Software Development

• Hack, hack, hack, hack, hack
  – Sorta works
Software Development

• Hack, hack, hack, hack, hack
  – Sorta works
• We keep a copy, in case we get stuck later on

Software Development

• Hack, hack, hack
• It works pretty well, so we keep another copy.
Software Development

- Hack, hack, hack
- Now it works (we think), we decide to release it.

Main.c
Main_old.c
Main_not_as_old.c
Main_rel_1.c
Software Development

- We keep working to improve our software
- Hack, hack, hack, hack, hack
- New and improved version works (we think), we decide to release it.
Software Development

• Oh, no!!! We have a bug in release one.
  – We need to fix it (and not force the to upgrade to rel_2).
• Hack, hack, hack, hack, hack
• Now have a fixed version.
Software Development

• Oh, no!!! Another bug in rel_1.
• Hack, hack, hack, hack, hack
• Now have a fixed version.
Software Development

• Oh, no!!! A bug in rel_2.
• Hack, hack, hack, hack, hack
• Now have a fixed version.
Software Development

• Now we go back to work on a new release.
• Hack, hack, hack, hack, hack
• Stable version

Main_after_rel_2_but_not_ready_yet.c
Main_rel_1.c
Main_rel_1_fixed.c
Main_rel_1_fixed_2.c
Main_rel_2_fixed.c
Main_rel_2.c
Main_old.c
Main_rel_1_fixed.c
Main.c
• Suppose we have to deal with a multi-file project
We need help!!!

• Welcome to CVS
  – Concurrent Versions System
  – Keeps track of the different versions of your files
  – Keeps track of the relationship between different version files
  – Allows more than one person to work on the files at the same time
CVS Development Model

Repository (cvsroot)

- Contains the various versions of your files
- You don’t access it directly, only indirectly via cvs commands
CVS Development Model

Repository (cvsroot)

Main.c

cvs checkout main.c

Extracts a working copy of main.c for us to work on
CVS Development Model

Repository (cvsroot)

Main.c

Hack, hack, hack
We are at a point where we wish to save a version.
CVS Development Model

Repository (cvsroot)

Main.c

Hack, hack, hack
CVS Development Model

We are at a point where we wish to save another version
CVS Development Model

• We are keeping a copy of each version of main.c

• The first version forms the root of a tree (only the trunk shown here now)

• Each new main.c grows the tree trunk higher
How can we specify a particular version of a file?

- **Use dates and times**
  - Awkward to use (hard to remember when something happens)

- **Use CVS internal numbering**
  - They end up being meaningless quickly
  - Multi-file projects end up with many version numbers that don’t relate to each other

- **We need something more useful**
CVS tags

cvs tag symbolic_name

– Allows us to give symbolic names to particular versions of files
  • E.g. cvs tag ass1-start
Multiple File and Tags
Tagging A Coherent Version

REL_V1

F1.2

A1.2

B1.3

C1.1

F1.1

A1.1

B1.2

C1.0

F1.0

A1.0

B1.1

B1.0
Tagging

• You can do the following on tags
  – Add
  – Delete
  – Move
    • Change the version a tag refers to
  – Rename
• Can tag the repository directly
  – cvs rtag
• See www.cvshome.org for details
CVS Development Model

cvs checkout main.c

Hacker 1

Repository (cvsroot)

Main.c

Concurrent Development

Hacker 2

cvs checkout main.c
How do we handle when Hacker 1 commits a copy and hacker 2 becomes out of date?
CVS status & update

- CVS status provide the “status” of your files

File: errmsg.h Status: Up-to-date

| Working revision: | 1.1.1.1 Fri Mar 14 03:47:33 2003 |
| Repository revision: | 1.1.1.1 /home/kevine/cs3231/cvsroot/src/kern/include/kern/errmsg.h,v |
| Sticky Tag: | ass1-pre3 (revision: 1.1.1.1) |
| Sticky Date: | (none) |
| Sticky Options: | (none) |

- CVS –q –n update
  - Perform an “update”
    - -q “quietly”
    - -n “no action”
CVS update

- Brings the file (directory, or directory tree) up-to-date with a specified version
  - When no version is specified, it brings it up-to-date with the latest release

