Multidisciplinary Education through Software Engineering

Susan A. Mengel
Texas Tech University
Computer Science
211 EC, Box 43104
Lubbock, TX 79409-3104

Locke Carter
Texas Tech University
English
216 EN, Box 43091
Lubbock, TX 79409-3091

Abstract *- Software engineering is a multidisciplinary activity with professionals from many different backgrounds interacting together to build systems. For example, a software engineer may find him/herself interacting with technical writers, application specific specialists, database analysts, production engineers, and marketing professionals. One of the most important people to interact with the software engineer is the client who may not have very much technical expertise. It is critical, therefore, for software engineers to gain experience in interacting with people in other disciplines to facilitate communication.*

Although working with clients is more common, working with students in other disciplines is not as common for software engineering students. To foster multidisciplinary interaction, the Texas Tech Computer Science and English Departments are working together through a sophomore-level software engineering course. In this course, students are grouped into 12 to 17 person teams to work on a project for an external client. The students go through the entire software engineering process from requirements to implementation and place all documents and software on the World Wide Web. The English students review the documents of the computer science students and write the user's manual for the system. A description of the outcome of the collaboration between the two departments is given and lessons learned.

INTRODUCTION

In order to aid students in learning how to integrate themselves into interdisciplinary system development environments, instructors utilize capstone experience courses where students work on small teams for a client. Students typically take these courses when they enter the junior or senior level. Having these courses at the junior and senior level helps to ensure that the students have the technical ability and maturity to complete a project for a client. Depending upon the curriculum prior to the capstone course, students may have done other small projects. Students, however, may have never tried to work on something that would take longer than a semester to implement. Further, they may never have worked in large teams. Quite possibly, they may never have been trained in teamwork and may not completely understand the importance of communication. Also, they may not have learned how to plan and manage their time wisely.

Teamwork, communication, problem solving, and time management are skills that industry would like students to have [3]. These skills certainly can be taught and are best learned through experience. Starting as early as possible to teach these skills and immersing students in situations where the skills must be practiced can help students perform even better in the capstone experience courses.

One of the most natural places to train students in these skills and to have them interact with people in other disciplines is software engineering. A software engineering course provides the mechanism for interaction with a client and with students in other departments to aid in project development. For example, at Texas Tech University, CS 2365 Software Engineering, a sophomore level course in the Computer Science Department, is allied with the courses: ENGL 4367 Developing Instructional Materials (senior level) and MKT 5361 Marketing Administration (graduate level) on three projects for clients. The CS 2365 students focus on project development, the ENGL 4367 students focus on the user's manual and help, and the MKT 5361 students on a marketing analysis for one of the projects. Since the CS 2365 and ENGL 4367 students work more closely together, the focus of the paper is on them.

SIMILAR WORK

Publications for project courses abound in the literature, such as [4, 10]. They, in general, report that students are more motivated in project courses particularly with clients and also address process improvement in the courses. They do contain practical advice, but, unfortunately, they do not always address how someone else could implement the same type of project course. The interested instructor would have to contact the authors of such publications to obtain course materials and even then may not find it easy to build upon the success of the authors.

Since project courses require a large effort to implement, it is preferable to build upon the previous success of others rather than to implement such courses from scratch. Melody Moore [5, 6] led such a successful effort when she was at the Georgia Institute of Technology. Based upon her experience in industry, she developed the Real World Lab sequence of courses at the Georgia Institute of Technology. She also placed projects and course materials on the World Wide Web (WWW) [9] making them available to other interested instructors. She designed the courses to

0-7803-5643-8/99/$10.00 © 1999 IEEE

simulate an industrial software development environment to the point that students had to fill out job applications to be eligible for registration in the courses. Once admitted, students took on different roles, such as requirements analyst, designer, programmer, marketer, and tester. Students could also be promoted as they progressed from course to course (3 total courses).

