

Version Control at UCB

*Version control with Subversion
and Subclipse*

Summary

- Version control in general
- Subversion in particular
- Version control at UCB
 - In IST-WA

What is version control?

- Keep track of revisions
- Investigate changes and revert
- Several people working on project (file) simultaneously
- Merge changes

Why do I need version control?

- A place to store your code
- Historical record of what was done over time
- Synchronization between developers

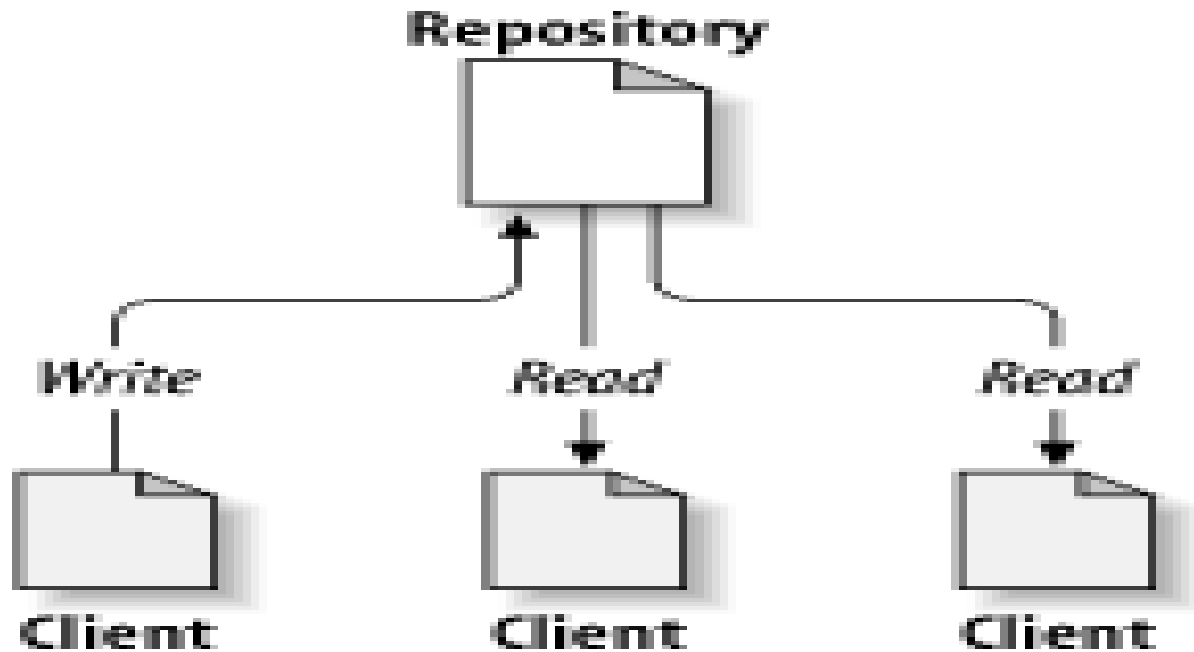
Why (cont.)?

- Standardized application layout
- Facilitates automated build, test, deploy
 - No machine-specific dependencies
- Developer not tied to one machine
 - Work from home
 - Work from any machine

Why (cont.)?

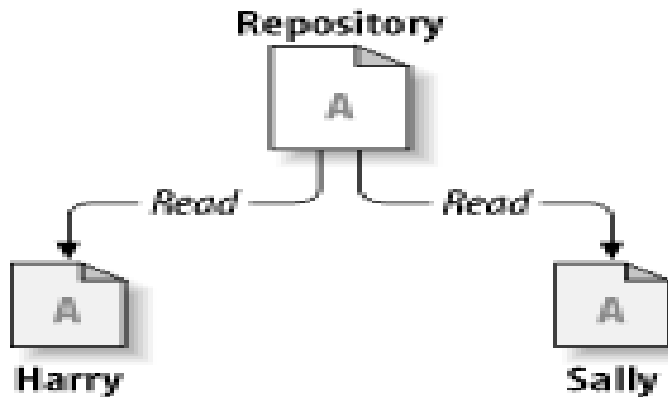
- Best practice
- Director priority
- We've had problems

