Satori: CS 2150 Course Tools Rebuild
(Technical Paper)

Course Management Tools
(STS Paper)

A Thesis Prospectus Submitted to the

Faculty of the School of Engineering and Applied Science
University of Virginia • Charlottesville, Virginia

In Partial Fulfillment of the Requirements for the Degree
Bachelor of Science, School of Engineering

Austin Sullivan
Fall, 2019

Technical Project Team Members:

Disha Jain
Andrew Lewis
Winston Liu

On my honor as a University Student, I have neither given nor received unauthorized aid on this assignment as defined by the Honor Guidelines for Thesis-Related Assignments

Signed: _____ Date _____
Austin Sullivan

Approved: _____ Date _____
S. Travis Elliott, Department of Engineering and Society

Approved: _____ Date _____
Aaron Bloomfield, Department of Computer Science

Introduction

In a book review of *Teaching with Technology: Creating Student-Centered Classrooms (*Sandholtz, J.H., Ringstaff, C., & Dwyer, D.C. 1997), a book based on a study "investigat[ing] how the routine use of technology by teachers and students would affect teaching and learning", Jennifer Handley states plainly that "[t]echnology has the potential to change education in beneficial ways, but only under certain circumstances... the operating principle of the study was not for the teachers to use technology all the time, but to use the tool that best supported the learning objective."

Technology is expanding the concept of what a classroom can be. My STS research paper will use the Actor-Network Theory framework to explore how technology has the potential to meaningfully enhance the student experience. Online course tools can provide students with better resources and faster feedback, allow more students to enroll in classes by enabling larger class sizes, and decrease the amount of time instructors spend on menial tasks, ultimately allowing them to help students in more direct ways. However, when not implemented well or used correctly, they also have the potential to detract from the course.

In order to address these concerns, my technical project is tackling a particular set of course tools that are providing a sub-optimal experience for both students and the course staff. The current course management system is clunky and occasionally suffers from performance issues during busy office hours, hindering the teaching assistants' ability to help students. An improved set of course tools would yield benefits to instructors, teaching assistants, and students alike. Our design will focus primarily on offering better feedback, primarily through automated grading, giving more reliable queue performance, and improving the user experience.

Technical Problem

Most modern university classes supplement traditional lectures and labs with a course tools website through which a great deal of a student's interactions with the course occur. CS 2150 - Program and Data Representation is a required class for all UVA undergraduate computer science majors and minors. Many functions of the course are run through its course tools website, including assignment submissions, regrades, and an office hours queue. The current website, stitched together over time and amended to achieve basic minimal functionality as required, leaves much to be desired.

As a teaching assistant for CS 2150, I have seen firsthand the impact the course tools website has on the course. The current system is functional but slow and often times unreliable. There is very little feedback to the student upon assignment submission. The office hours queue will often crash or slow down significantly during crowded office hours, exacerbating the busyness of office hours by inhibiting the ability of the TAs to help students. Additionally, the procedures for grading assignments are tedious and clunky; an inefficient use of the time of teaching assistants that would be much better spent in more direct and impactful ways. The system also fails to provide meaningful feedback to students upon assignment submission. As a whole, the system does not meet the requirements of, as well provides a poor user experience for both the students and the course staff.

To address these issues, my team is rebuilding the course tools website in its entirety. Our goal is to design a submission service equipped with auto-grading capabilities, providing students with feedback relatively quickly and reducing the time that instructors and TAs need to spend grading assignments. Furthermore, the queue will be revamped to better handle busy office hours as well as provide additional functionality that has been suggested by current TAs. A more intuitive and inviting user interface will also be designed .Overall, the development of

features to address user needs and the redesign of the user interface will benefit both students and instructors.

Since three of the four members of the team are CS 2150 TAs, we will have direct access to each of the main stakeholders of this system: the students, teaching assistants, and instructors. A minimum viable product will be developed by the start of next semester, allowing us to test our system in real-time with next semester's CS 2150 course. We will incorporate feedback from each of the main stakeholders, as well as observations from our own experiences, into the product.

