Web Based Collaboration for Introductory Programming Courses

Christopher Egert [1], Mary Flanagan [2] and Deborah Walters [3]

[1]*Department of Computer Science and Engineering, University at Buffalo, Buffalo, NY 14260 USA*
Tel:(716)645-3180x130, egert@cse.buffalo.edu
[2]*Department of Media Study, University at Buffalo, Buffalo, NY 14260 USA*
Tel:(716)645-6902x1491, maryf@acsu.buffalo.edu
[1]*Department of Computer Science and Engineering, University at Buffalo, Buffalo, NY 14260 USA*
Tel:(716)645-2711x1149, walters@cse.buffalo.edu

Abstract: This paper addresses the need for web-based collaborative tools within introductory Computer Science and Engineering programming courses. The World Wide Web has been a tremendous influence in how we present Computer Science and Engineering material to students. The Web's influence can range from the passive presentation online lecture materials to active demonstration of Computer Science and Engineering principles through Applet based animations and simulations. One of the most powerful uses of the Web is to foster collaboration and communication among students; a skill often overlooked in introductory classes. Although there are several web-based threaded discussion systems available, the collaborative method is based on purely textual communication. In order to increase the usability of such web-based collaborative systems, the focus must be changed from a textual framework to one that supports better integration of both text and active media content. To this end, we have adapted previous work in media based collaborative courseware to target the needs of the Computer Science and Engineering population. The new system allows students to submit, view, and collaborate on projects using both individual and group work models. The system has been re-engineered to allow students to develop real-world techniques such as specification reading, component integration, and user interface design, while minimizing instructor effort to maintain the system. To verify the design effectiveness of the new system, we have been performing usability tests with focus groups.

Keywords: world wide web, collaboration, courseware, submission systems

1. Introduction

The World Wide Web has proven to be tremendous influence pertaining to how material is presented and taught in introductory Computer Science and Engineering programming courses. The Webs impact can be as simple as a distribution medium for the dissemination of course notes and classroom materials. It can also be used as a persuasive tool to help students understand key concepts through the use of animations and interactive simulations. Web enabled programming technologies, such as Java Applets have also impacted the way we teach introductory programming.

With all of the changes that the web brings to introductory programming courses, one of the most important aides that the Web can provide is a framework to promote collaboration and communication. Although curricular assessment has called for greater attempts to provide collaboration at earlier points in a student's academic career [6], it often proves difficult to provide such a forum within the constraints of large lecture halls. Web technology can help to foster collaboration to supplement the introductory programming environment.

Over the last several years, we have seen several development efforts to transform traditional computer mediated collaboration technologies to embody Web capabilities. Collaboration applications such as threaded newsgroups, mail systems and talk systems now have web-based counterparts that mimic their basic functionality. Such web based collaborative components are often integrated in larger courseware packages to provide a collaborative ability. Such systems can encounter mixed success; findings show that they are not used as often or as effectively in larger introductory groups of students [5]. Such shortcomings are often times attributed to the size of the course or the lack of familiarity to technology by the students. However, this argument is insufficient by itself; one only has to look at the success of online communities such as MOO's and MUSHes to see that large groups with varying levels of skill and background can have meaningful interaction within a particular context [1].

The question that we pose is how can we increase the success the collaboration when the topic is programming and program design? Current tools make it easy to manipulate the text of a message, allowing the user to annotate, include, and reference textual passages from earlier points in a message thread. When activity based content comes into play, such as the execution and display of a particular Applet that a student wants to comment on, the interaction becomes much more involved. Students using web-based newsgroups often do not see the Applet at the same time they are formulating responses or reading other peoples comments. Worse yet, students often find that they are downloading the same Applet over as part of each reply.

Our solution to the problem was the development of a collaborative courseware system in which the focus of collaboration shifted from a purely textual one to a framework that could support collaboration based on media content. The goal of this work is not to merely create a technological solution to this problem, but to understand the work model student's employ to solve individual and group problems and how to embody this model within a social context. In order to perform this work, we decided to adapt earlier media-based collaborative software developed for digital media classes.

