Synchronous Cloud Instruction: A Model to Improve Students' Learning Experience

Completed Research

Michael Whitney
Winthrop University
whitneym@winthrop.edu

Abstract

This paper describes an instructional model based on the application of Cloud9 which is a cloud-based Integrated Development Environment (IDE). Cloud9's synchronous code sharing capabilities are leveraged to further engage Information Systems students in the learning process in an introductory web development course. This model actively engages students in a synchronized collaborative code writing process during lectures and labs where student work and contributions are incorporated into the presentations, group discussions, and individual work. The contribution of this work is showing how synchronized cloud IDEs can be leveraged to create and enhance an active learning experience in the form of pair programming, collaborative code writing lectures and labs, and cooperative code reviews. Reported data includes student perceptions and faculty impressions as captured through group discussions, interviews and surveys.

Keywords

Pedagogy; Active Learning; Collaborative Learning; Cloud IDE.

Introduction

A common core course for Information Systems programs is Introduction to Web Design. In such a course, students typically develop basic HTML, CSS, and JavaScript knowledge, skills, and abilities. As with many introductory courses, teaching this course can be a challenge due to large class sizes, the amount of materials, and a lack of engagement of students in large classes (Hornsby & Osman, 2014; Kivunja, 2015; Lynch & Pappas, 2017). Research has shown that the lecture-only approach might not be enough to appropriately engage students in the learning process (Burch, et al. 2015). To this end, many instructors attempt to overcome the engagement challenge with active learning approaches such as interactive lectures, individual and group learning activities, and comprehensive feedback on assignments and activities.

This paper reports on a novel approach that builds upon common active learning principles and engagement methods used in an introductory web design course. More specifically, this paper presents how Cloud9, a cloud-based Integrated Development Environment (IDE), can be used to enhance the learning process in an introductory web design course. Results of a multi-semester implementation of this approach indicate that Cloud9's synchronous code sharing capabilities can be leveraged to further engage Information Systems students in the learning process. This is accomplished by actively engaging students in a synchronized collaborative code writing process during lectures and labs by integrating student work and contributions into the lectures, presentations, group discussions, and individual work. The contribution of this work shows how synchronized cloud IDEs can be leveraged to create and enhance an active learning experience in the form of collaborative code writing lectures and labs, pair programming, and cooperative code reviews.

Leveraging an Online IDE to Facilitate Active Learning

Active learning is a common instructional approach used by many educators to help engage students in a learning-centered process that can enhance achievement levels, content mastery, and higher order thinking (Freeman, et al. 2014; Yoder et al. 2005). For IS Web based programs, successful engagement approaches include class discussions, problem solving, code review, debugging sessions, and peer programming to name a few. Research shows that these approaches generally result in reduced frustration, increased confidence, and increased retention (Mani et al. 2014; Williams et al. 2003).

At the core of creating an active learning environment is the use of an appropriate pedagogy which will afford the instructor the opportunity to transfer targeted knowledge, skills and competencies to students. Research shows that the key instructional approaches that facilitate knowledge transfer include active and engaged learning, in-class communication, collaboration and teamwork, focused discussions, group learning and, instructor feedback (Deslauriers, et al. 2011; Kirschner, et al. 2011; Nokes-Malach et al. 2015).

When creating an active learning environment, instructors have many resources to choose from. While paper and pencil resources are effective, technology brings opportunities for engagement to a whole new level. For IS programs, the advent of sharable cloud-based IDEs has given instructors the opportunity to actively engage students in the learning process. More so, with a cloud-based IDE, students can contribute to lecture materials in real-time, collaborate in pair-programming activities (locally and remotely), partake in code reviews, and receive faculty coding support all with a simple internet connection and a stable browser. But, as will be discussed, the successful integration of active learning with a cloud-based IDE is dependent on multiple factors.

At a small south-eastern public university, researchers implemented Cloud9 in a freshman level Introduction to Web Design course for a period of four semesters. Over this period, a model was developed in an iterative fashion based on gathered data. The result of which is a set of cloud-based IDE practices that aid in then creation and enhancement of an active learning environment. These practices include collaborative coding lectures, synchronous pair programming, synchronous coding support, collaborative code reviews, code monitoring, and code history review.