Typical client/server system

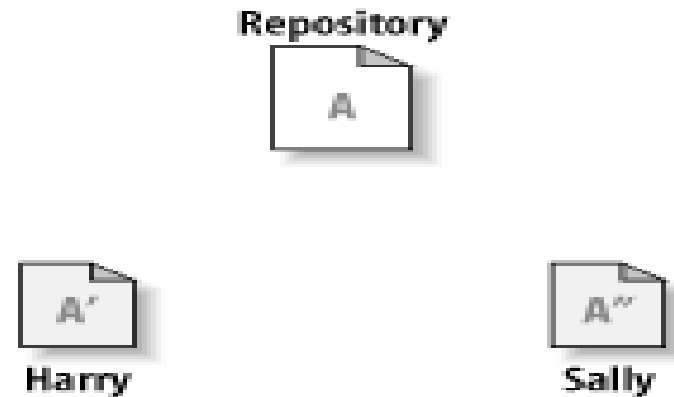


The problem to avoid

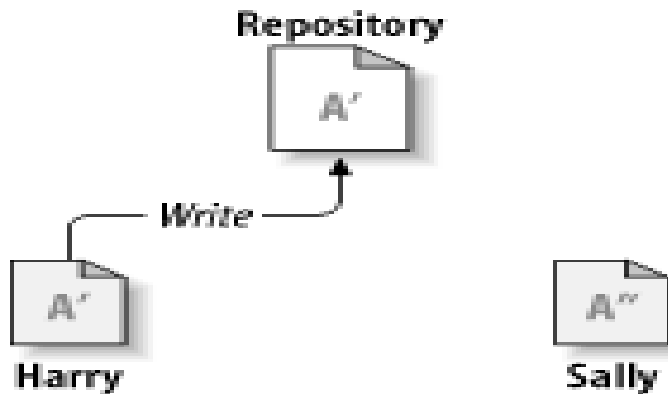
Two users read the same file



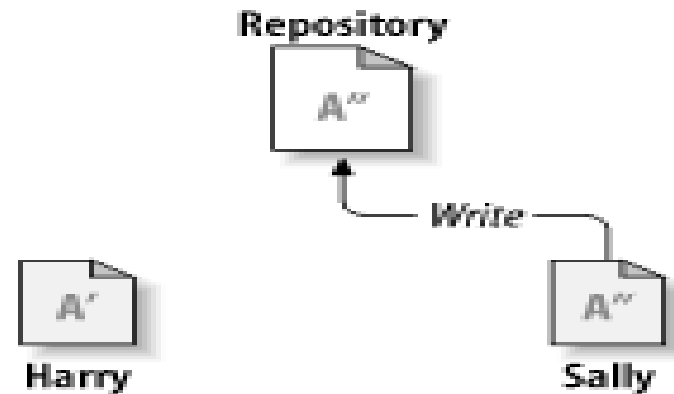
