Simulation of Real-world Development Processes in Computer Science Classes

Paula Gabbert and Kevin Treu

Department of Computer Science, Furman University, Greenville, S,C USA
pgabbert@furman.edu

Keywords: simulation, classroom teaching/practice, collaborative learning, learner-centred learning, pedagogy

Abstract: Educators have long recognised that learning takes place most effectively in an environment in which the student is engaged in the process of their own education. One method for doing so is to put more control over the teaching and learning of course material into the hands of the students themselves. This paper describes an approach to learner-centred learning that involves the simulation of industrial software development practices as the central teaching technique. Our initial experiences with two courses – Web programming and systems analysis – are discussed in this paper.

1. INTRODUCTION

Engaged learning, inquiry-based learning, learner-centred teaching and other similar terms have gained prominence in recent years as educators strive to get students to take a greater role in their own learning process (Abernethy, Gabbert and Treu 1998, Jackson, Manaris, and McCauley 1997, Leasure 1998, Walker 1997). The lecture-based method of teaching has fallen into disfavour to some extent as teachers attempt to be 'guides on the side' rather than 'sages on a stage'.

Though we are not prepared to sound the death knell of the traditional lecture style of teaching, we have had success with an approach to teaching certain computer science topics that de-emphasises lecturing on the part of

the professor. In this paper we describe this simulation-based approach and the courses in which we have applied it.

What we have attempted to do is to simulate real-world industrial development environments as closely as possible within the constraints applied by the academic model. Significant elements of this approach include:

- Real projects and deadlines
- Goal-oriented study and research
- Student ownership of projects
- Teamwork with clear divisions of responsibility
- Progress and summary reports.

Though lecturing is still a part of the process, lectures are delivered primarily by the students themselves. Their study is demand-driven – motivated by the need to learn the concepts and skills required to complete a project. They are required to learn these by any of various available resources, and are expected to instruct fellow classmates in what they have learned.

The teacher in this model has considerably less responsibility for writing lecture material, but more responsibility in terms of organising available resources, making them available, and guiding students in the appropriate directions. They must also be prepared to be flexible, as students choose to explore different concepts or skills than those anticipated.

We have applied this approach to two different courses: Web-Based Application Design and Systems Analysis. In this paper, we discuss our approach to teaching these subjects, the various implementation decisions that were made, and the results observed.

2. APPLICATIONS OF SIMULATION-BASED LEARNING

Our interest in teaching using simulation of real-world work environments was a natural outgrowth of our interest in simply teaching these subjects effectively. In essence, the course material required that something other than a lecture-based approach be taken. For the Web programming course, the sheer volume of available information and its changing nature made a traditional approach not feasible. For the systems analysis course, we found that it is far more effective to simulate a business environment so that students could observe the interconnections between modelling tools from CS and project management techniques from business.

2.1 Web-based application design

The basic objective of this course is to build a foundation for understanding some of the leading technologies that contribute to the power of the Web. In addition, students learn what is required to build state-of-the-art interactive Web sites.

The same explosive growth that makes the Web so important and interesting as a subject of study also presents significant challenges in the design of a course such as this. Covering all of the various Web technologies in detail would require multiple academic terms.

Also, the strong appeal of Web programming makes it a subject for many enthusiastic hobbyists outside the academic realm. Thus it is expected that there is expertise to be found among the collective participants in the class that would be difficult to find in any one person. In other words, it is likely that for any given topic, there is at least one student whose expertise exceeds that of the professor. Our experience suggests that this is a somewhat unique situation, but one that presents interesting opportunities.

2.1.1 Basic course organisation

These observations point the way to the use of the simulation-based format. In fact, they essentially demand it. The basic principle of the class is that all class participants (not just the professor) are responsible for learning certain topics and teaching them to the rest of the group. The course work is driven by a series of four case studies, real problems with real clients to which appropriate solutions are applied. In a sense, the participants in the course act as a fledgling Web design company (with the professor as CEO) which has deadlines, teamwork, performance reviews, and a vested interest in making sure that everyone has learned the technology.