Cloud9 (2019) was chosen as the online IDE for this work. Cloud9 supports multiple languages (e.g., HTML, CSS, JavaScript, Python, PHP, etc). Developed by Amazon Web Services, this free IDE allows individuals and groups to synchronously write, run, and debug code. Cloud9 has a built-in editor, a live html viewer, and server which allows students to develop and view their work in real-time (Figure 1). Cloud9 is free for basic use, but it does require a credit card to create an account. For a minimal fee ($1.00 per month) a faculty member can create a team workspace for those students who do not have a credit card or do not want to share their credit card information.

Cloud IDE Instruction: A Model for Engagement

The following section is used to present an effective approach to leveraging the benefits of a cloud IDE to support an active learning environment in a Web Design lecture/lab-based course. This active learning environment will include a cloud IDE that supports collaborative lectures, paired programming, collaborative code review, synchronous faculty code support, and code monitoring.

Collaborative Coding Lectures

Historically, a common lecturing approach to Web instruction is to write the code in a live session while describing the process and underlying concepts. By doing so, students witness the creation of a workable example rather than being shown a pre-coded PowerPoint slide. Within the live coding lecture, students can be actively engaged in the process by requiring them to provide input during key points of the code creation (e.g., ask them what something means or what the next step would be). One limitation of such an approach is the requirement that the instructor primarily plays the role of the driver and, therefore, writes all the code.

With Cloud9 (Figure 1), a single workspace can be shared amongst the class and thusly, provide the opportunity for each student to interact with and contribute to the live coding lecture process. During collaborative live coding lectures, the shared workspace is projected in front of the class and is available on each student's computer. Within this environment, researchers found two effective means for engaging students with the collaborative live coding lecture. The first is to choose one student and have them continue the lecturer's code writing process. As the student codes, the lecturer facilitates a group discussion based on the code being written. This discussion is organic and can be led in many directions depending on group knowledge and needs. Researchers have learned to provide simpler coding tasks early in the semester to ease students into the process and then more challenging tasks that deepen group discussions.

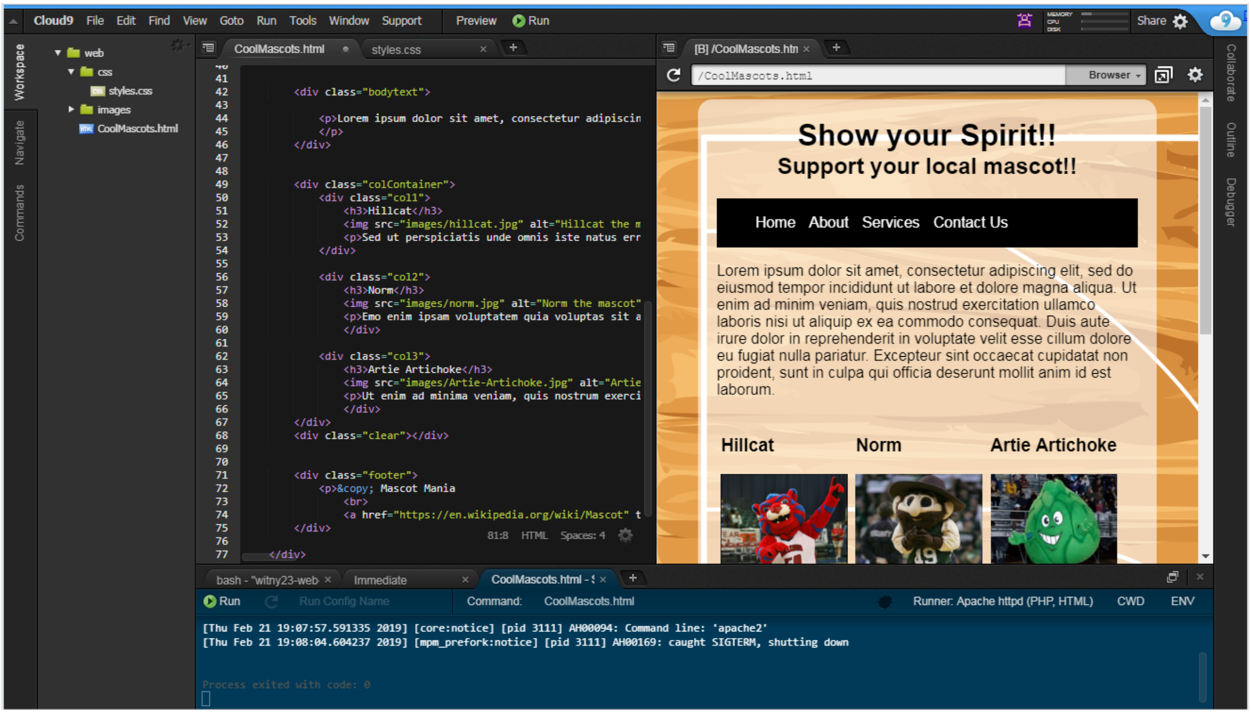

Figure 1. Cloud9 IDE.