2. The IOS System

The Integrated Online Seminar (IOS) system was originally developed to address the collaborative needs of digital arts students, participating in a distance learning user interface design class [3]. For each project, each student had to design and develop interfaces for web applications using industry standard graphical design packages. Early in the course development phase, existing courseware and web based newsgroup systems were explored to facilitate media based collaboration. Many of the systems would not allow for the image and the text to be displayed simultaneously, and the few that would required that the image be downloaded each time a reply was made by a student. Since many of the students enrolled in the class only have access to home computers with standard modems, multiple downloads of the same image proved unacceptable.

As such, the original IOS design had several goals. The first goal was to create a spatial metaphor for the posting of student work. Information was divided into three logical areas; a course area, student area and a project area. Each student's submission posted to the IOS system was contained in a fixed location. Once students became familiar with the system, they could easily find and critique the work of their peers.

The second goal was to minimize the administration of the system. Although the IOS system possessed a rigid spatial metaphor for the location of students and their work, it also allowed for students to have a certain degree of flexibility in how they used the space once they got there. For example, students could either post simple images or they could post complex web sites without the intervention of the instructor to configure the space for their particular needs. Also, automated logging assisted the instructor in keeping track of an individual student's progress with an assignment.

The third goal was to balance comfort and security with the desire to establish a permanent online body of work which the student could reference. To accommodate this goal, IOS was supplemented with a simple security system; all students within the course were given full read and write access to their areas up to the due date. After the due date, only people in the class could critique of comment on the work. Observers of the course, such as potential employers or interested students, were able to view the interactions. However, the actual collaboration was protected against outside comments or interactions.

The assessment phase of the original IOS system proved to be highly successful [4]. Many of the students did not feel restricted with its navigational structure and had participated more fully in discussions. Most students also appreciated the balance between encapsulating the specifics of the submission technology with the freedom to creatively post their work. Most of the criticisms of the system were positive in nature, requesting different ways to monitor and filter the content.

3. Modifying the IOS Model

Although the IOS system was a success with digital media students, there were several changes we had to make in order to make it successful for use within Computer Science and Engineering courses. At the simplest level, we needed to incorporate the inclusion of Java source and possibly Applet class files into the submission system, however we found that there were several changes that we had to consider to better support the programming framework. The following section outlines some of the fundamental changes that were made to the system.

One of the first changes that needed to be made was the inclusion of an assignment posting and scaffolding. The original IOS system relied on an electronic mail to distribute assignments to students. We found that among programming students, more visualization of the problem was needed to get them off to an effective start. In order to attack this problem and to create an initial scenario in which the student could become familiar with the IOS environment, assignment posts were treated the same as project submissions. Students in the programming class

could actually post to the critiques and comments section of the assignment to in order to clarify the project instructions. This helped to provide familiarity with the framework as well as provide uniform access to assignment information.

The second change involved an analysis of work patterns within large groups of students. Traditionally, students are given individual assignments in with they develop an entire solution or extend a given problem to completion. This is often at odds with industry work dynamics in which teams of programmers contribute portions of the solution to the entirety of a project. Using Constantine's classification of work patterns [2], we decided to provide support mechanisms for the parallel, individual work model as well as the group, concurrent model. In order to accommodate these patterns, we created abstractions not only for the student project relationship, but the group-project relationship via which individuals contribute to a larger program. We made look and feel of the student project relationship similar to the group project relationship to help ease our student's transition to this model.

Support of varying work models also meant that we had to create alterations to create an appropriate social context. One such social pattern we incorporated was the idea of hierarchical groups. The need for this was two-fold; it allowed students to experience different group roles at different parts of the integration process as well as to help acclimate students to structures analogous to real world corporate organizations. We also created mechanisms that allowed for the transfer of content from one group in the hierarchy to another. We also added more feedback mechanisms to give students better self-assessment mechanisms. Students could set graphical indicators to denote when they were done with a particular section, or they could use percentile indicators to establish their progress in the assignment. Other mechanisms included alterations to the upload system to provide feedback to students uploading submissions.

