CVS - Concurrent Versions System

Version Control System -- usually source

History:
- RCS: revision control system, early 1980s.
  - Managed single directories
  - Did not work over networks
- CVS: scripts in 1986, C in 1989
  - Managed directory trees
  - Worked over networks

Other Source Control Tools (open source)
- subversion (svn) -- a "rewrite" of CVS (2000)
- git -- Initially written by Linus Torvalds for Linux kernel (2005)
- mercurial (hg) -- python & C (2005)
- fossil -- written in C, has a web server, wiki and bug tracker integrated (2006)
- many others, both open and closed source
Features of CVS

Store all revisions, retrieve any revision you want
Log of changes, who did it, when it was done
Can ask for differences between revisions
Can tag revisions for later retrieval by tag
May have multiple revisions of the same file "active"
CVS -- Basic Ideas

Repository
- collection of files and directories where everything is stored
- Initialize the repository
  - `cvs -d /path/to/rep init`
- NO SOURCE DEVELOPMENT DONE IN THE REPOSITORY

Accessing Repository
- Local:
  - `-d /full/path/to/dir` (flag)
  - `CVSROOT=/full/path/to/dir` (Environment variable)
- Remote:
  - `-d user@machine:/full/path/to/dir` (flag)
  - `-e "ssh -p 922"
  - `CVSROOT=machine:/full/path/to/dir`
  - `CVSROOT=user@machine:/full/path/to/dir`
  - ssh: ~/.ssh/config
    - Host `linux-* cs.wwu.edu`
    - Port 922
Operations:
- check-out / get
  - copies FROM a repository to a working directory
- commit
  - copies change from a working directory TO the repository
- add or delete
  - add or delete from the WORKING directory
  - needs a commit to make the change to the repository
- update
  - add changes in from repository (for working together)
- many more ...
Revision "Tree"

☐ HEAD: the primary point of development
  ☐ "normal check-out" gets HEAD

☐ BRANCH: an alternate point of development
  ☐ requires a special form of check-out
  ☐ Often used for "release branches"
  ☐ May also be used for different machines (not so good)
Operations - details

- Initialize the repository
  
cvs -d /full/path init

- Import (add) existing sources into the repository
  
cvs import -m msg repository vendor-tag release-tag
  NOTE: must be IN directory importing

- Get a copy of all files to make a "working directory"
  
cvs get name (checkout, co)
  NOTE: not done in directory you want to get

- Make sure a working copy is up to date with the repository
  
cvs update
  NOTE: must be in directory you want to update

- Commit the changes in the working directory to the repository
  
cvs commit (ci, checkin)
  NOTE: in the directory from which you want to commit
Operations (Page 2)

- Add a file as a change to the working directory
  ```
cvs new   (add)
  NOTE: you must create file/directory first.
  not added until commit done.
  ```

- Delete a file or directory from the next revision in the repo
  ```
cvs delete
  NOTE: must delete file/directory first.
  can still check out old revisions!
  ```

- Compare two different revisions
  ```
cvs diff
  example: cvs diff -r1.1 -r1.2
  ```

- Show information about each line of the file
  ```
cvs annotate
```
Tag the currently checked in version (not the changes in working dir)
cvs tag
  Use this to tag your assignments
  May retag with same tag (-F option)
  May remove a tag (-d option)
  Done in directory in which you want to tag
    Example: cvs tag -F ASSIGNMENT-3

Get the status and tags on the current files in the working dir
cvs status
  Example: cvs status -v | more