The basic topics covered include:

- Dynamic HTML (HTML augmented with client-side scripting)
- Active Server Pages (server-side scripting and database interface)
- Database implementation
- Macromedia Flash animation
- Server maintenance and security.

The class is divided into four working teams. For each of the four case studies there is a Design Team, an ASP Team, a Database Team and a User Interface Team. The Maintenance Team is made up of members from the other four teams. The teams rotate from topic to topic with each case study, so every student has a chance to work in each area. Each team is responsible for doing the following for each case study:

- Writing a design document at the outset of the work

- Preparing and delivering a lecture on the technology used
 - Installation and testing of the team's portion of the site on the server.
- There is a project manager designated for each case study. Simulating as closely as possible the duties of a corporate manager, this person is responsible for:
- Scheduling and holding team meetings
 - Meeting weekly with the professor and the other managers
 - Maintaining a journal of team activities
 - Preparing brief written and oral progress reports
 - Assigning tasks to team members
 - Co-ordinating the development of the team's design document and lecture
 - Reviewing the work of the team members.

Development of each case study follows a three-week schedule. The class as a whole initially meets with the client to determine the requirements of the project. Each team subsequently drafts a design document for their area, which is refined in the first manager meeting and then distributed to all teams.

During this process, the teams are responsible for learning the basic tools of their area. An extensive online resource list is provided for this purpose. In addition, as the course progresses from case study to case study, previous work is available as a guide.

In the second week, teams are expected to make concrete progress towards implementing the designs. The manager meeting focuses on progress reports and the resolution of any design conflicts that arise.

During the third week, work is completed on each component of the case study and the site is uploaded and tested. Each team then delivers its final report. These lectures build upon previous presentations, so they do not all cover the same ground.

In class, time is devoted to the various stages of this process – design, implementation, troubleshooting, and various oral reports. In addition, special topics are introduced, by the professor, guest speakers and preferably students in the class. Topics covered during the most recent offering included Java servlets, JDBC, XML, COM objects, VRML and Cold Fusion. Though students are not able to get hands-on experience with all of these topics, exposure to them helps expand the base of knowledge that is being constructed.

2.1.2 Grading

Assigning grades to the students is a difficult challenge for a course organised in this fashion. In an effort to remain true to the notion of

simulating an industrial environment, each student was primarily evaluated based on the quality of work produced, rather than exam results.

That decision raises the concern about whether or not students – who do not in reality face the prospect of being 'fired' for poor effort – are actually doing their share of the work. We attempt to assess this via written peer evaluations. At the conclusion of each case study, students are required to evaluate the work of their team-mates. These peer reviews factor into the grade each student receives for a case study, with the manager's review being weighted more heavily.

It is important, of course, to ensure that students are actually learning the course material appropriately. To assess this in the absence of written exams, the students are encouraged to take active roles in the class through several incentives that earn them points towards a higher grade, rather than higher pay. Included among these incentives are taking on the role of project manager or maintenance team member, taking an active part in the final lectures, preparing and delivering lectures or tutorials on one of the special topics and discovering or reviewing new tutorials. A high level of participation in these activities beyond the project work makes it easy to confirm that a student is assimilating the material.

2.1.3 Results

The course has, by all indications, been successful. Not only in terms of student enthusiasm but also in terms of student effort and quality of work produced. Quantitative information about the success of the class has been gathered via student surveys. A sampling of the results paints a positive picture. The average response to the question "How much did you learn in this class?" was 4.5, where a response of 1 indicated 'Nothing', 3 indicated 'An average amount' and 5 'A great deal'. 12 of 16 students responded to the anonymous survey.

The average response to the question "How effective was the case study approach in helping you learn the basic course material?" was 4.33, where 1 indicated 'Ineffective', 3 'Reasonably effective' and 5 'Very effective'.

