Examining Classroom Interventions to Reduce Procrastination

Stephen H. Edwards, Joshua Martin, and Clifford A. Shaffer
Virginia Tech
2202 Kraft Drive
Blacksburg, VA 24060
+1 540-231-5723
edwards@cs.vt.edu, jdm522@vt.edu, shaffer@vt.edu

ABSTRACT

Procrastination is a common problem for students. Many believe procrastination may keep otherwise competent students from succeeding. However, the most effective interventions for procrastination are resource-intensive—providing supplemental training or courses in study skills and self-regulation. These techniques do not scale to large courses. This paper investigates three new classroom interventions designed to be low-cost and low-effort to implement. Reflective writing assignments ask students to reflect on how their time management choices affect their work. Project schedule sheets require students to plan out and schedule specific tasks on their projects. E-mail situational awareness alerts give students feedback on how their progress compares to others, and to expectations. 353 students over two semesters of a junior-level advanced data structures course participated in a study where these interventions were investigated. While neither reflective writing assignments nor schedule sheets produced any significant effect, e-mail alerts were associated with both significantly reduced rates of late program submissions, and increased rates of early program submissions. As a result, this intervention shows promise for further investigation as a potential strategy for reducing late submissions among students.

Categories and Subject Descriptors

K.3.2 [**Computers and Education**]: Computer and Information Science Education; K.3.1 [**Computers and Education**]: Computer Uses in Education

Keywords

procrastination, self-regulation, self-efficacy, scheduling, schedule sheets, e-mail alerts

ITiCSE'15, July 6–8, 2015, Vilnius, Lithuania.

ACM 978-1-4503-3440-2/15/07 ...$15.00.
http://dx.doi.org/10.1145/2729094.2742632.

1. INTRODUCTION

Procrastination is a pervasive problem that significantly affects students. Many computer science educators describe procrastination as one of the common problems that leads to non-success in courses. In this context, procrastination is "voluntarily delaying an intended course of action despite expecting to be worse off for the delay" [10]. It is theorized that procrastination has a particularly negative impact on student performance. According to a meta-analysis of a broad range of procrastination studies, between 70% and 95% of undergraduates procrastinate on coursework to a degree, while between 20% and 30% of undergraduates show signs of severe procrastination [10]. Such high rates of procrastination make it no surprise that students often use procrastination as an excuse for poor educational performance. More specifically, a "negative procrastinator" is one who procrastinates to the extent he or she experiences negative consequences from their delays. Negative procrastinators are more likely to turn in work late, receive lower scores both on assignments and exams, report greater stress, and have more health concerns than their peers [12].

The effects of procrastination in CS education are particularly acute. In a study examining the performance of 1,101 CS students over a five year period, a statistically significant correlation was found between when students started working on a project and their overall work quality [5]. At the same time, however, there was no statistically significant difference in the amount of time that students spent working—in fact, starting earlier was associated with finishing earlier rather than spending more time working.

While a number of interventions intended to address student procrastination have been investigated, the most effective seems to be some form of supplementary course or workshop on motivation and time management strategies [14]. While these approaches are effective, they are also costly in both time and manpower. While they are useful for targeted populations of students at risk, they do not scale to large classrooms. As a result, educators continue to search for practical methods to address procrastination that can be used across large numbers of students.

This paper reports on the preliminary investigation of three classroom interventions that are feasible for use with little additional class time or instructor effort: active reflection writing tasks, where students write a "minute paper" after each assignment on how their time management choices affected their work; schedule sheets that students fill out, requiring them to break down tasks and show how much time they plan to allocate for each piece, helping them to form, express, manage, and track smaller-scale deadlines; and situational awareness alerts based on a model of student progress that show each student how their current efforts compare to expectations for the class as a whole. All three interventions are grounded in temporal motivation theory, which explains why people procrastinate. The interventions were carried out in a junior-level advanced data structures course over two different semesters.