They both begin to edit their copies



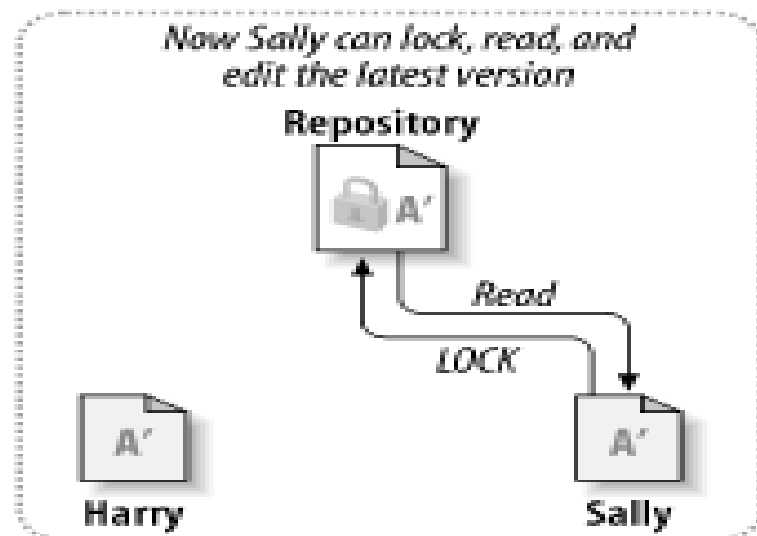
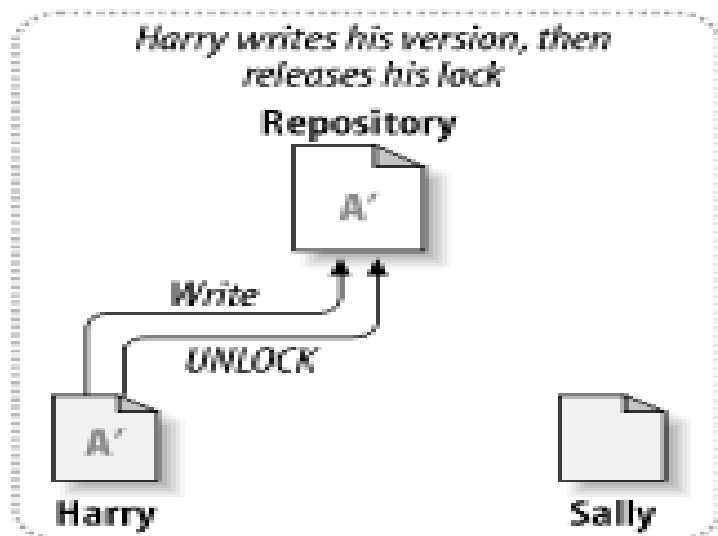
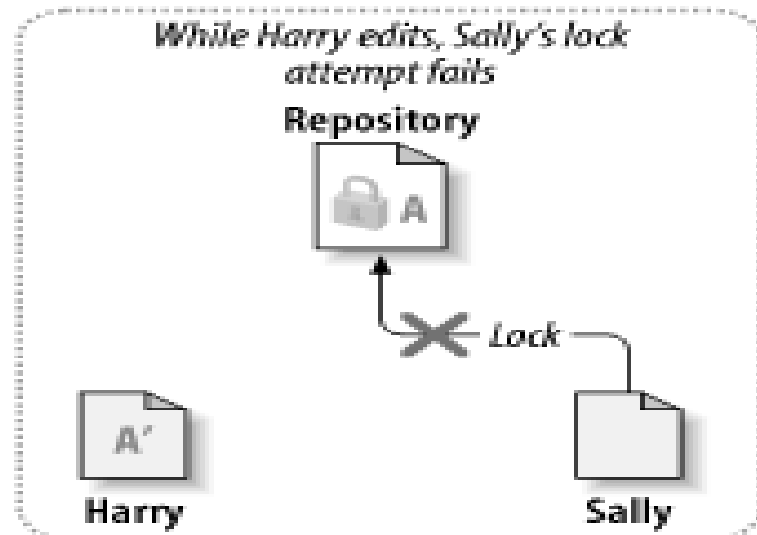
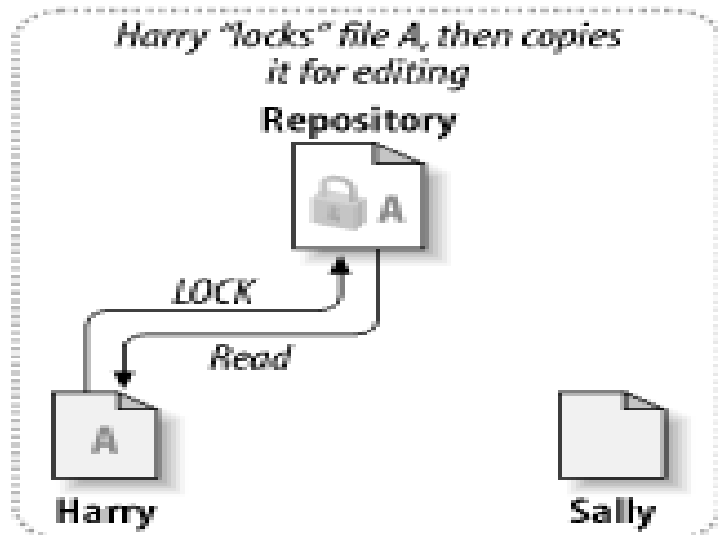
Harry publishes his version first