The students were also asked this question: "Would you consider it to be a good idea for the class to be made more professor-focused (e.g., based more on professor-delivered lectures)?" Potential responses to this question ranged from 1 meaning 'Strongly disagree', to 3 'Neutral', to 5 'Strongly agree'. The average response was 2.33. The comments of those who responded with a 3 or 4 (there were no 5's) expressed frustration with some of the student lectures that were ineffective. Our conclusion is that the students are generally receptive to the idea of taking more control over their

own learning, but that more must be done to ensure quality work on the parts of their classmates.

The anecdotal results were also favourable. Students in the class maintained a high level of enthusiasm and clearly learned the material even in the absence of the threat of an exam. From a teaching perspective, it has been energising to be in the position of learning alongside the students.

The results of the case study work for this course, along with additional information about course organisation and grading, can be found on the Web at <http://s9000.furman.edu/~treu/cs95/>.

2.2 Systems analysis

The Systems Analysis course is a project-based course that focuses on the analysis and design of large information systems. The projects used in the course are much larger in scope than those typically found in introductory level courses so that students will see the benefit of using the systems modelling tools that are developed in the course. A goal of the course is to engage the students in an interactive learning environment so that they experience the information system development process first hand.

The course also focuses on the skills necessary for the management of large projects including cost benefit analysis, project scheduling, and group management. Communication skills for meeting management and presentations are also developed. Many students have the opportunity to participate in internship experiences outside of the classroom at some point in their academic career, but rarely in these internships are they given leadership roles in the development of a project. These additional opportunities are becoming more important within the CS curriculum (Little et al. 1999).

2.2.1 Basic course organisation

In the last two offerings of our *Systems Analysis* course, the classroom was transformed into an IS development company with the professor as the CEO (Abernethy et al. 1998). The class was broken into 5-6 development teams where each team was responsible for their own information system. The classroom is our meeting room where members of the development teams discuss important issues relating to the projects, disseminate related material to the members of the team, and support each other in their efforts to learn and employ new technologies and techniques. The class attempted to involve all of the students in each of these activities. Central to this effort was a class web site that was designed and maintained by members of the class. The web site was used as a vehicle for disseminating appropriate

technical information, communicating with members in the class, and organising meeting materials.

Each class meeting had a facilitator who organised the meeting agenda, posted the agenda on the web one day before the class meeting, and then ran the meeting. If a student wanted to make a presentation on a topic, they contacted the facilitator who then put them on the agenda. In addition, there was a meeting secretary who took minutes for the meeting and distributed those minutes on the web. Students were encouraged to prepare appropriate material to contribute to the class meetings, but no official assignments were given. Possible course material included answers to questions found at the end of chapters in the textbook, reviews of journal articles, descriptions of job interviews for analysis positions, or any other pertinent information.

Course material was also presented by the instructor, but students were expected to reference the agenda and read the related chapters before the class meeting. Students were encouraged to present the review problems to start the meeting discussions. The instructor was responsible for facilitating discussions on the topics and providing additional materials on the analysis and design tools. These 'lectures' were developed on the spot based on the needs arising from the discussions. In addition, there were weekly case studies to provide students with larger applications for the tools and techniques being studied.

2.2.2 Grading

Student participation in all of the class and web site activities was encouraged by the requirement to earn a certain number of 'participation points' during the term. Participation points were awarded when activities were completed and a given number of points had to be earned during each half of the course, primarily so that students did not wait until the last week of the course. Students were required to explore a variety of activities by limiting the number of points allowed for each one. For example, students could only earn points for facilitating one class during the term, five review questions, 2 article reviews, etc.

Another component to the course grade was the team project. All members of the team were awarded the same grade for the project. In order to ensure fairness among the group, peer evaluations were used to determine whether all members were supporting the team effort. In addition, the professor observed group dynamics during the weekly laboratory when groups met to work on their projects. Although not all of the project work was completed during these two hours, it was clear from these observations who was contributing to the group effort and who was not.