The students' work on projects that may take anywhere from one to two years to complete, so, they personally may not see a project through to the end. They document their work and train inexperienced members as they progress through the course sequence to maintain continuity of project work. Also, they are mentored by a project advisor and interact with the client to help keep the project on track. Employers prize the Real World Lab experience so much that students have been hired full-time at high salaries upon completion of the course sequence before they finish their undergraduate degree.

CLASSES

Short descriptions are given of CS 2365 and ENGL 4367 to place their collaboration in context. More details can be obtained by visiting http://www.se.cs.ttu.edu/.

CS 2365

CS 2365 Software Engineering is a required course in the Texas Tech University Computer Science curriculum. Students entering the course must have completed the first two required Computer Science Department courses covering programming and data structures. Since CS 2365 used to be at the junior level before moving to the sophomore level (also becoming a prerequisite to a junior-level course), several juniors and seniors still register for CS 2365. It is expected that the number of juniors and seniors will shrink within the next year, but it is also expected that graduate students may register for the course to complete course deficiencies or to augment their education with Computer Science courses.

The software engineering process the students follow is fairly standard and was adapted from Real World Lab. It is specified enough to give students some structure, to ensure some continuity from semester to semester, and to help to reduce the likelihood that students will do time consuming things unnecessary for their project. The process requires several artifacts to be produced during the semester:

- Long-term Plan for semester-to-semester progression,
- Short-term Plan for the current semester specifying schedule and work packages,
- Configuration Management Plan for document format, software format, and WWW site configuration,
- Requirements Document specifying the client's needs,
- Design Document showing the solution structure,
- Software Manual to explain installation procedures and software structure,
- Software for the client, and
- Test Plan to assure software quality.

The process also requires major reviews of all artifacts to take place during the semester. In a major review, the authors of an artifact present their work and a review committee of team members raises issues/questions on the artifact. Major reviews help to improve the artifacts, keep the project on schedule, and help team members stay in touch with the entire project.

CS 2365 differs from Real World Lab as follows:

- Students work only one semester on a project which means they need to learn the software engineering process quickly and they need to be meticulous with the documentation to support the students next semester.
- A project manager has to be chosen at the beginning of the semester, not promoted through semesters.
- Students do not have extensive teamwork experience.
- Students do not typically have the problem solving maturity of a junior or senior.
- Students may still be uncertain about majoring in computer science and may not have the motivation of a junior or senior.
- Students do not have a dedicated lab to do project work.
- Students do not have individual project advisors.
- CS 2365 is a required course.
- Client projects must not require extensive research, be of a critical nature, be on strict timelines, or be developed to commercial success.

The differences must be addressed, but are not insurmountable.

In addition to addressing the differences, other, more implementation-oriented concerns include:

- Contacting prospective clients,
- Setting up a web server for students to post their work,
- Setting up the course materials on the web, and
- Ensuring that students have the tools and training necessary to implement the projects since they may have to use implementation tools not taught in the first two majors courses.

To address these concerns, the instructor contacted prospective clients, utilized a Pentium II PC to set up Microsoft Internet Information Server, used Microsoft FrontPage 98 to post course materials, and provided training to students or found some students with experience in needed tools. It took time to set up CS 2365 after the Real World Lab, but once the start-up period was over, things went fairly smoothly which is encouraging. Clearly, with one PC for a web server and a general computer lab for student use, a client-based project course can be implemented (not necessarily easily, but it is possible!).

The major goal of CS 2365 is to focus students on learning teamwork, communication, time management, problem solving, and process. Given that goal and only one

semester, large student teams are utilized to accomplish only a portion of the project. Large teams:

- help reduce the anxiety of working on a client-based project for one semester (some safety in numbers),
- cause a need for communication skills to develop since team members must keep up with each other,
- cause a need for a process to be followed so that project work can be traced in the current semester and documented for next semester's class, and
- reduce the number of projects so that the instructor can advise them better.

To support students in integrating themselves into the team, lecture material is augmented with team issues. To motivate students to implement only a portion of the project, the projects are large enough to take one to two years.