Sally accidentally overwrites Harry's version

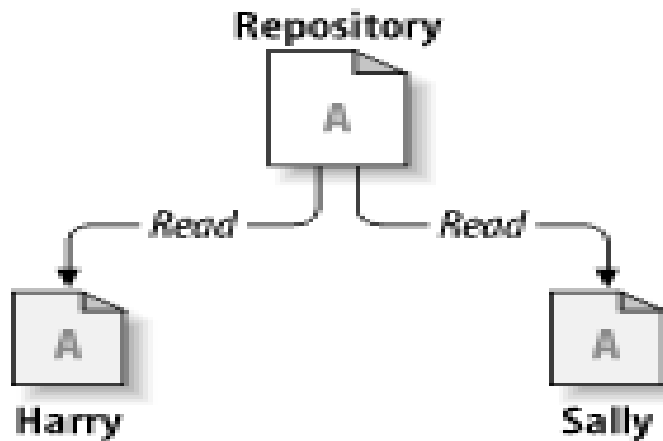


Lock-modify-unlock

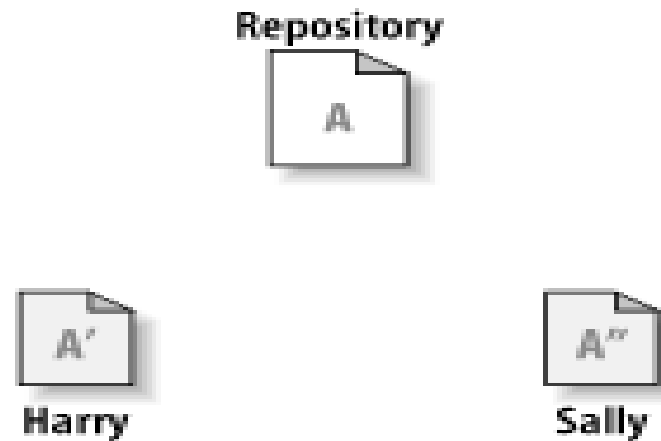


Copy-modify-merge

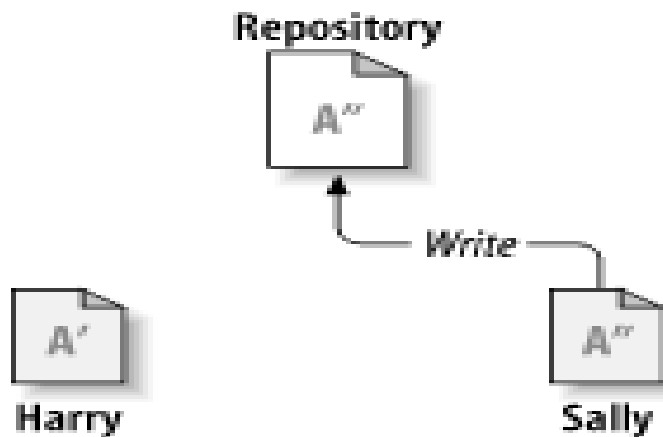
Two users copy the same file



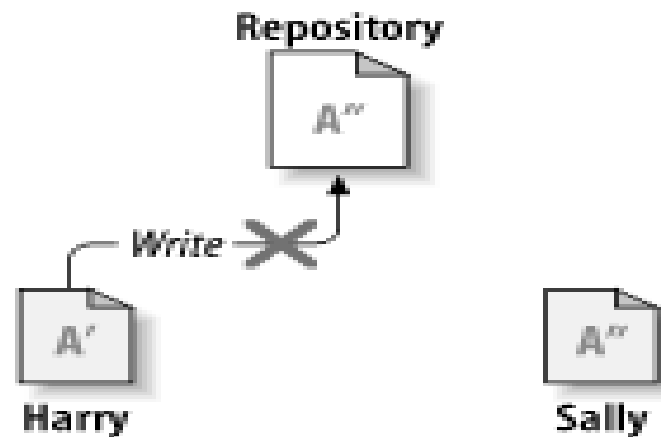
They both begin to edit their copies



Sally publishes her version first

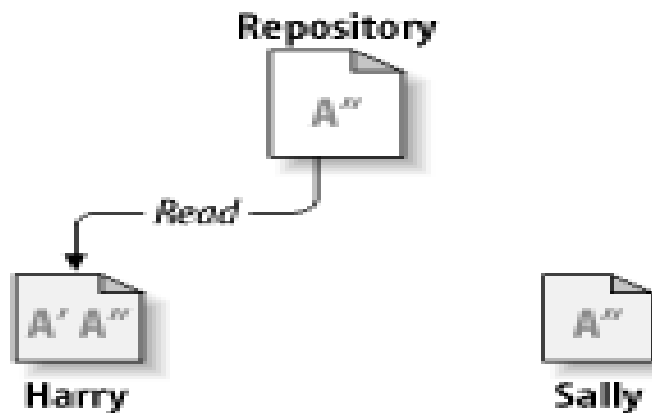


Harry gets an "out-of-date" error

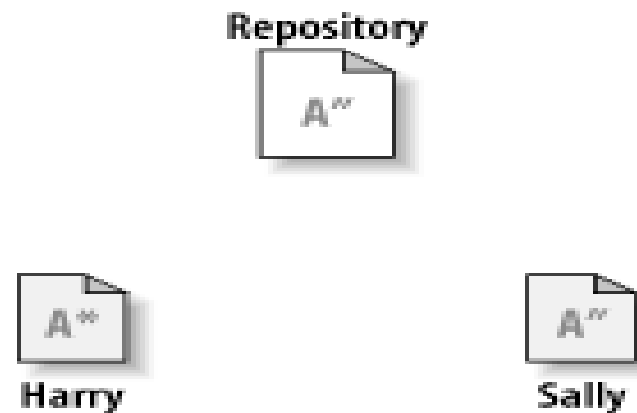


Copy-modify-merge (cont.)