Our final change to the system involved the work requirement for the instructor. We made further changes to allow students to self-administer the direction of the collaboration. We decided to give the students more responsibility with how the shape of the course flow went without disrupting the underlying navigational model.

4. The IOS2 Walkthrough

In this section, we show a walkthrough of a student working on a project using IOS2, the new version of the IOS system. The student was assigned to work on an introductory programming assignment with a series of teammates. We adapted an introductory programming assignment used at Brown University [7] such that each student was responsible for completing one graphical component of an animated cloud in an interactive environment.

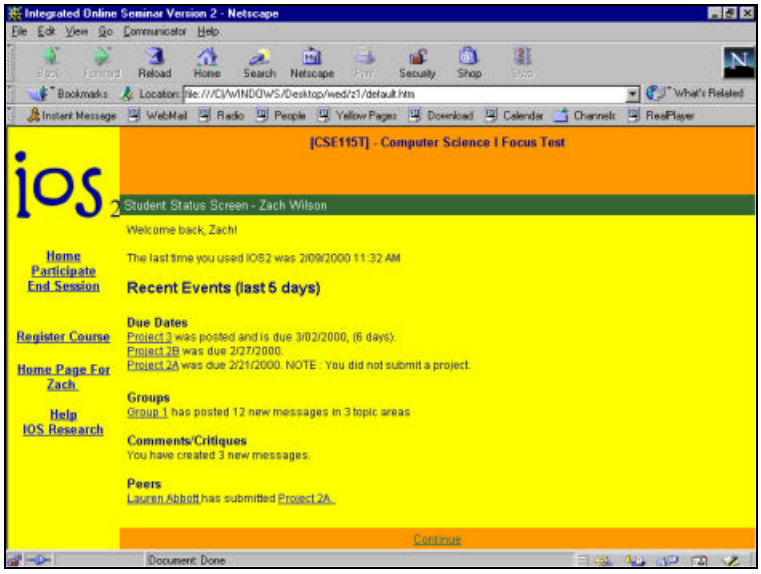

Fig. 1. IOS2 Status Screen

When a student first enters the IOS system, and is validated, he or she is greeted with a status screen indicating several important factors. As seen in figure 1, the student can quickly determine which projects are due, the number of comments made about their work, their progress as a member of the community, and important peer as well as group information.

After viewing the status screen, the student is presented with the main menu. The main menu allows the student access to critique the contents of peer submissions, interact with groups to which they belong. A student starting a new project can then choose to view the assignment as shown in figure 2. The student can also download supplemental materials pertaining to the project.

Fig. 2. IOS Project Description

Once the student is familiar with the goals of the project, a scaffolded example can be viewed as shown in figure 3. The student can interact with the sample as well as ask questions or comment on the assignment. The student then works on their own machine with the downloaded supplemental material, which provides a skeletal framework complete with "stub" code and interfaces for the standard parts the student must complete.

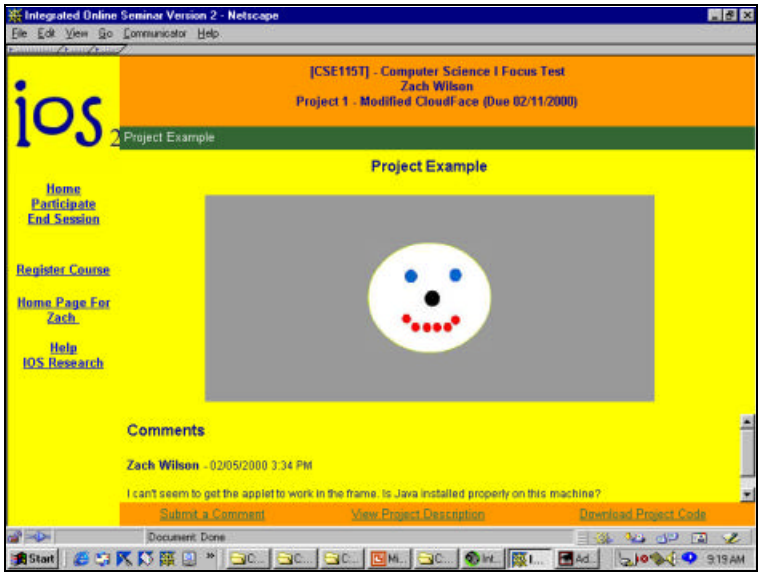

Fig. 3. IOS Scaffolding Example

When complete, the student accesses the submission upload section of the system. As seen in figure 4, the student can then select the files on the machine to upload as well as the portion of the assignment that they have completed.