Large teams are kept on schedule and motivated by the project manager (PM) making the choice of the PM a key issue. Choice of the PM is difficult since all of the students are probably new and unknown to the instructor. Fortunately, the perfect project manager does not have to be chosen, but a few criteria are useful to consider:

- The PM should not be overly motivated so that he/she micro-manages the team,
- The PM should not be under motivated so that he/she simply leaves it to the team to get things done, and
- The PM should not be over committed on time (some students carry a full load of courses and work a large number of hours).

Certainly, the PM could be chosen by chance and the educational process prevail where students simply have to cope, but the client is expecting progress and might not be willing to work with the students if the PM is not helping progress along. The current procedure for choosing the PM is to have all students fill out job applications so that the instructor can split the job applications as fairly as possible among the teams (The PM is later handed the job applications to assign students their respective roles on the team). Then student transcripts are examined to note student background and performance. Usually several PM candidates arise enabling the instructor to talk with them and see who is willing to be the PM.

CS 2365 switched to the Real World Lab format in the Fall of 1998 and due to a former student, an industrial client was found needing work on two projects (several on-campus clients were found, but were referred to a graduate software studio course). The students were split into two teams of 17 people each and at this time were working only with themselves. Although no major problems occurred, many lessons were learned and student feedback was utilized to enhance, streamline, and improve the software engineering process the students used. Specifically, written material was needed on:

- teamwork to reinforce how students should be working together,
- a grievance procedure to mitigate team problems, and
- a means to keep project documents updated during the semester.

These issues were addressed by posting teamwork documents to the web site, by adding a check at the end of a review to ensure a document was updated, and by adding periodic mini-reviews to check documents for currency.

In the current semester, Spring of 1999, the students are continuing the same two projects and have started a third. They are also working with students in other classes and have 13 to 14 students on each team.

ENGL 4367

ENGL 4367 is a required course for English majors with a technical communication specialization. Although the course title is "Developing Instructional Materials," it has been taught as a course in software documentation for several years. Before the fortuitous collaboration with CS 2365, students in ENGL 4367 were assigned the task of finding a piece of shareware on the Internet and developing online and print materials to support it. The procedure for approaching such a task involves the following activities, each of which generates a document:

- Analyzing the nature of the users
- Creating a list of tasks that are evident in the software
- Designing a plan for implementing all documentation
- Conducting usability tests on print and electronic materials
- Creating the materials themselves, including on-line help, print documentation, and tutorials
- Conducting a field evaluation of the documentation after the software product has been released

This approach [1] views software documentation as a highly rhetorical process that calls for the writer to do considerably more work than merely documenting how a finished piece of software works. Since this approach is extremely flexible and powerful, there was no reason not to continue using it for the current project. The instructor's main task, then, was to augment this approach with course units that would address the teamwork and cross-disciplinary nature of the collaboration with CS 2365.

The 19 students enrolled in the Spring 1999 course are almost entirely seniors, and have already completed courses in report writing, style, editing, and document design. These students belong to one of three production groups, corresponding to the three software-engineering projects. Each group has six members: a project manager, an online materials expert, an editor, a lead writer, a graphics/layout expert, and a usability tester. Students selected their own roles and projects based on their own self-assessment, as well as a public reality-check by their classmates. With the exception of one student, who was not able to keep up with the chores of project manager and had to be replaced, this initial placement has been productive.

The groups produce the previously mentioned deliverables and are also expected to participate in CS product meetings and discussions, to publish weekly status reports, and to present several oral presentations to the class. In addition to participating in their group projects, students write six memos, an assessment of the CS projects, and a final essay--all designed to help them evaluate the nature of the real-world curriculum, examine their own attitudes and progress, assess their own group's progress, and synthesize what they have learned over the course of the semester. Students' grades are 50% dependent on their group deliverables and 50% on their individual work.