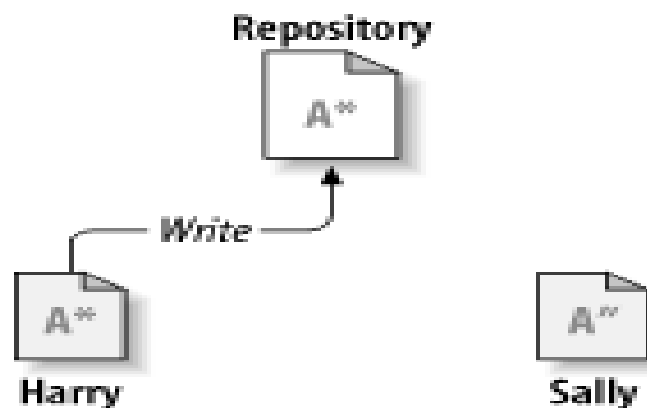
Harry compares the latest version to his own



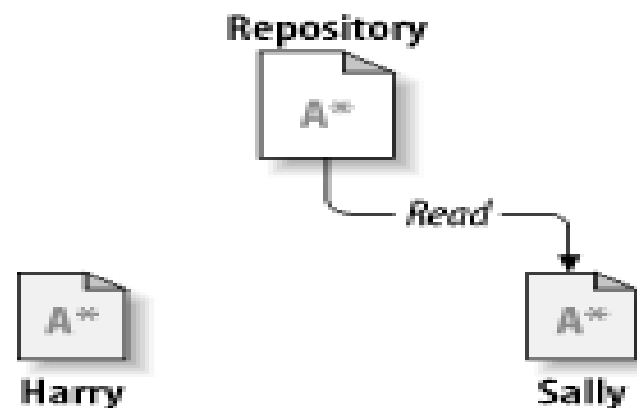
A new merged version is created



The merged version is published



Now both users have each others' changes



Standard Cycle

- Checkout/update
- Edit
- Update (merge)
- Commit

Standard Cycle - Ideal

- Checkout/update and run tests
- Edit and run tests
- Update (merge) and run tests
- Commit
- Deploy to dev integration environment
 - And run tests

Best Practices

- Commit early and often
- Commit logical units
- Don't break the build (unit tests must run)
- Communicate with team members
 - JIRA

What to store?

- Source code
- HTML, CSS, Javascript
- Images
- Configuration files
- Unit and other automated tests
- Everything required for your app to run

What to store (cont.)?

- Requirements documents
- Documentation – user
- Most documents/files related to your project

What not to store

- Compiled code
- Generated code
- Generated documentation
- Secured information (TBD)
 - Id/password files
 - License keys

Commercial Tools

- Bitkeeper
- Visual SourceSafe
- Rational Clearcase
- Perforce

Open Source Tools

- CVS
- Subversion
- Many others

Why SVN over CVS?

- Versioning for Files, Directories and Metadata
- Atomic commits
- Excellent network support (speed)
- Cheap Branching, Tagging, and Merging
- True Cross-Platform Support (runs very well on Windows)

SVN Over CVS (cont.)

“The goal of the Subversion project is to build a version control system that is a compelling replacement for CVS in the open source community.”