- `cvs update`
  - Update to latest release

- `cvs update -r os161-base main.c`
  - Update to version that was tagged os161-base
cvs update output

- **U file**
  - The file was brought up to date with respect to the repository.
- **P file**
  - Like `U`, but the CVS server sends a patch instead of an entire file.
- **A file**
  - The file has been added to your private copy of the sources
- **R file**
  - The file has been removed from your private copy of the sources
- **M file**
  - The file is modified in your working directory.
- **C file**
  - A conflict was detected while trying to merge your changes to file with changes from the source repository.
- **? file**
  - file is in your working directory, but does not correspond to anything in the source repository, and is not in the list of files for CVS to ignore
Example: cvs –q –n update

% cvs –q –n update
A kern/asst1/bar.c
A kern/asst1/bar.h
A kern/asst1/bar_driver.c
R kern/asst1/catlock.c
R kern/asst1/catsem.c
R kern/asst1/stoplight.c
A kern/asst1/test.h
M kern/conf/conf.kern
M kern/include/synch.h
M kern/include/test.h
M kern/include/version.h
M kern/main/menu.c
M kern/thread/synch.c
M kern/thread/thread.c
M lib/hostcompat/time.c
M lib/libc/exit.c
%
Example: Reverting to a different version of a file

% rm main.c
% cvs update –r tag_you_want main.c
%
How do we handle the “go back and bugfix an old release” problem?

• We would like to go to the version released and make changes
• We can’t insert in the middle of the trunk, and the head of the trunk is being used for REL_V3
How do we handle the “go back and bugfix an old release” problem?

- We can use a branch
  
  \% cvs rtag -r REL_V1 -b REL_V1_fixes main.c

- Note branch tags
  - are different to normal tags
  - always refer to the head of the branch
Checking out branches

• `cvs checkout main.c`
  – Checks out

• `cvs checkout –r REL_V1_fixes`
  – Checks out

• To continue development, you must check out a branch tag or the main trunk

  REL_V1_fixes

  REL_V1

  REL_V2

  REL_V3
Adding and removing files

- cvs add file.c
- cvs remove file.c

  - Note: Like always, you must *commit* to make the changes visible
View tags and commit logs

- cvs log
$ cvs log synch.c

RCS file: /home/kevine/cs3231/cvsroot/src/kern/thread/synch.c,v
Working file: synch.c
head: 1.1
branch: 1.1.1
locks: strict
access list:
symbolic names:
    ass1-v3-start: 1.1.1.12.1
    ass1-v3-test: 1.1.1.10.12
    ass1-v3-rel2: 1.1.1.2.2
    ass1-v3-rel1: 1.1.1.2.2
    ass1-v3-pre2: 1.1.1.2.2
    ass1-v3-pre1: 1.1.1.2.2.0.4
    ass1-v2-pre1: 1.1.1.10.10
    ass1_v1-start: 1.1.1.8.1
    ass1_v1: 1.1.1.0.8
    asst1: 1.1.1.1
    ass1: 1.1.1.1
    ass1-test-base: 1.1.1.6.1.0.2
    ass1-test-pre: 1.1.1.6.1
    ass1-test1: 1.1.1.0.6
    ass1-rel3: 1.1.1.2.2
    ass1-rel2: 1.1.1.2.2.0.2
keyword substitution: kv
total revisions: 8; selected revisions: 8
description:

-----------------------------
revision 1.1
date: 2003/03/14 03:47:33; author: kevine; state: Exp;
branches: 1.1.1;
Initial revision

-----------------------------
revision 1.1.1.1
date: 2003/03/14 03:47:33; author: kevine; state: Exp; lines: +0 -0
branches: 1.1.1.1.2; 1.1.1.1.4; 1.1.1.1.6; 1.1.1.1.8; 1.1.1.1.12;
Initial import of os161

-----------------------------
revision 1.1.1.1.12.1
date: 2003/03/27 01:46:22; author: kevine; state: Exp; lines: +87 -27
test start

-----------------------------
revision 1.1.1.1.8.1
date: 2003/03/19 08:34:15; author: kevine; state: Exp; lines: +87 -27
Start of assignment 1

-----------------------------
revision 1.1.1.1.6.1
date: 2003/03/17 23:30:03; author: kevine; state: Exp; lines: +87 -27
patched to bring up to date