Students found these interactive lectures to be engaging and a benefit to their learning process. Having access to the live code also proved to be beneficial. Student comments include:

"It was easier to follow along on your own computer instead of watching on a faraway screen"

"the process inspires an active think-tank that would not exist otherwise"

"I enjoyed being able to contribute to the process and how the practical explanations can be demonstrated easily and live"

A second effective means for engaging students in a live coding lecture is to create a collaborative project where small teams (2-3) are responsible for live coding a portion of the overall project. The process starts with an overview lecture of the project and underlying concepts. Once complete, teams are created, provided coding tasks, and then given time to discuss how they are going to approach to their provided task. The collaborative live coding lecture then proceeds with each group sequentially taking turns their portion of the lecture code. During this time, groups describe the code they are contributing which, at times, can initiate discussions.

Faculty have found the collaborative live coding lectures to be an effective means to further engage students in the learning process. Like most classes that promote student interaction during lectures, collaborative live coding is somewhat reliant on student preparation. If only a small percentage of students come prepared for class (e.g., read the assigned materials) then the process suffers. Instructors found that online quizzes, participation points, and individual discussions help alleviate this issue

Collaborative Coding Labs

Course labs provide students the opportunity to develop Web coding practices and abilities based on classroom knowledge gained and lessons learned. The common instructional approach to labs has students working individually or in pairs whilst the instructor is present to help facilitate experiential learning by further explaining topics and diagnosing problems. Typically, the instructor jumps from one group to another to either provide instruction or praise excellent work. To examine student work, it is necessary for the instructor to look over the student's shoulder or have the student move so the instructor can take control of the keyboard and mouse. With Cloud9, the instructor can remotely monitor workspaces, student teams can collaborate on more than one computer, live faculty support is not restricted by location, and group collaboration can easily move from a pair of students to the whole class. The following is a description of how such collaboration can be utilized in the lab with Cloud9.

Synchronous Paired Programming

Pair programming was developed by industry and capitalized on by academia as a learning method. Pair programming has two programmers simultaneously develop a program on the same machine. The first programmer, known as the driver, controls the entry of the code while the second programmer (a.k.a. navigator) monitors the coding process for potential issues. While the driver is focused on the design of the code, the navigator offers suggestions on problem resolution, and potential big picture issues. The team commonly discuss issues and regularly switch roles. Researchers have found that paired programming promotes active and collaborative learning, a reduction in student frustration, increase in confidence (Mani et al. 2014; Williams et al. 2013).

The common instructional approach to pair programming does have its limitations. Two of which include a physical space issue and a scheduling/location issue for work outside of required lab times. The physical space issue comes about when two students (sometimes three) must crowd around a single small monitor. The effect of which is a navigator who must enter the driver's space to observe the code writing process. This continuous space sharing / invasion can be uncomfortable and cause a disconnect from the process by the navigator. By sharing a synchronized workspace on two different computers, the space issue is relieved. In addition, the navigator can further contribute to the learning process by using their computer to research difficult coding concepts. Comments supporting this setup include:

"Working synchronously on our own computers was better because we can work together without getting in each other's way"

"We were more productive than if we were working on the same computer"

In addition to the physical space issue, many pair programming lab activities and assignments are not completed during a scheduled lab time. To this end, students must schedule a time and place to come together and complete their work. Such a requirement can be a burden, especially for non-traditional students. As Cloud9 is a synchronous workspace, the location burden is relieved with the help of a simple form of communication (e.g., Skype, cell phone, etc.). To expand, both students can log into their Cloud9 workspace from any location, start a Skype call, and continue their driver/navigator roles as needed. Non-traditional students have voiced their appreciation of being able to remotely work with their partners on given peer programming assignments. One student said about remote peer programming:

"not having to drive back to campus was a big plus because I would have had to take time off work"

Student Pairs

Having pair programming teams collaborate from two computers (either locally or remotely) can inadvertently contribute to a shift in roles from a driver / navigator combination to a leader / follower combination. To discover why this happens, researchers tracked group dynamics and classroom performance using course grades and peer evaluations. Out of this data, the researchers identified two indicators for the role shift.

