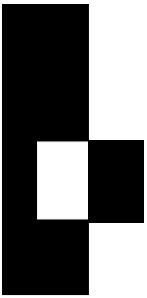

Linking Communication and Software Design Courses for Professional Development in Computer Science

W. J. Williamson, Department of Humanities
Philip H. Sweany, Department of Computer Science
Michigan Technological University

Linked communication and software design courses promote a more intensive and realistic learning environment for computer science students, especially when they work on real projects for real clients. Our students created web-based, educational software for middle-school math classes.

We link communication and software design courses in an attempt to offer computer science students a useful and realistic professional development opportunity. Our students develop educational software for a middle-school math class in a project-driven, service-learning environment. Michigan Tech students typically enroll in the university's general education, technical communication course during their final year of undergraduate education, often at the same time they are enrolled in upper-division courses in their major curricula. Unfortunately, students are only rarely encouraged to overlap these experiences in any way. Although many programs require one or more project-based course for their majors, most students never get to work with a real client on a project that will be used outside the classroom setting. We felt strongly that students would benefit more from both their communication and their software design courses if they could somehow connect their efforts across traditional curricular boundaries and work with a real audiences and purposes. And in fact, this is what we found—students understood the relationship between their technical and communication responsibilities much more fully in both classes than either of us had experienced in these same courses prior to linking them.

In September 1997, we invited computer science students to simultaneously enroll in two courses: (1) a specially designated section of the general education, technical communication course, and (2) a software design course. (Although some students were not enrolled in both courses, most were. Students enrolled in one course or the other still received the

benefit of being in a course designed to serve a real audience and purpose, although their experience was perhaps less rich overall.) Although we maintain final instructional authority in our respective courses, we approached their design as if we were developing a single course. As a result, computer science and communication interests and issues are as fully articulated as we can make them across the two courses. We were driven in our general design by two concerns: (1) we feel it is important to foreground the importance of communication in both courses, rather than encourage students to separate computing and communication; (2) we feel a need to provide students with "real" projects that will challenge them to meet the needs of "real" clients. The project as a whole asks students to develop a software package that can be integrated into the middle-school math curriculum and delivered via the world-wide web. Students work in project teams to develop several written documents in support of their software projects:

* functional description (description of software capabilities);
* design document (software design proposal);
* documentation plan;
* technical description of the software;
* software testing documents;
* user manual;
* software maintenance plan; and
* several progress reports.

In addition, we ask students to evaluate existing educational software packages and documentation as part of their early learning and planning process. Only a few of these documents are completed for credit in one course or the other. Most receive grades in both courses. Finally, students showcase their work in a "software fair" held in an open computing facility at the end of the quarter. In both courses, project-related discussions span issues in software and interface design, teaching and learning strategies, usability testing, and communication design. We encourage students to engage the theories presented in professional literature and if possible to extend those theories through their own work. Although the software design project is the centerpiece of this linked curriculum, students participate in a variety of discussions and assignments that help them develop the expertise they need to complete their work.

Although still under development, this curriculum has been met by students with an increased commitment to connect communication and computer science in their thinking about professional development. We have gathered feedback through a variety of means, including standard course evaluations (which have been high for both courses), anonymous

questionnaires, informal interviews with students, and word of mouth. And some students have gone on to use their course projects as professional portfolio material on the job market. The linked-course project has begun to acquire a favorable reputation among first- and second-year students, many of whom now look forward to participating in the project.

Theoretically, our program design developed along Toby Fulwiler's guidelines for successful writing across the curriculum initiatives. That is, we engaged collaborative learning groups in open-ended assignments that posed real-world challenges. We addressed student writing as managers rather than as teachers, offering guidance rather than grade-oriented commentary. We shared our values as communicators, researchers, and educators by discussing our pedagogical and research goals (183-185). We also looked to service learning scholarship for assistance in drafting our specific project goals. The spirit of this work is captured nicely by Randy Brooks who suggests that "the most valuable service learning includes reciprocity of outcomes: (1) the doing helps the community solve problems or address needs, and (2) the thinking helps the student develop disciplinary skills, community responsibility (ethos), awareness of cultural diversity through the integration of theory and practice" (12). We attribute our success to five strategies we have adopted and that we think might be helpful to others who embark on similar ventures.

(1) Plan curriculum-development time. We invested significant time prior to entering the classroom in discussions of our individual goals, project goals, and pedagogical values. We also discussed external funding sources and possible project clients and how we might approach them. In terms of our own professional development, this was some of our most valuable and rewarding time.

(2) Plan faculty development time. We invested significant time early in the project developing shared expertise in a variety of project-related issues, including educational software design, service-learning design, Java programming, and collaboration. Each area played its part in preparing us to enter into the project as a teaching team.

(3) Find a real client and project. This seems obvious, but projects can really vary. Although we focused on educational software, anything that gives students the opportunities to apply their talent and knowledge while helping the community will create a more enthusiastic work environment. Even simple projects will promote this kind of commitment.

(4) Visit each other's classrooms. For the first part of the term, we were regular participants in each other's courses. This helped promote the spirit of collaboration and connectedness we felt was important to display to our students. They take the courses more seriously knowing that we do too.

(5) Promote departmental consistency. We have promoted this project in our home departments to encourage other faculty who teach these courses to either adopt our approach or promote similar pedagogical values. This is an ongoing struggle.

Readers who are interested in seeing materials related to this project, including course syllabi and a sample software package, can visit the project website at <http://www.csl.mtu.edu/~sweany/educational_software/edsoft.html>.

Works Cited

Brooks, Randy M. "Technical Communication and Service Learning: Integrating Profession and Community" In Marilyn M. Cooper (Ed.) 1995 *Proceedings of the Council of Programs in Technical and Scientific Communication*. Houghton: Council of Programs in Technical and Scientific Communication. 12.

Fulwiler, Toby. "The Quiet and Insistent Revolution: Writing Across the Curriculum." In Richard Bullock and John Trimbur (Ed.) *The Politics of Writing Instruction: Postsecondary*. Portsmouth: Boynton/Cook Publishers. 179-187.

W. J. Williamson is a Ph.D. candidate in Rhetoric and Technical Communication in Michigan Tech's Department of Humanities. e-mail: wjwillia@mtu.edu

Philip H. Sweany is an Associate Professor in Michigan Tech's Department of Computer Science.