Finally, this course also lent itself to a midterm and a final exam to determine if the students were understanding the tools from the course. The course coverage is separated into two distinct sections, one for analysis and one for design. Hence, the midterm covered the analysis concepts and the final covered the design concepts.

2.2.3 Results

Student response to the first offering of this course was initially very tentative. They appeared to be confused and apprehensive about the expectations of the class participation. But as more students made presentations and updated the Web site, interest increased. It appeared that they simply needed someone to 'start the ball rolling'. Soon after the midterm point, all the meeting facilitator and secretary slots had been taken. The Web site was fully functional by then as well and had a variety of pages to help organise the class material. More exciting was the number of students offering to make contributions to the class that had not been originally suggested. The students seem to appreciate the chance to learn from each other and try to find new and interesting ways to accomplish the goals of the class. In the second offering of the course, the students became engaged much quicker as 'word was out' about the expectations of the class and the benefits of the classroom environment.

3. CONCLUSION

Simulation of real world processes is an important application of computer science. Accordingly, it seems natural for practical, applied CS concepts such as Web programming and systems analysis to be taught by simulating the environments in which professionals practice them. Though this idea is not unique, we are not aware of the use of such simulation-based learning as the organising principle of an entire course. In this paper we have described experiments with two courses that are designed to use this approach.

Numerous benefits have accrued. First, both courses are solid examples of engaged learning (learner-centred) principles. Rather than being the central focus of a student's attention, the professor becomes more of an organiser and guide while a student learns by exploration. This is meant to mirror the process by which a corporate project leader 'teaches' – he or she makes the objectives clear, makes the resources available, and provides guidance as the team member learns what is necessary.

Second, student enthusiasm for both courses is very high, as indicated by both quantitative and anecdotal results. The information to be learned is not viewed as being tedious or boring when it is leading to a product that will actually be used, and when there are team-mates relying on you to learn it.

Third, students have a sense of ownership of the information when they were primarily responsible for learning it. The pride that they feel in accomplishing their goals is a tremendous motivating force.

Finally, we have observed that experiences such as these in the undergraduate curriculum serve students extremely well in their professional pursuits after graduation. In the short time that these courses have been taught with the simulation-based approach, many graduates have reported that they were among the most useful and relevant of all their college courses.

REFERENCES

- Abernethy, K., Gabbert, P. and Treu, K. (1998) Inquiry-Based Computer Science Instruction: Some Initial Experiences. *6th Annual Conference on the Teaching of Computing and 3rd Annual Conference on Integrating Technology into Computer Science*, August.
- Jackson, U., Manaris, B. and McCauley, R. (1997) Strategies for Effective Integration of Software Engineering Concepts and Techniques into the Undergraduate Computer Science Curriculum. *ACM SIGCSE Bulletin*, Vol. 29, No. 1.
- Leasure, D. (1998) Improving Student Learning Through Real-World Internet Projects. *The Journal of Computing in Small Colleges*, Vol. 14, No. 1.
- Little, J. C., Granger, M. J. (co-chairs), Boyle, R., Gerhardt-Powals, J., Impagliazzo, J. Janik, C., Kubilus, N. J., Lippert, S. K., McCracken, W. M., Paliwoda, G. and Soja, P. (1999) Integrating Professionalism and Workplace Issues into the Computing and Information Technology Curriculum. *ACM SIGCSE Bulletin*, Vol. 31, No. 4.
- Walker, H. (1997) Collaborative Learning: A Case Study for CS1 at Grinnell College and University of Texas-Austin. *ACM SIGCSE Bulletin*, Vol. 29, No. 1.

BIOGRAPHY

Paula Gabbert is an Associate Professor of Computer Science at Furman University. She has done extensive development of new educational methodologies in the CS curriculum including foreign study programs, simulation, and peer review.

Kevin Treu is an Associate Professor of Computer Science at Furman University. His professional interests include Web programming, interactive Web environments, and innovation in computer science pedagogy.