In this study, e-mail situational awareness alerts showed a significant reduction in the rate of late program submissions by students, while the other two interventions did not provide evidence of consistent impact. Section 2 describes previous work on procrastination in general and how it relates to this study. Section 3 describes the three interventions presented here in more detail. Section 4 describes the study method and Section 5 presents the results, while Section 6 discusses future work to build on the results presented here.

2. BACKGROUND

Despite being such a relevant issue to student performance, there is still a lack of understanding about procrastination. Steel published a meta-analysis of research into procrastination in 2007. He defines procrastination as a "prevalent and pernicious form of self-regulatory failure" [10]. Procrastination can be further defined as delaying a task one intends to perform, in spite of potential negative consequences. Some research indicates that an individual's level of procrastination tendency may be a personality trait [10]. A number of instruments have been developed to measure one's procrastination tendency [7][13].

Many potential causes for procrastination have been researched, including self-efficacy, self-esteem, self-regulation, fear of failure, task aversion, task rewards or punishments, neuroticism, and impulsiveness. Above all, procrastination is a failure of self-regulation, where one does not exercise influence over one's own behavior in the manner required [10]. In a study of 456 undergraduates, Klassen et al. found self-efficacy for self-regulation, or a person's view of their own ability to self-regulate, was a stronger predictor of procrastination. This includes other variables such as self-regulation itself, academic self-efficacy, and self-esteem [6]. Tuckman provides reasoning for this in the following statement:

> As one proceeds through school, the responsibility for control of one's own performance shifts progressively from parents and teachers to oneself, reaching a high point during the college years. The inability to overcome procrastination tendencies may be related to problems encountered by many college students, leading some researchers to be on the lookout for effective strategies that may be used to help such students regulate their own learning [13].

A more recent theory explaining procrastination is called temporal motivation theory (TMT). Proposed by Steel, TMT describes the utility (desirability) of a task in terms of four major variables: the expectancy (E) of success, the value (V) one places on the task, the delay (D) before a reward is received, and the individual's sensitivity towards delay (Γ) [10]. The formula for utility is:

$$Utility = \frac{EV}{\Gamma D}. \tag{1}$$

This theory incorporates the idea that students prefer tasks they can complete, will enjoy, and will see the benefits of sooner over others.

Previous attempts to combat procrastination have been targeted at smaller groups. In 2005, Tuckman described utilizing a "study skills" course to teach students psychological principles and various theories about procrastination. The course also teaches students which type of rationalization they use for procrastinating, and ways to overcome these issues to meet their deadlines. Tuckman reports that students who participate in this class typically see a 0.5 increase in GPA during the next semester [14]. While such techniques are clearly effective, it is difficult to provide such resources to all students in a university.

Alternatively, many instructors try lower cost interventions, including offering extra credit incentives for early assignment completion. While many instructors report anecdotal success, a previous study did not find any evidence that extra credit bonuses for early completion of programming assignments had a statistically significant effect on when students finished [1]. Further, there are ethical concerns with such bonuses, which may disproportionately favor academically strong students, further disadvantaging students who are struggling [11].

In studying the behaviors of successful students, we have collected data on student classroom performance from the first three programming courses at Virginia Tech for five years [5]. Students who consistently scored highly on all programming assignments within a course, as well as students who consistently performed poorly, were separated out for analysis, allowing a within-subjects comparison among the large remaining group of students who earned mixed results. Analysis of this data yielded several important results. First, on assignments where a given student received scores above 80% (A/B range), that student was more likely to have started earlier and finished earlier than on assignments where that same student received grades below 80% (C/D/F range). There also was no statistically significant difference in the amount of time that students spent on their work when starting earlier; instead, they simply finished earlier. Around two thirds of the time, when a student received an A or B-level score, that student began electronically submitting their work for checking over a day before the final deadline, while two thirds of the time when a student received a lower score, that student also started electronically submitting their work for checking on the assignment's due date or later.