ENGL 4367 students meet twice a week in the Technical Communications Computer Lab, a 22-machine (Pentium II PCs) facility stocked with industry standard production tools, such as FrontPage, Robohelp, FrameMaker, Photoshop, Director, Authorware, and MS Office. Class time is divided equally between instruction and open lab production time. Students also have access to this lab after normal hours and all day on Fridays. The instructor has taken advantage of the English Department's Windows NT servers and created all course materials in FrontPage, which makes them available to all the students, instructors, and clients participating in the project.

In light of the collaboration with CS 2365, the goal of ENGL 4367 is augmented from its original intent of teaching students course materials with the following components:

- collaboration strategies in order to deal with both the English and the CS groups,
- communication techniques to adjust to the distributed, asynchronous nature of the project,
- self-documentation procedures to ensure that ENGL 4367 students in subsequent semesters may continue the work begun in the current semester, and
- mission statements and job descriptions to help the students understand the way their discipline works with others and the way their own jobs work in synergy with the other students on their teams.

COLLABORATIVE PROCESS

Since CS 2365 is a required course with a certain background required, it is not realistic for students in other disciplines to register for it. It is realistic, however, to enlist other courses to work in tandem with CS 2365 to accomplish a part of the project work or to enhance the project in another way. In order to enlist other courses, the instructor was fortunate to meet an interested professor in the English Department and by calling around, found other professors to participate both now and at a later date.

The following issues exist when considering collaborative work:

- All instructors and students must familiarize themselves to a certain extent with each others' course and student-produced project materials,
- An infrastructure must exist to support communication among all participants,
- Synchronization must occur in all courses so that students can get their work done appropriately, and
- Support of administrators in all Departments and Colleges needs to be obtained.

To address the first issue, the CS 2365 and ENGL 4367 instructors have web sites where course and student materials are posted. This enables everyone the opportunity to get an idea of what others are doing. The web sites, however, are not enough to clear up all miscommunications and misunderstandings that may occur. Additional meetings are required and currently, weekly meetings are taking place among students and among instructors.

The infrastructure to facilitate this communication proved more difficult to address than the instructors initially anticipated. One problem was that the course times were scheduled long before the collaboration was conceived, and neither course met at the same time. Students in both courses were invited to attend each others' classes, but busy schedules made this invitation little more than an ideal gesture. Communication among the students, therefore, had to be conducted electronically, and most students were prevented from augmenting this communication with face-to-face meetings. Initially, it became clear that while CS 2365 students were used to e-mail and utilized it frequently, ENGL 4367 students were not as inclined to utilize e-mail. As time went on, e-mail communication settled down, but another problem surfaced. Information was being lost by the use of private e-mail among the students. The ENGL 4367 instructor had originally set up discussion lists to mitigate this problem, but not all students were utilizing the mechanism to its fullest capability. Since students were now used to e-mail and would be more inclined to use it, additional discussion lists were not considered nor were newsgroups. Instead, two types of e-mail lists were set up:

- one for the client, PMs in all classes, and the CS 2365 requirements team leader, and
- one for team members in all classes.

The software used for setting up the lists is the L-Soft ListServ Lite Free Edition (http://www.lsoft.com) which archives all e-mail to the web.

Synchronizing student work is a critical issue and sometimes may hinge upon the CS 2365 students' ability to get their development work completed. Also, depending on how closely students in other classes need to work together, students may need to review each others' work. For example, the ENGL 4367 students must review the CS 2365

- Short-Term Plan to know when milestones occur,
- Requirements Document to understand client needs,
- Design Document to evaluate the user interface and learn how the software will work, and
- Software to perform usability testing.

The CS 2365 students must review the ENGL 4367

- User Analysis,

- Documentation Plan,
- Usability Report, and
- Final Materials.

Instead of making the reviews a formal process and having to schedule rooms and times, the document authors can send a "request for comment" to the team members in the other course to review the document and send back comments.

Political issues can hamper any collaborative work between departments in separate colleges. Fortunately, support was unanimous among Department Chairs and Deans. They were consulted before the classes were allowed to start working together. The loose coupling of the courses was a benefit here as resources and time did not become matters of contention. The infrastructure fit in naturally with existing departmental resources.