- Same developers as CVS
- Solve CVS's problems
- Majority of open source projects
 - Are using SVN
 - Are planning to migrate to SVN

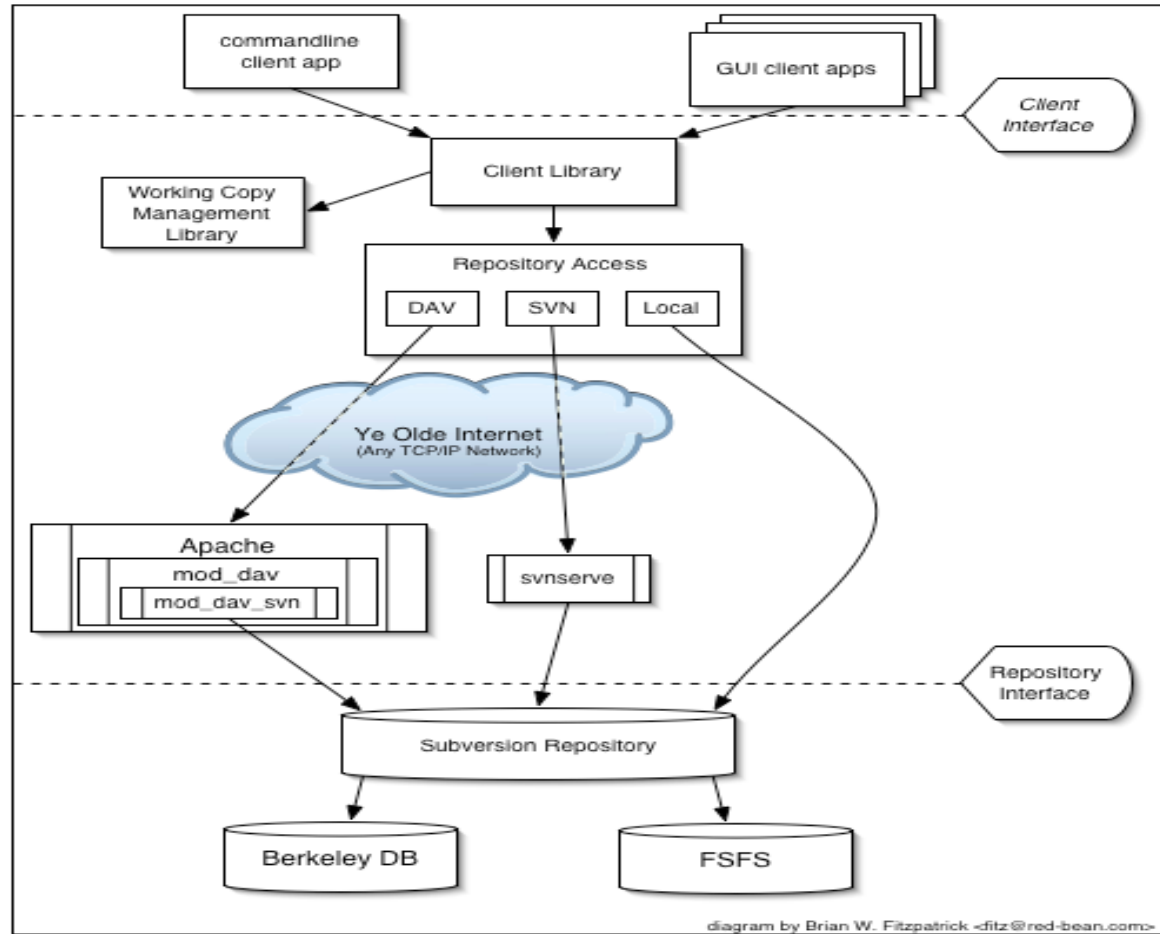
Build Tool Integration

- Java
 - Ant task SvnAnt
- Ruby on Rails
 - Capistrano
- Other environments can use the command line API

