Using the Web to teach programming

I'll describe today a pilot version of CS 3 taught summer 2002:

- implementation
- results
- implications for instruction at Berkeley and elsewhere.

CS 3

- CS 3 ("Introduction to Symbolic Programming") is a first programming course using Scheme.
- It covers
  - functional (side-effect free) programming,
  - recursion, and
  - use of higher-order functions.
- It ends with a relatively complex project (~200 lines of code).
- It includes activities based on *case studies*, narratives of worked-out solutions.

Traditional CS 3 contact hours

- 2 hours lecture
- 1 hour discussion
- 2 hours lab

Summer 2002 CS 3

- Contact hours for summer CS 3 (8-week session):
  - 0 hours of lecture,
  - 0 hours discussion section, and
  - 14 hours of lab per week.
- Activities were all online, developed in the WISE system and presented in a custom course "portal".
- Three lab sections (enrollment = 22, 18, 10) were each staffed by an instructor and two or three lab assistants.

WISE

- WISE (Web-based Inquiry Science Environment) combines a database of pedagogical activities with a front end that organizes them.
- Activities include
  - Web page reading
  - quizzes
  - online discussion
  - comparing online answers
  - online note-taking

CS 3 use of WISE

- Activities were organized in relatively small steps.
- The typical day started with a quiz, and proceeded through a bunch of the other kinds of activities.
- Some work was visible to the staff; mini-lectures or individualized tutoring occasionally resulted.
- Homework often involved contributing to discussion.

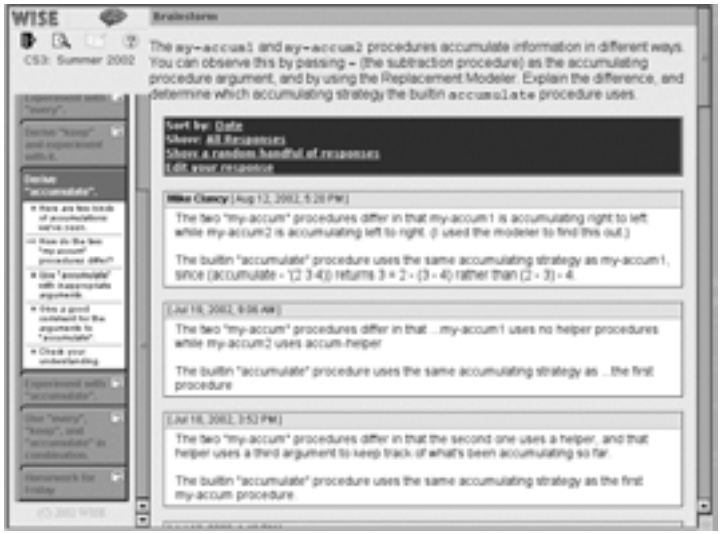
More CS 3 logistics

- Students did relatively more of their work in the lab rather than at home.
- Pacing was flexible; there were occasional “catch-up” days.

