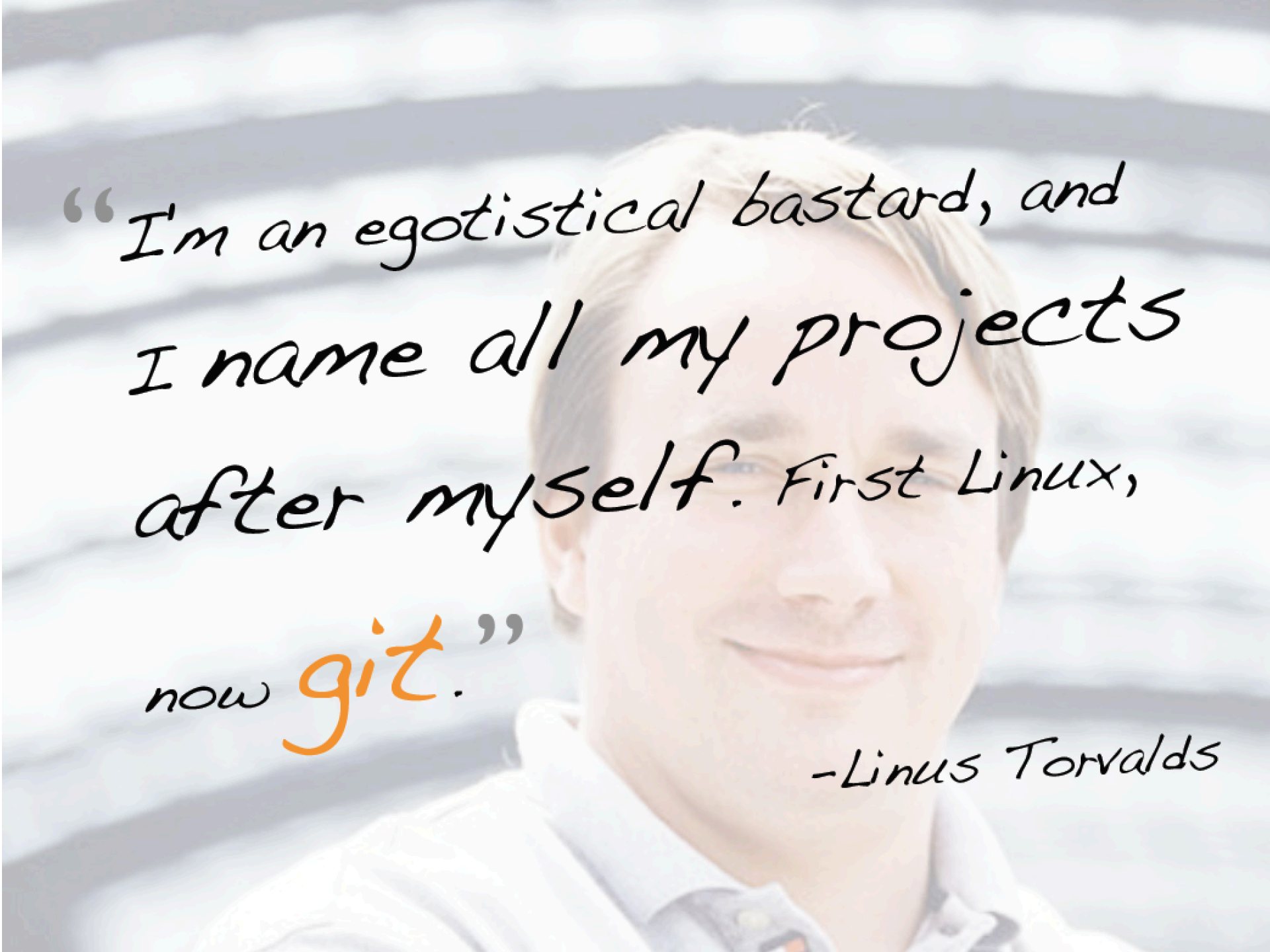

github
SOCIAL CODING
Signup and Pricing
Explore GitHub
Features
Blog
Login
1,067,856 people hosting over 3,012,331 git repositories
jQuery, reddit, Sparkle, curl, Ruby on Rails, node.js, ClickToFlash, Erlang/OTP, CakePHP, Redis, and many more
Find any repository
twitter
facebook
rackspace HOSTING
digg
YAHOO!
shopify
EMI
six apart
git /'gɪt/
Git is an extremely fast, efficient, distributed version control system ideal for the collaborative development of software.
git·hub /'gɪt,hʌb/
GitHub is the best way to collaborate with others. Fork, send pull requests and manage all your public and private git repositories.

« Git »

-noun

an unpleasant
or contemptible **person**

-Oxford English Dictionary


"I'm an egotistical bastard, and I name all my projects after myself. First Linux, now git."
-Linus Torvalds

Git

- version control system
  - designed to handle very large projects with speed and efficiency
  - mainly for various open source projects, most notably the Linux kernel.
- http://git.or.cz/

Git Advantages

- Resilience
  - No one repository has more data than any other
- Speed
  - Very fast operations compared to other VCS (I'm looking at you CVS and Subversion)
- Space
  - Compression can be done across repository not just per file
  - Minimizes local size as well as push/pull data transfers
- Simplicity
  - Object model is very simple
- Large userbase with robust tools

Some Commands

- Getting a Repository
  - git init
  - git clone
- Commits
  - git add
  - git commit
- Get changes with
  - git fetch (fetches and merges)
  - git pull
- Propagate changes with
  - git push

Git and SVN in depth

Mainly based on the content of

http://people.irisa.fr/Anthony.Baire/git/

(one of the best introduction of Git/SVN)

Relationship with PDL (your project)

Impacts

- My proposal: use existing system like github
  - Collaborative work
  - You will keep your work somewhere
  - You will be able to demonstrate your skills

- Not in this course
  - Code reviews
  - Continuous integration