A Snapshot of Studio Based Learning

Code Reviews as a Means of Community Building

Joseph Bergin
Pace University
New York, New York USA
jbergin@pace.edu

Rick Mercer
University of Arizona
Tucson, Arizona USA
mercer@cs.arizona.edu

David West
College of Santa Fe
Las Vegas, New Mexico USA
ProfWest@fastmail.fm

Robert C. Duvall
Duke University
Durham, North Carolina USA
rcd@cs.duke.edu

Eugene Wallingford
University of Northern Iowa
Cedar Falls, Iowa USA
wallingf@cs.uni.edu

Pamela M. Rostal
Perficient, Inc.
Minneapolis, Minnesota USA
pam.rostal@perficient.com

Richard P. Gabriel
IBM Research
Hawthorne, New York USA
rpg@dreamsongs.com, rpg@us.ibm.com

Abstract

Studio Based Learning is an educational process that has found more success in the humanities than the sciences. In these disciplines most learning is done in the studio, with apprentices and journeymen working at the elbow of a practicing master. When apprentices join a studio, their education progresses from the point of their current knowledge through journeyman status while working on real projects that become part of a lasting portfolio. Student work is subject to constant review by both peers and mentors as a means of providing valuable feedback and to solidify the shared sense of community. The Studio Based Learning presented in this session demonstrates the possibility of using the approach to advance computer science education at the university and begin to establish the community of practice that will improve the profession beyond university walls.

This Collaborative Activity Session will show one aspect of this approach in the context of a real course, by re-casting a typical Code Review as a Studio Review using principles from Writers' Workshops and the Touchstones Discussion Project. Using code provided by Educators' Symposium participants, we will show how a typically uncomfortable activity can be turned into a positive, enriching experience. By making space to discuss student concerns about the code they write, we hope to engage students better and to build mutual respect within the community. After asking participants to experience a constructive small group discussion, we will engage in a larger discussion of how to use these techniques throughout the curriculum.

Categories and Subject Descriptors D.m.1 [**Software**]: Miscellaneous – software *psychology*.

General Terms Design, Human Factors.

OOPSLA'08, October 19–23, 2008, Nashville, Tennessee, USA.
ACM 978-1-60558-220-7/08/10.

Keywords teaching and learning techniques; facilitating active learning; learner-centered education

1. Overview of the Proposed Session

Before coming to OOPSLA, participants will be asked to submit a program in a language of their choice, using only standard libraries, that implements a program that act on a data set of words. The program should read white-space delimited words from a specified file, treating them in a case-insensitive manner, and print an ordered list of the top 100 unique words in the file. The output should print one word per line, consisting of the number of times the word occurs in the file, a tab character, and the word itself. This specification provides the core implementation of a Tag Cloud, a popular technique for determining the important concepts within a set of documents. We will make a detailed specification and test data available via the web well before OOPSLA so that the results can be collected and brought to the conference.

These programs will be printed poster size, much like displayed artwork, and placed around the room. They will be the basis of discussion among groups of four to six participants each. Before breaking into groups, participants will be given guidelines for the discussion based on those from Writers' Workshops and the Touchstones Discussion Project. Among these are guidelines such as "constructive comments only", "something positive must be said", and "the programmer is not permitted to explain his work during discussion". Groups will then proceed to discuss all aspects of the displayed code for about 20 minutes.

When the groups have finished their discussion, they will be asked to share their impressions. This retrospective will focus on the effectiveness of the discussion, not on the specifics of the code. Principles from Writers' Workshops and the Touchstones Discussion Project will again be the basis of the retrospective discussion, with a focus on issues such as how effectively the groups communicated, to what extent each person felt respected during the discussion, and whether the feedback was reasonable and helpful. Pedagogical issues such as "how effective was this activity as a code review" and "how effective was this activity at improving students' coding skills" will also be explicitly discussed.

2. Outline of the Proposed Session

The session will be structured as a small play wrapped around a Code Review to provide a context in which such an activity might take place.

Act I (10 minutes)

A few students, played by the session presenters, are writing code in a studio setting. The room has tables, with computers, whiteboards, flipcharts, books, etc. – the typical workspace of an Agile team. The students are working on a Tag Cloud exercise such as the coding challenge that Owen Astrachan issued to the SIGCSE community in the spring of 2007. The students are working in pairs, but also frequently communicating across pairs. Books are consulted; progress is marked on the flipcharts; questions are asked and answered. Eventually, the coding comes to an end, and version 1.0 is ready for review. In reality, the symposium participants would have previously written the code produced in the opening act.

Intermezzo (40 minutes)

The Code Review described above will now take place with all symposium participants in groups of four to six. The session presenters will facilitate each group's discussion.

Act II (10 minutes)

After the activity, the play will continue by connecting the Code Review to a deeper discussion of how this problem and solution technique, as well as the discussion technique itself, fits into a larger cultural context. For example, "How does a Touchstones discussion relate to and differ from Platonic discourse?"

Denouement (30 minutes)

The play will be followed by a retrospective including all symposium participants. This discussion will be driven by participant questions and guided by the presenters. It is intended to cover practical issues relating to how we might integrate Studio Based Learning techniques into traditional curricula as an agent of more significant potential changes. Specifically, participants will be asked to consider how much of the play could be used in their class, or is being used already. Presenters have broad experience in using this and similar techniques as well as experience in fitting this into the broader context of Studio Based Learning.

3. Background on Studio Based Learning

A pilot program at New Mexico Highlands University eliminated courses in favor of discrete competencies that all students had to satisfy prior to graduation. A "one-room schoolhouse" pattern was followed, with students spending most of the day in the studio, leaving to attend other courses, returning to work, socialize, and even discuss what was learned in other courses. Students created "individual learning plans" every eight weeks, identifying which competencies they would be working on. These plans then defined the material that the mentor/faculty needed to prepare and deliver – on demand, in discrete and focused units – during the term. An apprenticeship experience, students working on work for real world, paying, customers was incorporated in the program and both the learning and work components were managed consistent with typical agile practices; e.g., weekly planning sessions, daily stand-ups, pair learning/pair programming, retrospectives, etc.

In the fall of 2007, an existing computer science degree program at the College of Santa Fe was modified and a new software design degree was introduced, both incorporating as much of the Highlands model as possible. Although both programs have standard courses, 40% of both cores are software studios that are modeled after the one-room schoolhouse experience at Highlands. Non-core requirements for the software design degree include performing art, business, philosophy, and effective communication. The studios are designed to be accessible for students from other disciplines, both to build a broader community of learners and as means for integrating liberal arts subject matter into the program.

Studio Based Learning, whether adopted in whole or in part, has potential to change the relationship of students to instructors, each other, and the university. It presents a more fulfilling, humane, educational experience for the student and so may have implications for issues of broader interest such as student recruitment, retention, and long-term satisfaction with their educational experience.

References

[1] John Herren, *Tag Clouds*, http://www.tagcloud.com/

[2] Owen Astrachan, *SIGCSE 2006 Challenge Exercise*, http://www.cs.duke.edu/csed/code/

[3] *Code Walkthrough Procedure from Forecast Systems Laboratory,* http://www-md.fsl.noaa.gov/eft/developer/CodeWalkthroughGuidelines.html

[4] *Touchstones Discussion Project*, http://www.touchstones.org/

[5] James O. Coplien, Bobby Wolfe, *A Pattern Language for Writers' Workshops*, Pattern Languages of Program Design 4, 1999