The Design of an Asynchronous Web-Based Project Review System to Support Studio-Based Learning in Computing Education

Anukrati Agrawal and Christopher D. Hundhausen
Visualization and End User Programming Laboratory
School of Electrical Engineering and Computer Science
Washington State University
Pullman, WA 99164−2752 USA
{aagrawal, hundhaus }@eecs.wsu.edu

Abstract

Learning computer science is no longer simply a matter of learning computer programming. Indeed, modern day computing jobs demand design, communication, and collaborative skills as well. In order to address this need and make computing education more engaging, motivating, and community-oriented, we have been exploring a "studio based" approach in which students (a) construct computational solutions to problems that have many possible solution strategies, and then (b) present their solutions to their peers and instructors for feedback and discussion. In the fall of 2007, we implemented this approach in a pre-CS 1 course by requiring students to present their solutions to five course programming projects in face-to-face review sessions. Interview, and observational data collected in this course pointed out several practical and logistical problems surrounding the face-to-face review sessions that diminished their educational effectiveness. To address these problems, we describe the preliminary design of a novel asynchronous web-based project review system to support and augment face-to-face studio-based review sessions. This system, which is being iteratively designed and implemented in 2008, will provide a foundation for future empirical research into the effectiveness of studio-based approaches in computing education.

1. Introduction

The studio-based instructional approach has been successfully used to teach design skills in architecture and arts education for over 100 years [1]. In this approach, students come together within so-called "design crits" to present the design of solutions to assigned problems that lend themselves to multiple solution strategies. Based on the feedback and discussion generated within these "design crits," students iteratively improve their solutions.

Over the past several years, Washington State University (WSU) has been exploring the potential for this approach in undergraduate computer science courses. Most recently, we used approach to teach the fall, 2007 offering of a pre-CS1 course at WSU. The course was structured around five programming projects of increasing difficulty. The students enrolled in the course were required to present their solutions to these projects to their lab sections (15-20 students plus a teaching assistant) within five of the regularly scheduled lab periods. To obtain feedback on these sessions, we held debrief sessions at the end of three of the sessions. In addition, we conducted semi-structured interviews with five students in order to obtain more comprehensive and formal feedback on the approach.

For the interview question "Did you receive helpful feedback from the written peer reviews of your work?", the following response was typical:

> "*I think that in peer reviews we (students) only worry about keeping everyone's feelings up more than telling the presenter what they need to do right and how*".

When asked, "Did you find the process of participating in "design crits" as an audience member helpful to you in learning computer programming?," one student gave the following response:

> *"It would have been a lot more helpful to get the code later on, and break it down and look at it, rather than just watching it once, and say[ing] - oh that was kind of cool"*.

In addition to these comments, students in the debrief sessions expressed a desire for the organization of the presentations to be improved. They offered two suggestions: (a) reduce the time wasted in switching laptops between the presentations, and (b) allow students to submit their assignments on-line, rather than during the presentation sessions.

Given this feedback, we wondered whether an online system might help in improving and adapting the studio based approach for computer science instruction. This led to the following research questions:

RQ1:	*How can we take advantage of an asynchronous online learning system to improve and tailor the studio based approach for teaching computer science?*

RQ2:	*What might the design of such a system look like?*

In this paper, we address these questions by (a) discussing the ways in which an online learning system can provide features to complement the traditional face-to-face studio based approach, and also tailor it for computer science, (b) presenting the preliminary design of an online learning system that is presently under construction, and (c) presenting our plans for evaluating the effectiveness of the system.

2. Background and Motivation

Since 2003, we have been exploring a studio-based approach to teaching a variety of computer science courses at WSU, including CS 1 [2] and an upper-division course on human-computer interaction design. Most recently, we implemented the approach in a pre-CS1 course entitled "Introduction to algorithmic problem solving." Taught using the Alice programming environment [3], the course is designed to be a gentle and motivational introduction to computer programming for students who have never programmed before.

In the fall, 2007 offering of the course, which enrolled 90 students, we implemented the studio-based approach by requiring all students to present, for feedback and discussion, their solutions to five progressively more difficult programming projects. Students gave their presentations in five of the regularly scheduled 2 hr. 50 min. lab periods. Due to time constraints, half of the 15 to 20 students in each lab section were required to give presentations on a given day, while the other half were required to write and post peer reviews of selected students' solutions using the Blackboard system [7].

We established clear requirements for both the presentations and peer reviews. In a 10-15 min. presentation, presenters were asked to "lead the class on a walk-through of their code," as well as to explain, reflect on, and justify their solutions. For example, in one of the assignments that involved writing a scavenger hunt game, presenters were asked to address questions like "How does your code support scoring?" and "What were the most difficult parts of the assignment to program, and why?" Following each presentation, the presenter's teaching assistant and peers were invited to provide verbal feedback and suggestions for improving the project.