STS Problem

According to Wikipedia, "[a] learning management system (LMS) is a software application for the administration, documentation, tracking, reporting, and delivery of educational courses, training programs, or learning and development programs" (Learning management system, 2019). The first software LMS was built in the 1990s, and today there are several LMSs offering a wide range of services. Online learning tools are a standard part of the modern classroom. Canvas, the world's fastest-growing online learning management system, has over 30 million users across the globe (About Canvas).

"At issue is the basic job description of a professor" (Young, 1997). One of the largest impacts of online learning tools is its role in the shifting of responsibilities of the professor. Online learning technologies vary widely in scope, from simple course gradebook management to fully autonomous educational systems that exist without the need for a human professor. Where the lines are drawn contributes a great deal towards the quality of the student experience and the role of the professor.

A core responsibility typically assumed by professors is grading. Online learning platforms commonly provide some form of automated grading, removing the bulk of this responsibility from the professor. One of the most important considerations regarding grading is feedback, which “plays an important role in the student learning process as it gives the learners greater insight into what they have actually done to arrive at an outcome” (Alharbi, 2017). While it may seem that online course tools serve as an impediment to feedback by providing an extra layer between instructors and students, a study from Price & Petre (1997) showed that “the nature and quality of feedback are comparable on paper and electronic assignments.” The same study noted that this may be due in part to the fact that “electronic assignment handling makes everyone’s handwriting better.” For computer science classes in particular, where grading scripts can compile and run student submissions immediately upon receiving the submission, automated grading also allows for nearly instant turnaround time on feedback. Web-CAT, a well-known open-source automated grading system which is known for grading students in part based on how well they test their own code, explains the added value to students that feedback from near-immediate automated grading provides: “instructors usually allow multiple submissions for a given program. This allows a student to receive immediate feedback on the performance of his or her program, and then have an opportunity to make corrections and resubmit before the due deadline” (Edwards, 2008).

Automated grading also reduces the time instructors spend grading students, theoretically freeing up that time for activities that are more beneficial for the students. “By automating the process of assessing program behavior, TAs and instructors can spend their grading effort on assessing design, style, and documentation issues.” (Edwards, 2008)

A simple but important realization is that “[a]utomated grading is a vital tool in providing quality assessment of student programs as enrollments increase” (Edwards, 2008). Automated

grading is asynchronous, breaking the standard dynamic that larger class sizes would require more time spent by the instructor on grading. Larger class sizes allow for increased enrollments, bringing revenues up on a per-professor basis. It is difficult to say where that extra money is going, but it should be factored into any discussion about whether increased class sizes are beneficial as a whole.

However, increased class sizes comes with its own set of concerns for students. A Binghamton University study found that increased class sizes caused not only a decrease in student performance, but student retention, as well (Keil & Partell). Another study, focused specifically on introductory level computer science classes, found that smaller class sizes increased student satisfaction and provided more benefit to students in traditionally underrepresented groups in addition to echoing similar increases in student performance and retention (Boyer et. al., 2011).

While automated grading may help courses scale as enrollment increases, in order to take humans completely out of the loop means that cheating must be handled in an automated fashion, as well. Cheating detection systems such as MOSS can help detect plagiarism in programming assignments in the case where students share code with each other, but these systems, by their own admission, still have flaws (Boyer & Hall, 1999). Automated cheating detection systems introduce an interesting variable in the relationship between students and professors. Traditionally, students aiming to cheat merely have to fool the professor. However, the introduction of automated cheating systems means that most professors will not take the time to check a student's submission unless the cheating detection system raises a red flag on the submission. If a student is trying to cheat, they can focus on beating the automated cheating detection system rather than the professor. Despite the prevalence of what most professors consider to be quality cheating detection systems, this shift in mindset from fooling a professor

to fooling a software program may be among the reasons why cheating is such a pervasive problem, especially in introductory-level computer science classes. The New York Times published an article in 2017 describing how cheating has increased in conjunction with increased class sizes. In the fall 2017 semester of Harvard's introductory computer science class, more than 60 out of 655 enrolled students were referred to Harvard's honor council on allegations of academic dishonesty.