3. INTERVENTIONS

3.1 Active Reflection

The first of the interventions we investigated is active reflection in the form of *reflective writing assignments*, an approach inspired by active learning techniques. The intent of these assignments is to engage students in reflection about their own time management behavior and how it affects their performance. This intervention was loosely inspired by the active learning technique called a "minute paper" [9][2]. After consulting with the instructors involved, the reflection activity was expanded to writing a brief reflective response to four prompts. Although conducting the activity in class is one option, in this study students completed their reflective writing activity outside of class, entering their responses in a one-page electronic form on-line. Students received a grade for completion of this activity, amounting to 1% for each reflective writing. Assignments were given around the time each programming assignment was given, asking students to reflect on their choices in completing the previous programming assignment in this (or an earlier) course. The writing activity was as follows:

This assignment involves writing one-paragraph responses to four questions regarding the last programming assignment you completed for this class, and should only take you about 15-20 minutes to complete.

Consider the following questions and answer each one in a separate paragraph.

1. *On the last software project you worked on for a class, consider when you began working on it, the way that you spent your time, and how you managed yourself as you worked to complete the project. Then describe the key elements of the plan or strategy you used to manage your time on that project.*

2. *Again considering the most recent software project you completed for a class, describe how your strategy for managing your time on the project affected the quality of your work or your ability to achieve your goals on the project.*

3. *Reflecting on your experience with past projects, describe the strategy or plan you intend to use for managing your time on the next software project that you will do for this class.*

4. *Again considering the most recent software project you completed for a class, a) describe your development strategy that you used to implement and test the program, b) describe how your development strategy affected the quality of your work or your ability to achieve your goals on the project, and c) describe any different strategy to implement and test your program that you will use for the next software project that you will do for this class.*

3.2 Schedule Sheets

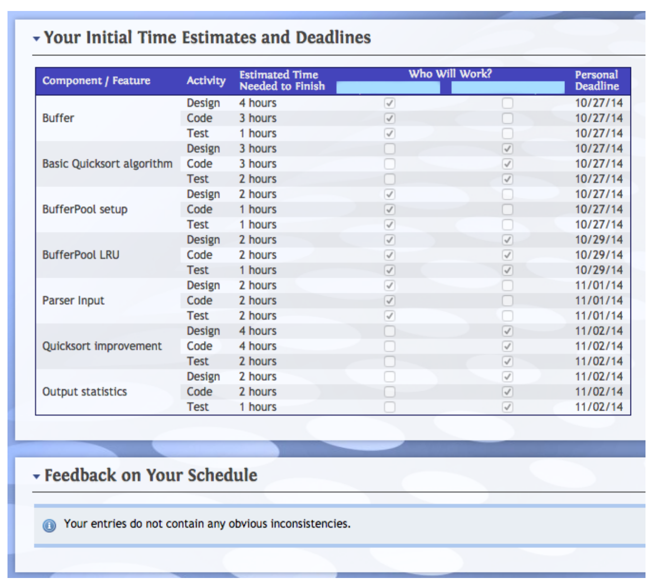
The second intervention we investigated was the use of *schedule sheets*. For several years, instructors in our junior data structures course have been using "painless" schedule sheets [8]. Student survey data indicate that many students find them helpful for managing their projects, while others find them to be unhelpful. The purpose of schedule sheets is to encourage students to set a series of small, intermediate deadlines themselves, and to think about their progress on these tasks periodically. By helping students to form, express, manage, and track smaller-scale deadlines, this intervention aims to reduce procrastination.

To provide for effective schedule administration, an electronic schedule entry system was developed to allow students to submit schedules on-line. Students entered (or changed) their work breakdown structure in the form of a series of tasks–often, specific modules, classes, or identifiable pieces of required behavior. Each task had form fields for the student to enter the estimated number of hours of design time, coding time, and testing time, although students could optionally leave one or more of these blank if they were not applicable. In addition to estimating the number of hours for each task, students were also required to enter their own personal target deadline for completing that task.