Course portal

Student learning environment

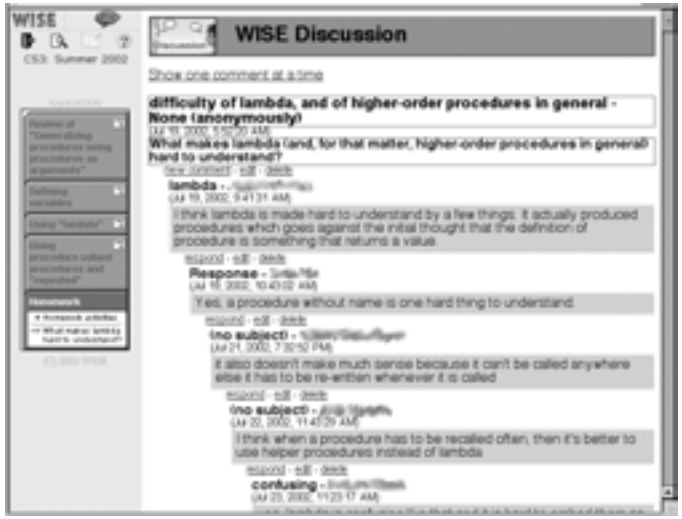
Collaborative problem solving



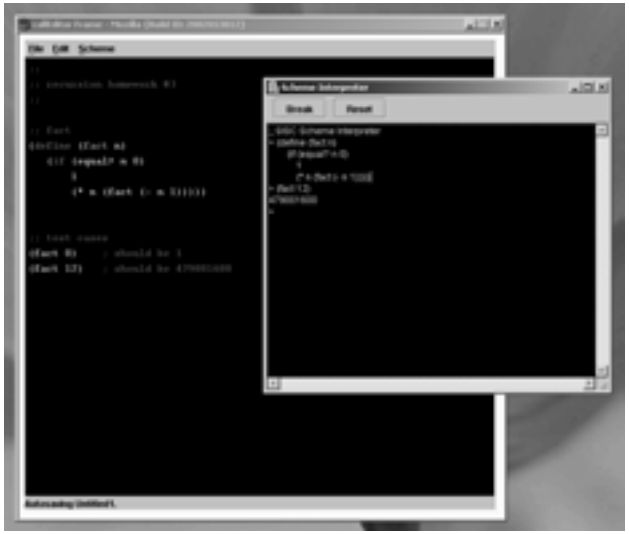
Students must post their own responses before they can view the responses of others.

Collaborative problem solving

Student discussion



Scheme tools



Summary of differences

- Higher proportion of supervised online activity.
- Constant monitoring of students; timely tutoring.
- Wider variety of activities, including implicit and explicit collaborations.
- Flexible pacing.

Results

- Better performance on comparable final exams (average = 32.9 out of 60 compared to 25.8).
- Extremely high course evaluation ratings.

Ratings of “respond then review” questions (5 is best, 1 worst)

- (3.7) Supply a solution to an exercise with several “right” answers.
- (3.5) Supply a suggestion for understanding ___ .
- (3.5) Explain how you got an answer.
- (3.4) Explain how ___ works.
- (3.4) Report the results of experiments.
- (3.3) Supply a good comment or parameter name.

Ratings of "respond then review" questions (5 is best, 1 worst)

- (3.1) Supply the "largest" or "smallest" solution to a problem.
- (2.9) Provide an opinion about choice between alternative designs or code segments.
- (2.8) Reflect on what might make ___ hard to understand.

Ratings of discussion questions (5 is best, 1 worst)

- (3.8) Comment on mistakes you made.
- (3.6) Suggest tips for understanding ___ .
- (3.6) Explain productive techniques for understanding larger programs.
- (3.4) Suggest how you might handle a given programming difficulty (e.g. reaching a dead end).
- (3.3) Devise a quiz for the case study.
- (3.3) Summarize the case study.

Ratings of discussion questions (5 is best, 1 worst)

- (3.1) Supply and justify a preference between alternative designs or code segments.
- (3.0) Describe features of a given design or program that you'd like to imitate.
- (2.8) Explain why ___ is hard for you or someone else to learn.
- (2.8) Provide and compare images of recursion.

What does the instructor do?

- Monitor online work
  - to help identify students who need help, and
  - to identify misconceptions.
- Tune the curriculum.
- Train staff.

Research questions

- Does it work in the regular school year? (Summer students may be exceptional somehow.)
- What good are the new activities?
- How much does timeliness matter?
- What role does physical location play? Would a “virtual” lab section work?
- Can CS 3S (self-paced version of CS 3) benefit?

More research questions

- Who succeeds that didn't in the old system?
- What value does a good lecturer provide, and how can we capture it?
- How do curricular segments compare?

Plans for Berkeley CS 3

- (Fall 2002) Experimental section of CS 3: 6 hours of lab, 1 hour of discussion/lab.
- (Spring 2003) New format for CS 3: 1 hour lecture, 5 hours lab, 1 hour discussion/lab.

Implementation at U.C. Merced

- The CITRIS project is supporting the transfer of our lower-division courses to U.C. Merced.
  - Campus is expected to open in fall 2004.
  - Students will take classes in Merced and at remote “learning centers” in Modesto, Fresno, and Bakersfield.
- Plans for spring 2003
  - A version of CS 3 will be run through Merced Community College (one instructor, no lab assistants).
  - Berkeley staff will watch Merced online activities and somehow alert instructor/student of difficulties.

Questions

- Can the system accommodate Merced students?
- Does virtual tutoring work?
- How much expertise is needed at the remote site?

Longer-term CITRIS-sponsored work

- Tools
  - The Curriculum Builder, which manages a data base of richly annotated course activities.
  - The Course Customizer, which allows a prospective instructor to build a course from activities in the data base.
- Development
  - Versions of the other lower-division CS courses, plus courses in other areas.

Research directions

- Methods for benchmarking educational innovation.
- Analysis of elements of technologically-based instruction.
- Studies of the process of course customization.