The first indicator for the role shift is approximately a 20% difference in course performance between the team members as measured by quiz/midterm scores. Under this condition, the higher performing student takes on the role of leader with their partner becoming passive in the programming process. However, when the performance scores are closer to each other, the expected level of driver / navigator engagement does occur.

The second indicator for the role shift was the lack of preparation as reported by team members. More so, students reported that working with an unprepared partner hinders their learning process as they feel as if they become the instructor rather than the collaborator. Alternating the labs between individual activities and pair activities help to overcome this tension. The individual activities require students to build a knowledge base that will be used in the next pair programming lab. In addition, requiring students to take an online quiz prior to pair programming labs helps ensure they are better prepared to contribute to the pair learning experience.

Faculty report that by alternating individual and paired programming labs combined with online quizzes resulted in a more interactive and engaged experience for the students. In addition, the alternation and quiz preparation combination ensured that students develop their own knowledge base needed in the rest of the curriculum rather than relying on a partner to acquire a passing grade.

Faculty Code Support: Live Code Review

It is expected that students will make mistakes, hit brick walls, and experience breakthroughs as they develop their coding skills and abilities. With a shared Cloud9 workplace, it is possible to help students break through their brick walls from a centralized location (e.g., podium computer). An effective method to initiate the faculty support process, is to require students to document (with comments) what they believe the barrier to be, where they believe the barrier exists in their code, and what they have attempted to overcome the barrier. Once the request for support is submitted (e.g., raising hand or email outside of class), students can continue to work on their code as a live code review is conducted by the instructor.

The process of providing live code support has multiple advantages. First, students can continue evaluating their own code. Second, once the faculty has identified the issue, they can provide simple guidance with code comments, or discuss the issue with the team depending on the needed instructional support.

This form of live support is also effective outside of the lab. Once a time is established, faculty members can remotely open the student's workspace, open a Skype session, and discuss with the students the issues the students are experiencing within their program. Students have reported an appreciation for the support because it helps them understand the issue in a timely fashion and allows them to continue coding rather than having to wait until they were back on campus.

Collaborative Code Review

The code review is an important process that helps identify coding mistakes and potential issues. The process is typically a standalone lecture component that does not play a key role in a web development course. Arguably, the code review process can further engage students in the code learning process. With Cloud9 a code review can easily be collaboratively performed by the class on any team's code by projecting their workspace at the front of the lab and stepping through their code as a group. Faculty have found that the best time to incorporate a collaborative review is when they have discovered a novel coding problem that a team has introduced in their code. By doing so, the class becomes engaged with the process because it directly benefits their own work and, ultimately, their overall understanding of the underlying coding concepts.

Code Monitoring and Development History

Requiring students to write each line of code (no copy / paste) engages students in the code development process. Observing a student's code writing process provides insight into their train of thought. The historical method of code writing observation was to look over the student's shoulder which can create an unwelcomed pressure on the student. With Cloud9, a faculty member can observe the code writing process from the podium. In addition, a faculty member can easily track the progress of all lab projects simply by navigating from one Cloud9 workspace to another.

In addition to facilitating code monitoring, Cloud9 also has a version history (Figure 2) that makes it easy to observe the coding process and roll back to an earlier version. More so, a document's history can be viewed as a video with the play button, stepped forward or backward with the revision buttons, or moved through with the slider. The history feature can be quite useful beyond rolling code back to a previous version. First, it can complement the code review process by affording the collaborators the opportunity to observe the code writing process. Second, the history feature can keep students honest as code that is copy / pasted can easily be identified.