Student projects in the course used for this study lasted approximately four weeks on average. Students were required to fill out an initial schedule sheet showing their task breakdown and initial elements early in the process, typically within a week of when the assignment was given. As students progressed, they were required to turn in an intermediate schedule sheet approximately one week before the assignment was due. This intermediate sheet asked the student to report the number of hours actually spent on each task, as well as which tasks had been completed. Next, the student would have a chance to revise or update their previously entered schedule, entering revised estimates for time remaining to complete tasks and updating personal deadlines as necessary. When the assignment was due, students turned in a final schedule reporting the amount of time spent on completing tasks during the final week of work.

To promote more effective scheduling and to encourage students to plan their time effectively, students received immediate automated feedback as they entered their schedule information. A "check my work" button allowed students to get immediate feedback at any point while editing their schedule. The automated feedback included a large number of diagnostics on the consistency and appropriateness of time entries and deadlines on individual tasks and on the entire schedule as a whole. As a result, students were alerted to missing required elements, to over scheduling their own time, to creating schedules that were too coarse-grained or too fine-grained, and so on. A total of 23 different kinds of automated checks were performed to provide this feedback.

In addition to automated feedback, the scheduling system also permitted manual review of all schedules by the instructor and course staff. The instructor hand-checked all schedules that required attention and wrote additional feedback comments. For each programming assignment, students re-



Your Initial Time Estimates and Deadlines

Component / Feature	Activity	Estimated Time Needed to Finish	Who Will Work?		Personal Deadline
Buffer	Design	4 hours	✓		10/27/14
	Code	3 hours	✓		10/27/14
	Test	1 hours	✓		10/27/14
Basic Quicksort algorithm	Design	3 hours		✓	10/27/14
	Code	3 hours		✓	10/27/14
	Test	2 hours		✓	10/27/14
BufferPool setup	Design	2 hours	✓		10/27/14
	Code	1 hours	✓		10/27/14
	Test	1 hours	✓		10/27/14
BufferPool LRU	Design	2 hours	✓	✓	10/29/14
	Code	2 hours	✓	✓	10/29/14
	Test	1 hours	✓	✓	10/29/14
Parser Input	Design	2 hours	✓		11/01/14
	Code	2 hours	✓		11/01/14
	Test	2 hours	✓		11/01/14
Quicksort improvement	Design	4 hours		✓	11/02/14
	Code	4 hours		✓	11/02/14
	Test	2 hours		✓	11/02/14
Output statistics	Design	2 hours		✓	11/02/14
	Code	2 hours		✓	11/02/14
	Test	1 hours		✓	11/02/14

Feedback on Your Schedule

Your entries do not contain any obvious inconsistencies.

Figure 1: A screenshot of the schedule contents. Student names have been blocked out; checkboxes show which student is participating in which tasks.

ceived a grade corresponding to 1% of their semester grade, spread among the three schedule sheets for that assignment.

3.3 Situational Awareness Alerts

The final intervention we investigated was the use of automatically generated e-mail *situational awareness alerts*. These were periodic messages sent to students as they work on an assignment to raise their awareness of how their current level of effort compares to what others are doing. Because students had no concrete deliverables to turn in as part of this activity, they did not receive any grade for e-mail alerts. Instead, this group completed additional homework to earn the 1% grade credit allocated to other interventions.

While instructors routinely tell students to start programming projects early and spread the workload over time, admonishments alone appear to do little to affect student behavior. This intervention takes a different approach by providing alerts that are meaningful because they are individually relevant. Based in part on past student assignment data, we constructed a model of student progress that could be used to automatically assess a student's progress. In this study, students electronically submitted their program assignments using Web-CAT, an open-source automated grading system for programming assignments [3][4]. As part of this work, we built an automated alert system as a prototype extension to Web-CAT. By using data from the students' work-in-progress, in comparison with the model that we developed, it is possible to provide feedback about how well the student is doing compared to classroom expectations.

