Teaching Technology for the Community

E.R. Jessup
Department of Computer Science
University of Colorado
Boulder, CO 80309-0430
Phone: 303 492 0211
Fax: 303 492 2844
jessup@cs.colorado.edu

KEYWORDS: teaching, community service, women and technology, education, diversity, technology education, task-centered design, heuristic evaluation, cognitive walkthrough, thinking aloud evaluation

Teaching Technology for the Community

ABSTRACT
Technology for the Community is an undergraduate computer science course taught at the University of Colorado under the auspices of the Institute for Women and Technology's Virtual Development Center. In the course, students use computers to do community service. More specifically, they work with local community service agencies, building computational solutions to problems confronting those agencies. Student projects are developed following the method of task-centered design and are evaluated by the students and the users according to three formal methods. In this chapter, we describe in detail the content of the course and review the design and evaluation methods we use.

INTRODUCTION

Technology for the Community is an undergraduate long-term projects course offered by the Department of Computer Science at the University of Colorado at Boulder [UCB]. In it, students develop computational products designed to serve the needs of local community service organizations.

BACKGROUND

During the summer of 2000, UCB's Department of Computer Science was selected to participate in the Institute for Women and Technology's [IWT] Virtual Development Center (VDC) (IWT, 2002). The VDC is a network of collaborative educational centers, termed Development Centers, that share the dual goals of stimulating the participation of women in technology and increasing the positive impact of technology on the lives of women. Through the Development Centers' educational activities, a diverse community of women is encouraged to participate in the development of technology. In particular, universities participating in the VDC all commit to teaching technical courses in a style thought to be open and appealing to women. Presently, the Development Centers are at Purdue, Santa Clara University, Smith College, Texas A&M, the University of Arizona, the University of California at Berkeley, UCB, the University of Texas at El Paso,

and the University of Washington. UCB's Center is unique in its emphasis on computer science; the others all focus on traditional engineering disciplines.

The primary activity of UCB's center is the Technology for the Community course offered through the Department of Computer Science. Project ideas for the course are generated in community brainstorming workshops, held annually and facilitated by IWT staff members. Participants in the workshops include technical and non-technical women from the Boulder community at large, representatives of local community service agencies, a few interested academics, and the students enrolled in the course. At the workshop, the attendees identify problems confronting the agencies that can be solved computationally. Over the course of one or more semesters, the students work in groups to bring their solutions to product form with technical and non-technical faculty members and interested community members all serving as advisors. A particular strength of the workshops is that they encourage participants to think beyond their workaday concerns: the resulting project ideas have been creative, interesting, and broad in scope.

The course has been offered each semester since Spring 2001. Its pilot offering will extend for four semesters, but we expect that it will become a permanent part of our curriculum.

We note that, for the purposes of this course, "community service organization" is quite loosely defined. Our clients have included a variety of non-profit and governmental organizations that provide aid or education of some kind to the people of Boulder, Colorado. In particular, the Boulder Valley School District has been an important participant.

CONTENT AND RESULTS

In the course, students learn to use software tools for such computational tasks as composing and editing websites, maintaining and manipulating databases, and creating CD-ROMs. The students study the process of turning an idea into a useful software product and methods for evaluating that product in terms of usability.

In the first three weeks of the semester, students put their new computational skills to work by creating "guest books." A guest book is a dynamic website where users can enter or retrieve information. That information is stored in a

database. Each guest book is developed independently by one student; it may have any purpose and design that the student chooses. This assignment gives all of the students a common experience, a common vocabulary, and a common set of computational tools. The guest book represents a substantial piece of individual work: for students who enter the course with little computational experience, designing of a website, the creating the database, and hooking the two parts together is a major accomplishment.

The remaining weeks then are devoted to completing a computational project in collaboration with a local community service agency. Representatives of the agencies meet with the students in order to refine the project ideas generated at the workshop, and they later work with the students to test and evaluate prototypes. To date, all of the projects have been Web-based although that characteristic is not a requirement of a course project.

While representatives of many agencies have played some role in the course, some of our most important projects to this point include

- A self-paced introductory tutorial about computers and the Internet for Boulder Senior Services,
- A variety of educational products for Sojourner Charter Middle School,
- A game for fourth graders simulating the ecology of a local marsh for the Boulder Valley School District,
- An on-line events scheduling calendar for use by local non-profit agencies developed for the Boulder Community Foundation.