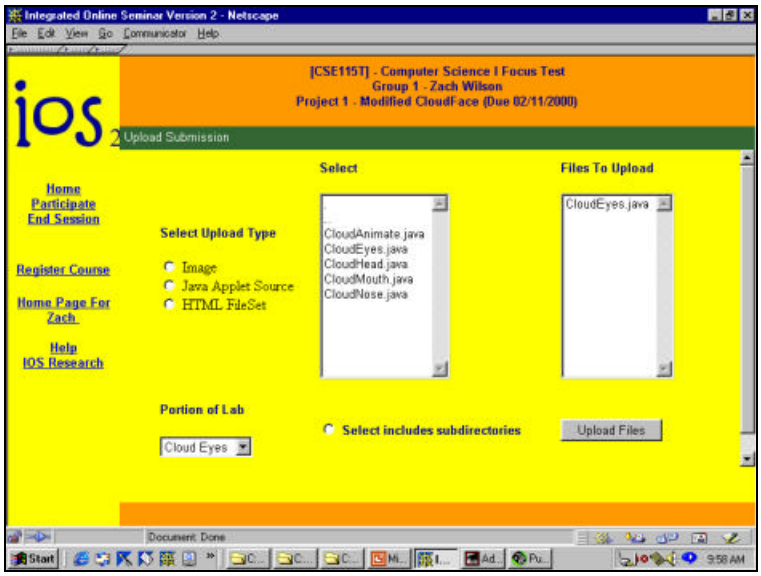

Fig. 4. Student Upload Process

5. Testing and Conclusion

We have started testing the new version of IOS with small focus groups of introductory programming students consisting of 12 students. We have conducted usability testing several ways, including cognitive walkthroughs, user observation, user surveys, and the gathering of statistics from the server to ascertain specific usage patterns [XX]. Our initial testing has revealed that one of the most important factors contributing to the success of students is to make sure that students feel that they can successfully complete a group project if the group dynamic fails. Our approach of creating skeletal code for introductory lab experiments has proven crucial in the creation of a comfortable group atmosphere. Although students are not directly encouraged to use the skeleton code as their only testing criteria, we have found that students who know that the skeleton exists are more comfortable with the model. We plan on conducting more exhaustive testing in the Fall of 2000.

4. References

[1] Bruckman, A. and Resnick, M., "The MediaMOO Project : Constructionism and Professional Community", *Convergence*, 1(1), 1995

[2] Constantine, L., "Work Organization : Paradigms for Project Management and Organization", *Communications of the ACM*, 36(10), pp. 35-43, 1993.

[3] Flanagan, M. and Egert, C., "The Course Submission System: Providing Seminars on the Web", *Proceedings from the Association for the Advancement of Computing in Education Webnet98 Conference*, Orlando, FL, pp. 313-317, 1998

[4] Flanagan, M. and Egert,C., "Assessing the Success of Seminars on the Web", *Proceedings from the Association for the Advancement of Computing in Education*, Honolulu, HI, pp. 382-386, 1999

[5] Goldberg, M., "WebCT and First Year : Student reaction to and use of a web-based resource in the first year Computer Science", *Proceedings of the Conference of Integrating Technology into Computer Science Education*, Uppsala, Sweden, pp. 127-129, 1997

[6] Tucker, A., " A Summary of the ACM/IEEE-CS Joint Curriculum Task Force Report, Computing Curricula 1991", *Communications of the ACM*, 34(6), pp. 68-84, 1991

[7] Van Dam, A., http://www.cs.brown.edu/courses/cs015/