The e-mail alert treatment involved sending periodic e-mail messages to the students, beginning approximately one week before the assignment was due, then again at 4 days before and 2 days before the assignment deadline. Each e-mail alert was customized based on the work most recently submitted for electronic checking at that point. The student's work was classified along 4 dimensions: the amount of code written (relative to an approximate target size for the given assignment), the proportion of instructor-written reference tests passed (an approximation for functional correctness), the degree of testing performed (if the assignment requires students to write their own software tests), and the number of static analysis checks failed (measuring adherence to coding style guidelines, if required by the assignment). Along each dimension, the student's work so far was rated on a 4-point scale: Good, meaning the student's progress indicated advanced progress compared to the model for the amount of time remaining before the due date; Neutral, meaning the student's program was in line with the model for the amount of time remaining; Bad, meaning the student's progress was significantly behind the model; or Undefined, meaning there was insufficient information to assess the student's work.

A custom e-mail message was generated based on the four ratings. The subject line of the e-mail was phrased as "CS 3114: Your progress on Project 2". However, if the student's work indicated insufficient progress in one or more dimensions, the subject line would instead be "CS 3114: You may be at risk on Project 2", or even "CS 3114: You are at risk on Project 2". The body of the e-mail message contained a separate paragraph corresponding to each of the 4 dimensions on which the student's work was rated. As an example, here is an alert message that was automatically generated for one student:

This notification is to increase your awareness of your current progress on Project 4 compared to the rest of the class. Project 4 is due in 4 days.

Based on the work you have submitted to Web-CAT, it looks like you are making good progress towards a working solution for this assignment. Starting early is associated with a statistically significant increase in scores earned, compared to when the same student starts later. This increases your chances of success on the assignment.

Based on the tests you have submitted, it appears that you may be waiting until later to test your work, instead of testing it incrementally as you develop it. Typical students earn higher scores when they write tests incrementally with their code compared to when the same student waits to write tests until the code is substantially complete. This increases your risk of performing poorly on the assignment.

Based on the style checks on your assignment, it appears you are not formatting your code as expected when you write it, and may be intending to go back and correct the formatting later. Adjusting your code writing style so that you produce properly formatted and documented code as you write will increase your efficiency and reduce the time needed to clean up code later.

We wish you the best of luck as you work to complete this assignment.

– Web-CAT Situational Awareness Service

4. METHOD

We studied these three interventions in four separate sections of CS 3114: Data Structures and Algorithms, over two separate semesters. In Fall 2013, one section of this course had no treatment or intervention, serving as a control condition, while a second parallel section used the reflective writing assignment treatment. In Fall 2014, one section of this course used schedule sheets, while a second parallel section received e-mail situational awareness alerts. In each of the sections under study, students received the same treatment on all programming assignments in the course. In total, 362 students participated in the course as part of these four sections. All sections were taught by the same instructor using the same techniques and course materials. 353 of the students consented to the use of their data for educational research in this study (control: 71, reflective writing: 79, schedule sheets: 100, e-mail alerts: 103).

Students in all four sections completed a total of 4 separate programming assignments. Assignments ranged in difficulty and size, with solutions averaging from approximately 400-1800 non-commented/non-blank source lines of code. While the same assignments were not used in the two semesters, the assignments were comparable in complexity. In both semesters, the third assignment was the shortest, with the second and fourth assignments being hardest/longest. All assignments involved implementing one or more data structures, together with a simple command line interpreter modeled on entering, retrieving, and operating on real-world data to be inserted into the structure(s).

Students were required to write their own software tests to test their own work, and turned in both software and tests together using Web-CAT. Students received immediate feedback on correctness and programming style, and were encouraged to revise and resubmit their work frequently to maximize their scores. In all four sections, students were allowed to work in pairs on programming assignments, where they were encouraged to use pair programming [15], the extreme programming technique where two programmers work together with one playing the role of reviewer or "navigator" while the other actively writes as the "driver". However, since work on programming assignments was completed outside of class, no supervision of pair activities was available and the degree to which students adhered to pair programming practices is unclear. Students were also given the option to complete projects alone without a partner, although the majority of students chose to work together. Program assignments were then classified as "late" if they were submitted after the due date had passed.