While most of the projects satisfy small scale needs of the client agencies, a local bank executive who is very active in non-profit work expects the events calendar to save local non-profits $100,000 a year by reducing events scheduling conflicts.

The full set of projects is showcased at http://www.cs.colorado.edu/VDC.

ENROLLMENT

The course is open to all undergraduate students at UCB. About 48% percent of the students who have enrolled in the first three semesters are female. That is a remarkable percentage in the Department of Computer Science where females have totaled between 11 and 16% of undergraduate majors during the last 14

years. It demonstrates that UCB's Development Center is achieving its goal of woman-friendly technical education.

The student body is also diverse in terms of major. In the first three semesters, 40% of the 41 students have come from the College of Engineering and 60% from three other UCB Colleges (Arts and Sciences, Business, and Journalism and Communication.) About 28% of the students have been Computer Science majors. Freshmen and sophomores have enrolled, but most (85%) of the students have been juniors or seniors. The course is part of the Technology, Arts, and Media [TAM] certificate curriculum (TAM, 2002), and 35% of the students have come from the TAM program. Enrollment has grown each semester, but teaching facilities will likely limit class size to about 20-25 students.

The heterogeneous student body is suitable. Students work in small, interdisciplinary teams to complete the projects. Students with capabilities in such areas as design, art, writing, and social science find natural roles, as do students with strong computational skills. There is particular need for students with expertise in databases, website development, and the writing of educational materials.

There are no formal prerequisites for the course, but, for maximal comfort, participants should be able to write a simple program, compose a web page, or have some familiarity with at least one commercial software package. Students with more advanced computing knowledge are invited to help with computational instruction.

PHILOSOPHY

The course is a long-term projects course, meaning that projects may be larger than can be completed in a single semester. Students who do not deliver a finished product at the end of the semester are asked to write a continuation plan to guide student teams who take up the project in subsequent semesters. Students may also repeat the course any number of times for credit. Indeed, students are strongly encouraged to participate in the long term. We hope that future years will find a varied group of collaborating students enrolled, from beginners to experienced technical leaders.

Because the course is based on new and developing concepts in technology education, we encourage students to help to drive its direction and content.

Students in the course also need to be independent learners, willing to take responsibility for learning new material from sources other than textbooks. Students must also be open to both teaching and learning from a variety of people of very different backgrounds.

METHODOLOGY

In building their projects, students learn and use the process of *task-centered user interface design* (TCD, 1994). In this process, the users' statements of their own needs and capabilities drive the product design. While developed for user interface design, task-centered design applies to the creation of any product where usability is the key concern.

Task-centered design follows these steps:

- Figure out who'll use the system to do what
- Plagiarize!
- Rough out a design
- Think about it
- Prototype
- Test with Users
- Iterate
    - Build it
    - Track it
    - Change it

Students begin by interviewing prospective users of their product, learning about their backgrounds and about what they want the product to do. From the interviews, the students formulate a list of *tasks*. A task is a statement of something that needs to be accomplished that does not include information about implementation: a sample task for the events scheduling calendar might be "Mary, the events coordinator for XYZ organization, wants to change the starting

time for an event she posted on the calendar last week." The collection of tasks defines the functionality of the product.

Working from the list of tasks, the students research commercial products and websites, looking for existing materials that relate to their projects. From those products, students liberally borrow features and function (within the constraints of copyrights and patents.) The value of this "plagiarism" step is two-fold: borrowing reduces development time and borrowed features are often already familiar to users. The students then sketch out a detailed design on paper (a "low-fidelity" prototype) and proceed from the sketches to a rough computer implementation (a "high-fidelity" prototype.) See, for example, (Rudd, 1996) for more information about prototyping.

In practice, evaluation of paper sketches proves too difficult for the students, so students instead "think about" or evaluate the high-fidelity prototypes. Project groups first judge their work using *heuristic evaluation* (Nielsen, 2002a). Each group member examines the prototype individually, looking for usability problems. The students are guided by Jakob Nielsen's ten usability heuristics (Nielsen, 2002b) which are crafted to cover most known usability problems. The group members compare the results of their evaluations and determine how to correct the prototype in response to their findings. Each project group then carries out a *cognitive walkthrough* (Rieman, Franzke, & Redmiles, 1995) of the prototype. The students begin with a task statement and the list of steps needed to accomplish that task using the prototype. They walk through the list of steps, keeping in mind the interests and abilities of the users, trying to tell a believable story about what the user might think and do at each step. Again, the students correct their prototypes to fix awkward or non-intuitive features. The two different types of evaluation help the students to identify different types of problems with their prototypes.