Peer reviewers were asked to base their reviews on an established scoring rubric; however, rather than just providing checklists of requirements met and unmet, reviews had to "contain evidence of critical thinking and/or concrete suggestions for improvement."

We obtained student feedback on this approach and its implementation in two ways: (a) by holding informal debrief sessions at the end of three of the studio sessions and (b) by conducting in-depth semi-structured interviews with 5 students at the end of the semester.

Many of the students who contributed to these evaluations found the studio-based activities useful and interesting. At the same time, several key issues came up again and again:

- *Wasted time in studio*. Students observed that a substantial portion of the studio sessions (approximately 30 percent, according to our estimates), was dedicated to technical and bookkeeping endeavors that did not contribute to their learning, including (a) having to reconnect a student's laptop computer to the projector prior to each presentation; (b) having to reboot computers due to crashes and freezes during presentations; and (c) having to submit the source code for assignments to teaching assistants via USB drives.
- *Need for central repository*. Students commented that they wanted a central repository for all student work, so that they could access that work both before and after the studio sessions took place.
- *Real-time feedback difficult*. Students said that they found it difficult to provide constructive feedback on their peers' work within the real-time context of a studio session. They felt that they would be able to make more substantial and in depth comments and suggestions if they were afforded the opportunity to review their peers' work on their own time outside of class.
- *Written reviews preferred.* Student presenters expressed a preference to receive the reviews of their peers in written form instead of through verbal communication, as this would help them keep track of the feedback and allow them to take it into consideration in future assignments. However, the written reviews that were provided through Blackboard [7] tended to be too brief and superficial to be useful.

3. Preliminary System Design

The results just presented suggest that augmenting the traditional studio based approach with an asynchronous online learning system might greatly enhance the studio-based approach. We now turn to a discussion of what such a system might look like.

Our proposed system will provide a central repository for, and asynchronous access to, all students' solutions. This will allow students to read and respond any time and anywhere, which will give them more time to reflect and compose a response [4]. Moreover, the central repository will smooth out the presentation process

in the studio sessions, eliminating the need to connect a new machine to the projector prior to each presentation.

Student feedback submitted through the system will remain anonymous. This will help overcome some of the concerns that arise with traditional face-to-face studio sessions, like shyness or prejudiced reviews [5]. It may also embolden students who would otherwise be intimidated by speaking in front of their peers or instructor to express their views.

The on-line student reviews will enable instructors to assess what has been written, as well as enabling students to read what they and their peers have written as many times as they wish [6]. In addition, the online reviews will give the students an opportunity to improve their technical and presentation skills and learn beyond what was discussed in the class [1].

While general-purpose online learning systems such as Blackboard [7] support the above functionalities, our environment will be tailored specifically to support the kind of studio-based learning that we described in the previous section. To that end, the system will support four key features that are absent in general-purpose online learning systems:

1. *Support for submitting annotated computer programs*. To support studio-based learning in computer science courses, our system must allow students to talk about, review, and execute computer programs that meet requirements established in advance by a course instructor. Thus, our on-line system must support *artifact-centered discourse* [8], with computer programs being the primary objects to be discussed. Prior to submitting their solutions to a posted assignment through our system, students will be required to annotate their solutions with a set of annotation tags established by the assignment. Each pair of annotation tags will be used to mark a section of the solution that satisfies a given assignment requirement. Only assignments annotated with the required tags will be accepted by the system.
2. *Support for viewing and commenting upon assignments based on code section*. To root discussions and comments firmly in specific sections of code, our system will allow any user to view and comment upon a student's solution on a section-by-section basis, with the sections being defined by the required annotations described above.
3. *Support for linking specific blocks of code in one student's solution to blocks of code in other students' solutions*. Our system will encourage students to note and discuss areas of agreement and disagreement with other students by enabling them to link sections of their code with sections of other students' code. When specifying a link, a student must elaborate on what the link means. The following choices will be given by default: (a) You wrote this the way I did! (b) You wrote this totally differently from how I did!. In addition, free-form comments may be specified. Based on these links, our system will provide a global tree-based visualization that depicts the relative similarity of students' approaches to each code section.
4. *Support for executing co solutions directly from our system*. In order to avoid the need to switch presentation computers within studio sessions, our system will make it possible to execute, *within our on-line system*, any computer program written in a language for which a plug-in execution environment has been defined in our system. The ability to execute code from within our system will have the added benefit of allowing student reviewers to execute the code solutions they are reviewing without having to explicitly download the code to their own machines, which may provide students with more incentive to write thoughtful, in-depth reviews.

In sum, the above features support the studio-based instructional method in computer science courses more directly than a generic on-line learning system could.

4. Evaluation Plans

We believe that the integration of our on-line system with the traditional studio based approach will result in the following positive outcomes:

(a) *Students' increased participation in the review process*. Because reviews can be done anytime and anywhere, and because they can refer directly to code that is available within the on-line system (as opposed to requiring the user explicitly to download the code), we believe that students will be more motivated to take part in the on-line review process than they would in a face-to-face situation.