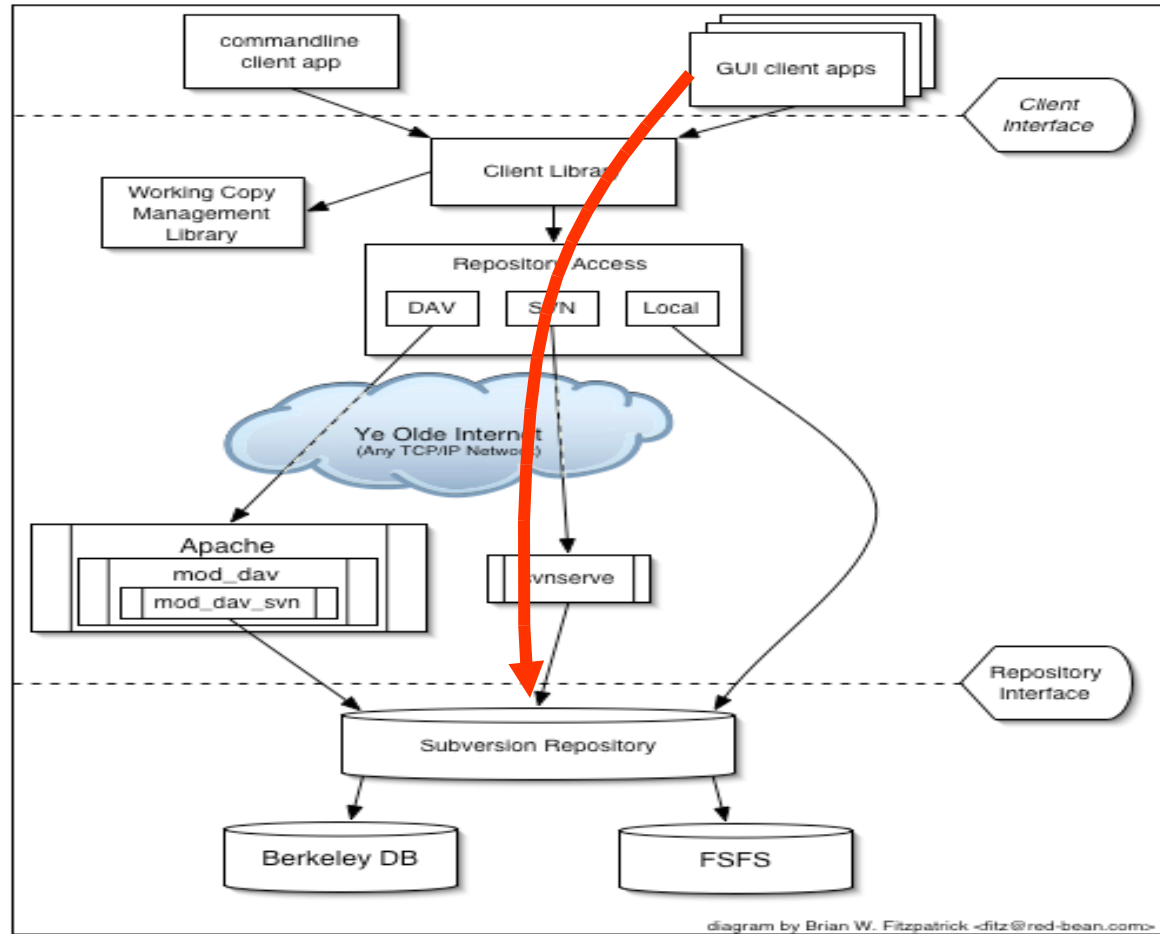
Source Control at UCB

- CVS
 - Heavy use in former SIS
 - Other in IST
 - Other on Campus
 - Conversion schedule TBD
- SVN
 - Early days in Application Services
 - Sakai project
- Alchemist

Subversion's Architecture



Subversion's Architecture



Subversion Clients

- Subclipse
- TortoiseSVN
- Command line

Command Line

```
$ svn import /tmp/myproject  
  file:///path/to/repos/myproject -m "initial  
  import"
```

```
Adding /tmp/myproject/branches
```

```
Adding /tmp/myproject/tags
```

```
Adding /tmp/myproject/trunk
```

```
Adding /tmp/myproject/trunk/foo.c
```

```
Adding /tmp/myproject/trunk/bar.c
```

```
Adding /tmp/myproject/trunk/Makefile
```

```
...
```