CONCERNS, BENEFITS, AND RESULTS

Some issues of concern exist with project-based courses that should not be ignored, such as making sure:

- the clients are working well with the students,
- the students understand they are only to post project related materials to the web site and avoid the use of copyrighted materials,
- the students are to facilitate communication among each other and not intentionally withhold information,
- the client understands the project software is provided without warranty and neither the instructor or students have any responsibility to the software after completion of the semester, and
- the students do not overwork themselves since they seem to want to complete the entire project in one semester (PMs are counseled to schedule students no more than 6 hours per week during low work periods and 10 hours per week during high work periods).

These issues should be addressed early with e-mail or meetings to reduce the possibility of problems arising.

The cross-curricular nature of the collaboration between CS 2365 and ENGL 4367 may also give rise to problems associated with the expectations of the different disciplines. While the literature does not deal explicitly with Technical Communication students working Computer Science students in a real-world collaboration, Beheler and Malar [2] do describe student engineering teams that incorporated writers and editors for a long-term project. They report that writers lacked initiative and did not feel respected by their engineering teammates. The instructors attempted to address discipline-specific issues through detailed mission statements, organization charts, and cross-classroom visits.

A number of benefits, however, arise with project courses that have been enumerated in the literature and that overshadow the concerns. Students are more motivated in working with a client, feel more challanged, and enjoy solving the problems inherent in the project. They feel like they are gaining valuable experience that will enhance their attractiveness in the job market. One student even commented this semester that 50% of the interview questions he was asked came out of the project experience and material he was learning in CS 2365. Moreover, students think they are developing skills they will take with them the rest of their career. PMs are especially appreciative since they may not get an opportunity to become a manager until they graduate and have a few years of experience. A serendipitous effect occurred with two of the PMs in the Fall 1998 CS 2365. They decided they wanted to climb the career ladder and be project managers rather than remain development team members.

Benefits to ENGL 4367 students are due to the process-oriented nature of the collaboration. Instead of coming into the project at the end and documenting a stable piece of software, the students are involved in the development process. They attend meetings with the client and the software engineering teams, their opinions on the graphic user interface are apt to be considered and implemented, and their analysis of the users may be folded into the software development process.

Student feedback plays an important part in both classes and occurs throughout the semester. Students pinpoint trouble points in the process enabling the instructors to make possible changes during the semester or for next semester if the trouble points are not critical. Overall, students experience higher frustration in dealing with client-based projects and communication in large teams, but they feel the experience is worth the extra effort they have to expend.

There are benefits for the instructor as well. Instructors get to know many of the students better than if they had just done a lecture course with small assignments. They may find out their students are already working with clients and have mature skills in web or database programming. They may also see students with relatively low grade point averages blossom in this setting and become key team members. They may also find students with low motivation more easily and get an opportunity to work with them early in the semester to help them learn how to become a contributing team member.

Benefits to the client are several. If the client is an industrial client interested in hiring students, they will get to know students much better through sponsoring a project than through a short interview. The client will also get some needed software and a project history through the artifacts if the client wishes to take the software to a professional for further maintenance. The client will have some satisfaction knowing that he/she helped with the educational process.

The interest generated in the collaboration between the two courses has been quite extraordinary. Already a journalism student has written an article [7] for the Texas Tech student newspaper, *The University Daily*, on the courses. The journalism student found out about the courses from one of the project students. Another project student spoke to an English Ph.D. student in technical communications interested in interdisciplinary

communication, and now the Ph.D. student is observing the CS 2365 and ENGL 4367 students as part of his dissertation work. The project students are telling prospective employers about what they are doing. A student in the Fall 1998 CS 2365 course is analyzing the current software engineering process this semester for what needs to be done to move the process up to Level Two of the Software Engineering Institute's Capability Maturity Model (CMM) [8]. Although some of the interest was predicted and hoped, it has exceeded expectations. It is very gratifying to see how the collaboration has been of use in so many areas.