Carrying out the evaluations is typically difficult for the students as they, as undergraduates, are seldom asked to evaluate their own work. As a result, not all of the students appreciate the importance of the evaluations, and many are not patient enough to do a thorough job. They often believe that they have made only the best design choices and that there is no room for improvement. Their opinions change markedly, however, with the final round of *thinking aloud* (TA) evaluations (Lewis & Rieman, 1982). In contrast to the other two evaluation types, TA evaluations are done with users. In TA evaluations, the students observe a user trying to accomplish a task with the prototype. The user is asked to speak aloud all of her or his thoughts about the prototype during the process. Almost without exception, most of the user's thoughts and decisions (and

stumblings and errors) catch the students completely by surprise. More than one student group has redesigned its prototype completely from scratch following the first round of TA evaluations, but responding to the users' actions and opinions typically gives the students a good sense of accomplishment.

The steps of high-fidelity prototyping, evaluating, and correcting the projects correspond to the last, iterative phase of task-centered design. Students repeat these steps until they and the users are satisfied with the results. Typically, the first round of TA evaluations has a large impact on the projects while the second and third rounds deliver refinements.

ASSESSMENT OF STUDENTS

Given the experimental nature of the course and the large time commitment of setting it up, assessment of the students' work to date has been quite informal. Grades are assigned as about 85% work on the project and 15% on preliminary assignments (mainly the guest book). The project grade is determined in part by the instructor (30% of total), in part by the client (10%), but mainly by the project group members themselves (45%). By means of an evaluation form that poses several questions, group members are asked to evaluate themselves and their teammates with respect to both effort and results. In reality, a well-functioning group with an ecstatically happy client can expect high grades regardless of the other factors. It has not proven necessary to distinguish students according to major, but freshmen and sophomores are graded on a somewhat easier scale than are upperclassmen. We expect to include more formal evaluation of learning gains and outcomes in future semesters.

WHAT THE STUDENTS SAY

Every semester, the students have reported that the strongest feature of Technology for the Community is the multidisciplinary project groups. Engineers rarely have the opportunity to interact with Fine Arts or Journalism majors in their usual coursework. The students come to recognize the diverse strengths of their project teammates and the variety of contributions those teammates are able to make to the project. The students also report that the biggest challenge of the course is learning to deliver a usable technical product to someone with little or no technical background. Not only must the features of the product match the abilities of the users but also the students must find ways to present those features clearly to the users. Even the liberal arts majors in the

course have grown up with some exposure to computers while not all of their clients and product users have. For most, successfully finding language that allows the students of different majors to communicate among themselves and with their community service partners represents the biggest accomplishment of the course.

References

Institute for Women and Technology [IWT] (2002), Retrieved July 20, 2002 from http://www.iwt.org/.

Lewis, C. (1982), Using the "*thinking aloud*" method in cognitive interface design (Research Report RC 9265 (40713)). Yorktown Heights: IBM Thomas J. Watson Research Center.

Lewis, C. & Rieman, J. (1994), Task-Centered User Interface Design: A Practical Introduction. Retrieved October 11, 2001 from http://www.acm.org/~perlman/uidesign.html.

Nielsen, J. (2002a), How to Conduct a Heuristic Evaluation. Retrieved June 10, 2002 from http://www.useit.com/papers/heuristic/heuristic_evaluation.html.

Nielsen, J. (2002b), Ten Usability Heuristics. Retrieved June 10, 2002 from http://www.useit.com/papers/heuristic/heuristic_list.html.

Rieman, J., Franzke, M. & Redmiles, D. (1995), Usability Evaluation with the Cognitive Walkthrough. Retrieved June 10, 2002 from http://www.acm.org/sigs/sigchi/chi95/Electronic/documnts/tutors/jr_bdy.htm.

Rudd, J., Stern, K. & Isensee, S. (1996), Low vs. High-Fidelity Protyping Debate, *Interactions*, January 1996, 76-85.

Technology, Arts and Media Certificate Program, University of Colorado at Boulder (2002). Retrieved June 11, 2002 from http://www.colorado.edu/ATLAS/.