In all programming assignments across both semesters, the insturctor offered students an extra credit bonus for completing the assignment at least one day early. This bonus was equivalent to 10% of the total project score. Since this bonus was consistent across all groups including the control group, and prior work did not show any evidence for its effectiveness, it does not appear to have affected the study.

5. RESULTS

Figure 2 shows the percentage of assignments turned in late after the due date for each treatment condition, together with the corresponding percentage of assignments completed at least one day early. Across all four assignments, 42.3% of programs were submitted late vs. 35.0% early in the control group, 38.6% late vs. 43.3% early in the reflective writing group, 38.1% late vs. 34.1% early in the schedule sheets group, and 32.4% late vs. 46.1% early in the e-mail alerts group. The data are suggestive, in that there were lower rates of late submissions for all interventions compared to the control, and higher rates of early submissions for two.

By using both the assignment and the treatment condition as independent variables, a multinomial logistic regression on whether programs were submitted late indicated that the null hypothesis that no differences were observed should be rejected ($DF = 6, \chi^2 = 44.2, p < 0.0001$). However, likelihood ratio test results did not indicate clear evidence for an effect from treatment condition alone ($DF = 3, \chi^2 = 6.9, p < 0.076$), while assignment differences were clearly significant ($DF = 3, \chi^2 = 38.0, p < 0.0001$). Instead, only the difference in late submissions between the e-mail situational awareness alerts and the control group were significant ($DF = 1, \chi^2 = 6.6, p = 0.01$).

From these results, it is also clear that the assignments themselves have an effect. Figure 3 shows the percentage of assignments turned in late for each assignment, grouped by treatment condition. While some instructors believe procrastination increases later in the semester, here this was not the case. Instead, the high point seems to be on the second programming project, which is one of the two harder and longer assignments. One possible explanation is that students who are struggling early in the course decide to drop out after that point, leaving only more persistent students who are more likely to complete work on time. Of course, other explanations are possible and it is likely that multiple factors are important.

Overall, however, these results are encouraging. First, it is notable that all four groups had comparable rates of late submissions on the first programming assignment, averaging near 40%. On the very first assignment, students had no previous exposure to the interventions being studied and were just trying them out for the first time. It is also worth

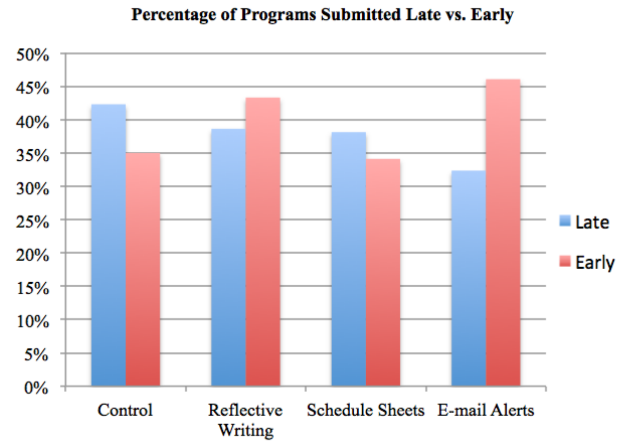


Figure 2: The proportion of assignments turned in late and early for each treatment condition.

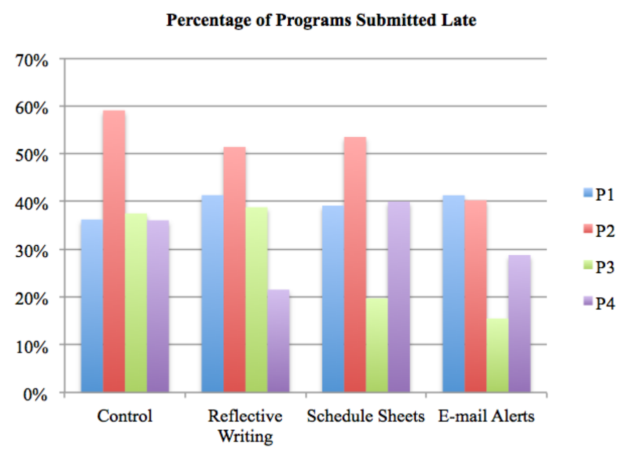


Figure 3: The proportion of assignments turned in late for assignments P1–P4, grouped by treatment condition.