FUTURE DIRECTIONS

When deciding to follow in the steps of Real World Lab, it became clear very quickly that the process followed by the students would have to be constantly improved and updated. It is planned to do this through student feedback and analyzing the process for compliance with the CMM. Other advances in the software engineering literature and in industry will be considered as well.

Even though it was found that a dedicated lab was not necessary for students to work on the projects, the logistical complications of students finding places to meet and getting necessary software on the general lab computers creates minor frustrations. It is planned to develop a dedicated lab equipped with the necessary hardware, software, and meeting rooms, for students to work on their projects.

Additional collaborations with other classes are planned in the future to enhance the students' educational experience and to enhance the work on the projects. For example, industrial psychology and communication students could observe project students as they meet and communicate to find ways to reduce communication problems. Communication students could observe student project presentations and help students present material better. Management students may assist project students in better ways to schedule time and resources. Students in other disciplines may have the expertise needed to complete specialized hardware/software projects.

ACKNOWLEDGEMENTS

All terms known to be trademarks or registered trademarks are capitalized.

Special thanks to:

- Melody M. Moore and Jon Preston for their help in explaining some of the nuances of Real World Lab,
- OnBoard Software, Inc., and the Texas Tech University Recreation Center for sponsoring student projects,
- David Spencer, Bill Hudson, Charlie Barondes, Scott Kirk, Jon Burgin, and Betty Blanton for making the projects possible and working with the students,
- Dan Cooke - Chair of Computer Science, William Marcy - Dean of the College of Engineering, Madonne Miner - Chair of English, Carolyn Rude - Director of Technical Communication, and Jane Winer - Dean of Arts and Sciences, for their support,
- Fred Kemp for helping with the technical details of the implementation platforms, and
- CS 2365 and ENGL 4367 students for their work.

REFERENCES

[1] T.T. Barker. *Writing Software Documentation: A Task-Oriented Approach*. Boston: Allyn & Bacon, 1998.

[2] T.M. Beheler and J. Malar. "Student Collaboration: The Ups and Downs of a Real Life Project." *Proceedings of the 42nd Annual Conference of the Society for Technical Communication*. Arlington, VA: STC, 1995, pp. 87-90.

[3] P.J. Denning. "Educating a New Engineer." *Communications of the ACM*, vol. 35, no. 12, December 1992, pp. 83-97.

[4] W.B. McCarty and G.T. Plew. "How Mature is Your Software Process?" J.L. Diaz-Herrera, ed., *Proceedings of Seventh Conference on Software Engineering Education*, New York, NY: Springer-Verlag, 1994, pp. 555-564.

[5] M. Moore and T. Brennan. "Process Improvement in the Classroom." R.L. Ibrahim, ed., *Proceedings of Eighth Conference on Software Engineering Education*, New York, NY: Springer-Verlag, 1995, pp. 123-130.

[6] M. Moore and C. Potts. "Learning by Doing: Goals and Experiences of Two Software Engineering Project Courses." J.L. Diaz-Herrera, ed., *Proceedings of Seventh Conference on Software Engineering Education*, New York, NY: Springer-Verlag, 1994, pp. 151-164.

[7] T. Nishimura. "Two Classes Offer Hands-On Experiences for Students." The University Daily, vol. 73, Issue 86, February 10, 1999, p. 2.

[8] M.C. Paulk, C.V. Weber, B. Curtis, and M.B. Chrissis. *The Capability Maturity Model: Guidelines for Improving the Software Process*. Reading, MA: Addison-Wesley, 1995.

[9] Real World Lab, Georgia Institute of Technology, http:// www.cc.gatech.edu/classes/RWL/Web/.

[10] L.H. Werth. "An Adventure in Software Process Improvement." J.L. Diaz-Herrera, ed., *Proceedings of Seventh Conference on Software Engineering Education*, New York, NY: Springer-Verlag, 1994, pp. 191-210.