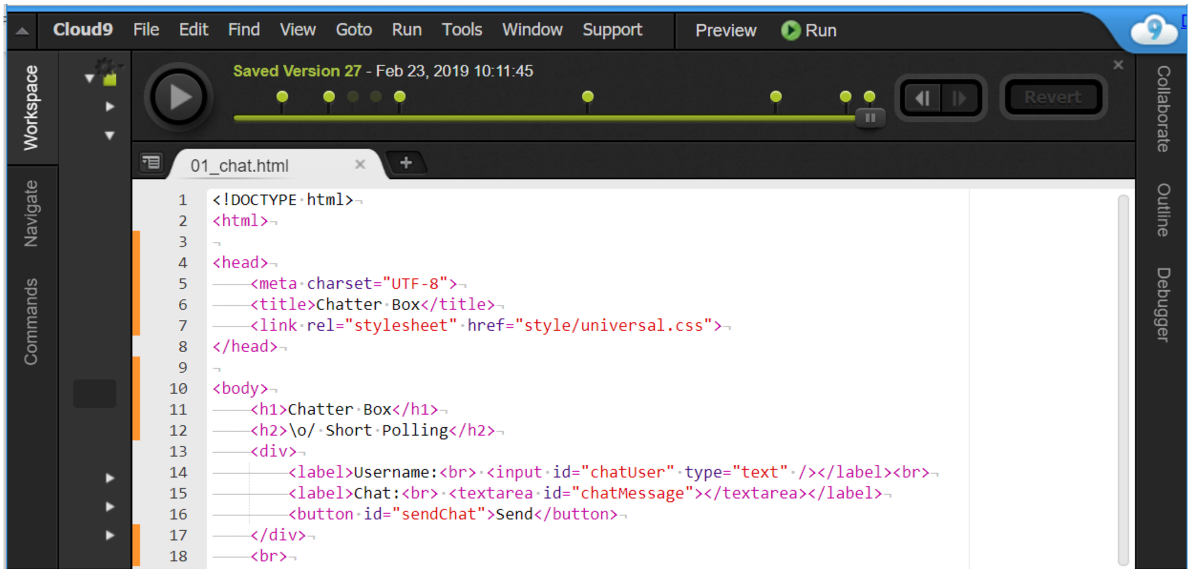

Figure 2. Development Version History

Model Development

The development of the Synchronous Cloud Instruction Model was motivated by the identification of a means to enhance the learning process in an introductory web design course. Over the course of two years, the Synchronous Cloud Instruction Model was developed in an iterative fashion based on gathered data. The following describes development process and related findings.

Methodology

The Synchronous Cloud Instruction Model was researched and developed over the period of two years at a small south-eastern public university with an enrollment of 6,000 students and a college enrollment of 900 students. The freshman level Introduction to Web Design course served as the foundation for the development of the Synchronous Cloud Instruction Model. The course was taught by the same instructor over the period of four semesters and averaged approximately 15 students per semester. In total, 58 students agreed to participate with at least one part of the data collection process (surveys, group discussions, interviews). The instructor worked directly with researchers on the development and deployment of the Synchronous Cloud Instruction Model.

Data Collection

To evaluate student perceptions of the Synchronous Cloud Instruction Model, surveys were administered at the end of each semester for all sections. The surveys consisted of ten Likert scaled questions and seven open ended questions. Data gathered from the surveys consisted of perceptions on collaborative coding lectures, synchronous pair programming, synchronous coding support, and collaborative code reviews (Table 1).

In addition to surveys, researchers also conducted group discussions and student interviews throughout the development of the model. Each class experienced multiple group discussions at the end of labs and several students agreed to share their thoughts on their learning experience during interviews. Data gathered during group discussions and interviews was based on satisfaction and recommendations related to collaborative coding lectures, synchronous pair programming, synchronous coding support, and collaborative code reviews.

	Semester 1	Semester 2	Semester 3	Semester 4
Collaborative coding lectures	4.6	4.3	4.1	4.2
Synchronous pair programming	4.8	4.5	4.4	4.2
Synchronous coding support	4.7	4.4	4.1	4.3
Collaborative code reviews	4.6	4.2	4.1	4.4
Notes - Items measured on a 5-point Likert Scale from Disagree (1) to Agree (5)				

Table 1. Synchronous Cloud Instruction Model Survey Data

Discussion

After four semesters, the feedback and critiques (surveys, group discussions, interviews) proved to be positive and constructive. More so, the data was used to refine the Synchronous Cloud Instruction Model in an iterative fashion. Importantly, the qualitative data gathered heavily influenced the development process while the quantitative Likert style data provides a positive indication that the most recent iteration is perceived as an improvement on previous versions (Table 1).

Throughout this paper, individual components of the Synchronous Cloud Instruction Model (with supporting data) have been presented. A summary of the proposed model can be found in Table 2. Future research includes the expansion into other courses and an evaluation of the effects of synchronous cloud instruction on performance.