Another area to explore is the user experience of these technologies. Given that more and more of the student experience is defined through students' interactions with these course tools, the user experience these tools provide can have a significant impact on students' perceptions of the university experience as a whole. A study at The University of Newcastle looked into the percentage of students who were aware of various university services. They found that Information Technology Services such as myHubOnline, the online student gateway to their own enrollment information, was the most well-known service the university provided, beating out services providing information about graduate study, scholarships, sports facilities, and even food outlets (Morahan, McConkey, & Young). Improving the student experience is among the chief concerns of the university administration, especially with the increasing role of students as consumers (Green, et. al., 1994).

While lectures are still a significant portion of the learning experience, the proliferation of supplementary online learning tools begs the question of how exactly these tools impact society. Actor-Network Theory (ANT) affords a useful framework to analyze these impacts. A core concept from ANT is the agency of non-humans in an otherwise human-centric network (ANT). In this analysis, I will look at how online learning technology is a key component to the modern education system, paying special attention to the impacts of non-human agents such as autonomous grading, cheating detection systems, and user experience.

Conclusion

The technical report will detail the ways in which my team implements meaningful improvements over the current system. The details of autograding, feedback, queuing, and user experience are among the most relevant issues.

The STS research paper will further explore the broader impacts of course tools technology on universities and how it impacts students, teaching assistants, professors, and university administrators.

Together, this research, in addition to testing performed by my team as we roll out our system next semester, should help inform how we can design our system to produce the most positive impact for a course which is taken by every undergraduate computer science student at the university.

References

About Canvas: Edtech Learning Platform. (n.d.). Retrieved October 30, 2019, from https://www.instructure.com/canvas/about.

Actor-network theory (ANT). (n.d.). Retrieved October 30, 2019, from https://web.archive.org/web/20160511003951/http://www.stswiki.org/index.php?title=Actor-network_theory_(ANT).

Alharbi, W. (2017, April 4). E-Feedback as a Scaffolding Teaching Strategy in the Online Language Classroom - Wael Alharbi, 2017. Retrieved October 30, 2019, from https://journals.sagepub.com/doi/10.1177/0047239517697966.

Bidgood, J., & Merrill, J. B. (2017, May 29). As Computer Coding Classes Swell, So Does Cheating. Retrieved October 30, 2019, from https://www.nytimes.com/2017/05/29/us/computer-science-cheating.html.

Boyer, K. W., & Hall, L. O. (1999, November 13). Experience using "MOSS" to detect cheating on programming assignments. Retrieved October 30, 2019, from http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=840376&isnumber=18097.

Edwards, S. H. (2008, July 2). Web-CAT: automatically grading programming assignments. Retrieved October 30, 2019, from https://dl.acm.org/citation.cfm?id=1384371.

Green, D., Brannigan, C., Mazelan, P., & Giles, L. (1994). Measuring student satisfaction: A method of improving the quality of the student's experience. *The student experience*, 100-107.

Learning management system. (2019, October 21). Retrieved October 30, 2019, from https://en.wikipedia.org/wiki/Learning_management_system.

Morahan, B., McConkey, K., & Young, A. (n.d.). Helping students find their place in the sun – improving the non-academic student experience. Retrieved October 30, 2019, from https://www.aair.org.au/app/webroot/media/pdf/AAIR Fora/Forum2011/Helpingstudentsfindtheirplaceinthesunimprovingthenon-academicstudentexperienceMorahanYoungMcConkey.pdf.

Price, B., & Petre, M. (1997, June 5). Teaching programming through paperless assignments: an empirical evaluation of instructor feedback. Retrieved October 30, 2019, from https://dl.acm.org/citation.cfm?id=268849.

Young, J. R. (1997, October 3). Rethinking the role of the professor in an age of high-tech tools . Retrieved October 30, 2019, from https://crab.rutgers.edu/~goertzel/rethinkingroleofprofessor.htm.