Using CVS to Manage Student Assignments

Karen Reid and Greg Wilson
{reid, gvwilson}@cs.utoronto.ca

Context

- Professional software developers use version control to:
  - Coordinate with their teammates
  - Keep a record of who's done what
- In contrast, students use email, ftp, a floppy, or a USB key
  - More effort
  - More likely to go wrong
- Hm…

The Experiment (So Far)

- Since May 2003, we've used CVS to:
  - Distribute assignment starting points
  - Collect solutions
  - Get grades back to students
  - Track progress
  - Resolve disputes
- It works—well
- Here's what we do, and what we've learned

What Is Version Control?

- Instead of mailing files around, keep a master copy on a shared server
  - Everyone sends changes to this *repository*
  - Then fetches other developers' changes
- Server saves old versions, along with who made particular changes, when, and why
  - Allows "infinite undo" for the project as a whole
  - Reduces or eliminates the risk of one person overwriting another's work

Long-term Goals

- Help students see that software development skills can be learned and improved
  - Programming is not a “black art”
- Persuade them that tools are worth learning
  - Get them past Glass’s Law
- Prepare them for working in teams
  - Not trying to learn CVS, *and* learn how to program with a partner, at the same time
  - Able to do team exercises by end of CSC207

Short-term Goals

- Introduced CVS in CSC207: Software Design
  - Students know basic Java, but little else
- Immediate goals:
  - *Convince students to use real software tools*
  - Help them work on both personal machines and university lab machines
  - Let TAs get up-to-date copies of students' work during tutorials and office hours
  - Provide history of students' work to help us tune the course (and detect plagiarism)

Setup

- Instructor creates a repository for each student
  - User owner is student, group owner is **`csc207`**
  - Instructors and TAs belong to **`csc207`**
- Create one sub-directory in each repository for each exercise
  - Seed it with exercise spec, starting files, etc.
- All done using shell scripts
  - 4 sections, 3 campuses, 400 students…

Instruction

- Very important for students to have a clear conceptual model of what CVS does *before* they start using it
  - Lots can go wrong otherwise
- Spend at least one class + one tutorial on it
  - Plus lots of on-line help
  - Still get lots of questions in first two weeks…
  - …but it settles down quickly after that

What Students Do

- To start assignment, the student does **`cvs update`** in a checked-out copy of her repository to get the "seed" files
- Then code, test, document, etc., as usual, doing **`cvs commit`** periodically
  - When something works
  - When moving from machine to machine

What Instructors Do

- Instructors/TAs check out a copy of each students' repository at start of term
- Use **`cvs update`** to get current state of work when student comes into office with question
- Use **`cvs update -D`** ***`date`*** to check out repository as it was at due date
  - Can be done any time

Grading

- Initially had TAs add a file `marks.txt` directly to the students' repositories
  - Update it repeatedly as each part of exercise marked
- Caused problems:
  - Students panicked about "partial marks"
  - Students could actually edit their marks files…
- Now have TAs store marks in separate repo
  - Copy/commit file once grading done

Other Issues

- Initially gave each student a copy of all libraries needed for course (self-contained)
  - Required too much disk space
  - Now store those centrally, and require students to download them for their personal machines
- Some students damaged their repositories
  - E.g., edited version control files directly
  - Better up-front instruction has almost completely eliminated this

Other Issues (cont.)

- Managing security for group projects is difficult
  - Ideally, create one Unix group per project…
  - …but default Linux kernel only records the first 31 groups a user belongs to…
  - …so instructors couldn't check out many teams' repositories
  - Work around this using a setuid script
  - Look forward to ACLs on Linux

Other Issues (cont.)

- In theory, students could edit version histories in order to fake early submission
  - It would be complicated
  - No evidence that anyone ever did this…
- Investigated the use of CVS *triggers* to log repository activity securely
  - Moving to Subversion may take care of this

Advantages

- Students learned a core professional tool early in their careers
  - Many started using CVS in other courses
- Students could double-check submissions by checking out into a temporary directory
- Fewer lost files (particularly in group projects)
- Instructors had an accurate record of who did what, when
  - Which forestalled a lot of whining…

Usage Patterns

- Is there any correlation between CVS usage patterns and course grade?
- Keir Mierle and Prof. Sam Roweis used data mining in Summer 2004 to try to find patterns
  - To our surprise, there aren't any
  - Start early, start late, lots of small check-ins, one big check-in…
- Maybe because we're looking at the learning curve?

Conclusions

- It works, and works well
  - Forces students to adopt good practice early
  - Makes team projects easier to assign (and do)
  - Streamlines interactions between students, TAs, and instructors
- Next steps:
  - Switch to Subversion
  - Look for patterns in upper-year usage
  - Integrate with unit testing: “build and smoke tests”