Approach	**Description**	**Benefits**
Collaborative coding lectures	Integrate live student code into the lecture process.	Students become part of lecture by reviewing and contributing code.
Synchronous paired programming	Real-time collaboration on separate computers.	Remove personal space and location limitations.
Faculty code support	Synchronously review student code from remote computer.	Interactive feedback on live code.
Collaborative code review	Push out individual code to the class for a code review	Engage students in the review process.
Code monitoring and development history	Remotely observe the writing process and view history.	Gain insight into student code writing thought process.

Table 2. Synchronous Cloud Instruction Model Summary

Conclusion

For many introductory web courses, it is far too easy for students to become passive learners. More so, the podium styled lecture followed up with a step-by-step lab lacks the level of engagement needed to promote active learning of course materials. The objective of this paper is enhance the historic instructional model by leveraging the potential of synchronous cloud programming into an active learning environment. To this end, this paper contributes to research on how IS student engagement can be increased in an introductory Web design course. It does so by proposing a model that is built upon concrete methods for inclusion of learning interactions in the introductory Web design course. In addition, these methods do not require a complete rebuild of any course. More so, there is no need to "flip" the class or re-build a course from the ground up. All the methods can be infused into the course with minimal efforts.

Based on experience with utilizing Cloud9 to facilitate an active learning environment over the past two years, the researchers believe strongly in the use of a cloud-based IDE in introductory Web design courses. Over this period, the lessons learned and knowledge gained have helped improve the approach to the development of an active learning environment. Our hopes of sharing our lessons learned and knowledge gained is that other educators might also benefit from moving their IDEs into the cloud.

REFERENCES

Burch, G. F., Heller, N. A., Burch, J. A., Freed, R. and Steed, S. A. 2015. "Student Engagement: Developing a Conceptual Framework and Survey Instrument," *Journal of Education for Business*, (90:4), pp. 224-229, DOI: 10.1080/08832323.2015.1019821.

Cloud9 2019. Amazon Web Services Cloud9. Retrieved from: https://aws.amazon.com/cloud9/

Deslauriers, L, Schelew E. and Wieman, C. 2011. "Improved learning in a large-enrollment physics class," *Science*, 332, pp. 862-864.

Freeman, S., Eddy, S., McDonough, M., Smith, M., Okoroafor, N., Jordt, H., & Wenderoth, M. 2014. "Active learning increases student performance in science, engineering, and mathematics," *Proceedings of the National Academy of Scientists*, 111(23), pp. 8410–8415. https://dx.doi.org/10.1073/pnas.1319030111

Hornsby, D. J., & Osman, R. 2014. "Massification in higher education: large classes and student learning," *Higher Education, 67*(6), pp. 711-719.

Kirschner, F., Paas, F., Kirschner, P. A., & Janssen, J. 2011. "Differential effects of problem solving demands on individual and collaborative learning outcomes," *Learning and Instruction*, 21(4), pp. 587–599.

Kivunja, C. 2015. "Innovative methodologies for 21st century learning, teaching and assessment: A convenience sampling investigation into the use of social media technologies in higher education," *International Journal of Higher Education, 4*(2), pp. 1-26.

Lynch, R. P., & Pappas, E. 2017. "A Model for Teaching Large Classes: Facilitating a "Small Class Feel"," *International Journal of Higher Education, 6*(2), 199-212.

Mani, M., Alkabour, N., & Alao, D. 2014. "Evaluating effectiveness of active learning in computer science using metacognition," In *Frontiers in Education Conference (FIE), 2014 IEEE*. pp. 1-8.

Nokes-Malach, T. J., Richey, J. E., & Gadgil, S. 2015. "When Is It Better to Learn Together? Insights from Research on Collaborative Learning," *Educational Psychology Review*, 27(4), pp. 645-656.

Williams, L., McDowell, C., Nagappan, N., Fernald, J., & Werner, L. 2003. "Building Pair Programming Knowledge Through a Family of Experiments," *In Proceedings of the 2003 International Symposium on Empirical Software Engineering (ISESE '03). IEEE Computer Society*, Washington, DC, USA, pp. 143-152.

Yoder, J. D., & Hochevar, C. M. 2005. "Encouraging active learning can improve students' performance on examinations," *Teaching of Psychology*, 32(2), pp. 91-95.