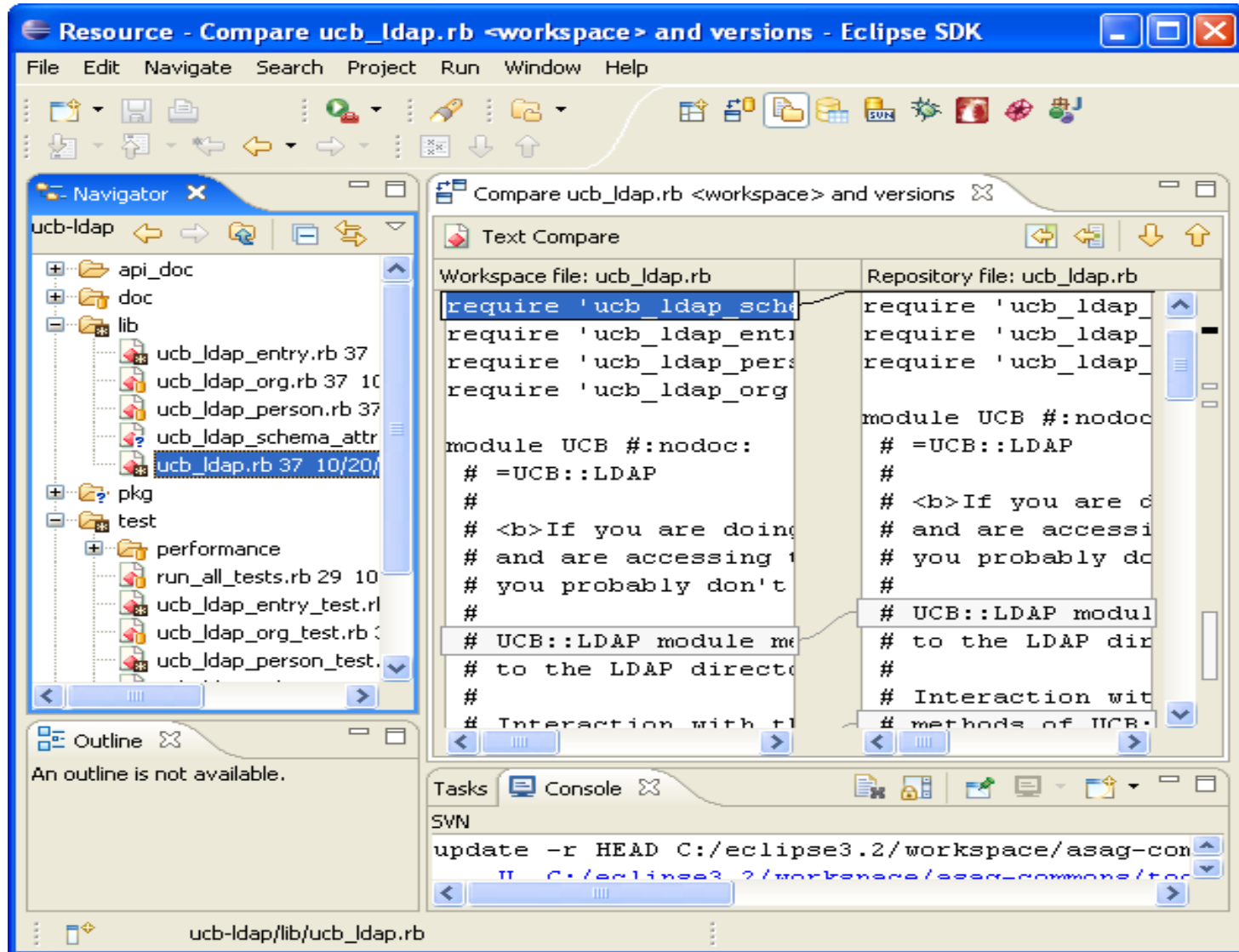
```
Committed revision 1.
```

```
$
```

TortoiseSVN

- Context menus in Windows Explorer
 - [http://
tortoisesvn.tigris.org/ExplorerIntegration.html](http://tortoisesvn.tigris.org/ExplorerIntegration.html)
- Texas A&M tutorial
 - <http://tagd.cs.tamu.edu/resources/WeeklyStuff>

Subclipse



Repository Organization

- Per-project directories
- Three sub-directories
 - Trunk
 - Tags
 - Branches
- Fine-grained security

Security

- Write to your projects
- Read everything (some exceptions)
 - Code sharing/snooping
 - No more broken windows
 - How did she do that?
- No “security by obscurity”

Campus Proposal

- Free SVN repository to anyone on campus
- Part of IST/AS/AG campus outreach
- Easy win, concrete accomplishment
- Facilitate sharing of code

Getting Started

- <http://ist.berkeley.edu/as/ag/>
 - Tools > Dev Box Setup
 - Install Eclipse and plugins
 - Install Subclipse
- Get userid
- Play in sandbox

Resources - Links

- This document
 - <http://ist.berkeley.edu/as/ag/pub/pdf/VersionControlAtUCB.pdf>
 - <http://ist.berkeley.edu/as/ag/pub/ppt/VersionControlAtUCB.ppt>
- Application Services – Architecture Group Web Site
 - <http://ist.berkeley.edu/as/ag/>
- Subversion book
 - <http://svnbook.red-bean.com/>
- Online version of Subclipse documentation
 - <http://svn.collab.net/subclipse/help/index.jsp>
- TortoiseSVN
 - <http://tortoisesvn.tigris.org/>

Resources - Books

- Version Control with Subversion
 - <http://safari.oreilly.com/>
 - <http://svnbook.red-bean.com/>
- Pragmatic Version Control Using Subversion
 - <http://www.pragmaticprogrammer.com/titles/svn2/index.html>