(b) *More reflective, deeper, more critical feedback*. Since feedback can be submitted anonymously, we believe it will be more honest, although we need to be careful to ensure that it remains constructive. In addition, since students will be able to review and execute the solutions they are reviewing through our system, we believe they will be willing to take more time to thoroughly consider the solutions they are reviewing than they otherwise would.

(c) *More organized and efficient studio sessions*. Since our system provides a common repository for all students' solutions, and since our system supports the execution of students' solutions, there will no longer be a need to switch presentation computers during the studio session, or to swap USB drives with student work in and out of a presentation com-

puter. As a result, we anticipate that the face-to-face studio sessions will be able to proceed more efficiently, with more time dedicated to educationally relevant activities and less time dedicated to administrative overhead.

(d) *A greater sense of community*. Because they will be able to explore and link to each other's code on-line, students will have more opportunities to build a sense of community around the programming activities in which they are engaging than they otherwise would.

(e) *Higher learning outcomes in assignments*. As mentioned above, we believe that our system will provide greater access to student assignments and encourage deeper, more critical reviews of student assignments. We anticipate that this will lead to the production of higher quality code than if no such online system were available.

We will employ a mixed-methods approach to evaluate our system's effectiveness in meeting the above outcomes. The basic idea will be to study the same studio-based course during two consecutive semesters. During the first semester, we will implement the course as a traditional studio-based course with face-to-face studio sessions only; the on-line system will not be available. In the second semester, we will implement the course in the same way, except that we will require the use of our on-line system. In both courses, we will collect the following data to address the above outcomes:

(a) Number of student contributions to the review process, both on-line and face-to-face.

(b) Quality and depth of student contributions, according to a rating system whose reliability will be established.

(c) Percentage of time within face-to-face studio sessions dedicated to educationally relevant vs. irrelevant activities, as determined through a post-hoc video analysis of review sessions.

(d) Pre- and post-surveys that include sense-of-community measures, as well as exit interviews administered to a small sample of students.

(e) Assignment scores, broken down by requirement.

5. Summary and Future Work

Our preliminary exploration of the studio-based approach to teaching computer science suggests several problems with traditional face-to-face reviews, including difficulties with providing thorough, honest reviews within the real-time context of a face-to-face review, and a tendency to waste time with irrelevant administrative tasks like switching computers and transferring files. To address these issues, we have presented the preliminary design of an on-line, asynchronous learning system specifically tailored for computer science courses. Our system goes beyond generic on-line learning systems by (a) supporting artifact-centered discourse on annotated computer programs, (b) allowing students to build areas of agreement and disagreement by having them link specific sections of their together, and (c) supporting the direct execution of code solutions from within the system, thereby avoiding the need to explicitly download and execute code.

In the upcoming year, we plan to implement this system using ASP .NET and SQL, and to conduct formal classroom studies not only to evaluate the effectiveness of the studio-based approach, but also to evaluate the ability of our system to enhance the studio-based approach to teaching computer science courses.

6. Acknowledgments

This project is funded by a National Science Foundation CPATH Award (CNS-0721927). Contributions to this work by other members of the research team—N. Hari Narayanan, Martha Crosby, Margaret Ross, Michael Trevisan and Rita Vick—are gratefully acknowledged.

7. References

[1] K. Lynch, A. Carbone, D. Arnott, and P. Jamieson, "A studio-based approach to teaching information technology," in *Proceedings of the Seventh World Conference on Computers in Education*. ACM Press, New York, 2002, pp. 75-79.

[2] C. D. Hundhausen and J. L. Brown, "Designing, visualizing, and discussing algorithms within a CS 1 studio experience: an empirical study," *Computers & Education 50* (1), 2008, pp. 301-326.

[3] W. Dann, C. Cooper, and R. Pausch, *Learning to Program with Alice*. Prentice-Hall, Upper Saddle River, NJ, 2006.

[4] M. A. Freeman and J. M. Capper, "An anonymous asynchronous web-based role play," in *Proc. ASCILITE '98*. Australasian Society for Computers in Learning in Tertiary Education, Wollongong, 1998, pp. 251-260.

[5] J. M. Wiecha, R. Gramling, P. Joachim, and H. Vanderschmidt, "Collaborative e-Learning using streaming video and asynchronous discussion boards to teach the cognitive foundation of medical interviewing: a case study," *Journal of Medical Internet Research 5* (2), 2003,

[6] A. Black, "The use of asynchronous discussion: creating a text of talk," *Contemporary Issues in Technology and Teacher Education 5* (1), 2005, pp. 5-24.

[7] Blackboard, Inc., http://www.blackboard.com/us/index.bbaccessed 28 Feb. 2008.

[8] D. Suthers and J. Xu, "Kukakuka: an online environment for artifact-centered discourse," in *Proc. Eleventh World Wide Web Conference*. WWW2002, Honolulu, 2002, pp. 472-480.