noting that all assignments except for Project 2 had similar rates of late submissions throughout the semester in the control group—Projects 1, 3, and 4 were all between 36–38%. At the same time, however, all three interventions showed fewer late submissions on at least some later assignments, with the e-mail alert intervention showing consistently lower late submission rates on all assignments after the first.

Second, it appears the reflective writing intervention had minimal impact. A chi square test between just the reflective writing group and the control group does not reveal any difference ($\chi^2 = 0.75, p < 0.39$). While the number of late submissions was lower for Projects 2 and 4 compared to the control, the number was actually slightly higher for the other two projects. The schedule sheet intervention similarly had a questionable impact ($\chi^2 = 1.01, p < 0.32$), with two assignments having greater rates of late submissions.

In comparison, e-mail situational awareness alerts produced the only significant impact ($DF = 1, \chi^2 = 6.6, p =$

0.01). Students in this group saw a one third reduction in late submissions on the second assignment compared to the control group, a 58% reduction on the third assignment, and a 20% reduction in the final assignment. At the same time, as shown in Figure 2, the greatest proportion of students finished at least a day early with this intervention as well. This increased rate of early submissions was also significant ($DF = 1, \chi^2 = 7.1, p < 0.01$). However, while promising, it is important to keep in mind that differences between the assignments were also quite significant, and more research is necessary to strengthen evidence for this treatment.

6. CONCLUSIONS

This paper summarizes an investigation of three classroom interventions intended to address student procrastination in programming assignments. Of the interventions studied, the reflective writing assignments did not produce evidence for any significant impact on timeliness of student work. However, changes to the intervention, such as having students discuss their reflections, perhaps using a think-pair-share classroom activity or a peer learning activity, could improve its effectiveness. Similarly, schedule sheet assignments also did not produce consistent evidence for a significant impact.

This study does indicate that e-mail situational awareness alerts showed a statistically significant reduction in late submissions (23% fewer on average), as well as a statistically significant increase in early completion (31% more on average) compared to the control group. Because of the nature of studies such as this, however, there is a risk that other factors may also have played a role, such as differences in student populations, differences in assignments, or differences in competing class deadlines in other courses. Any of these factors could have contributed to the observed differences between the rate of late submissions between the groups.

To investigate further, we hope to conduct a future study using a randomized block design that will support within-subjects comparisons across treatments. Such a follow-on study may shed additional light on the scope of effect that may be achieved with interventions such as these. Also, it may be worthwhile to study them in combination—do they produce a bigger impact if they are used together?

At the same time, these interventions are only practical if instructors have access to the necessary tools. Both schedule sheets and e-mail alerts are only feasible with automation, since they would be prohibitively costly to employ manually. As a new form of intervention that is more data-driven, and that provides more immediate and more individual feedback, the personal nature of these interventions may be one part of why they have an impact. To this end, we have made the prototype tools used in this study available as part of the Web-CAT open source project. While the tools themselves are not ready for more generalized use by a wider audience, we hope that incorporating them into a more widely adopted project will make it easier for other educators and researchers who wish to explore these techniques.

7. ACKNOWLEDGMENTS

This work is supported in part by the National Science Foundation under grant DUE-1245334. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

8. REFERENCES

[1] A. Allevato and S. Edwards. Effects of extra credit opportunities on students' time management on programming assignments. In *Proceedings of the IEEE 43rd Annual Frontiers in Education Conference*, pages 1831–1836. IEEE, October 2013.

[2] C. Bonwell and J. Eison. *Active Learning: Creating Excitement in the Classroom.* School of Education and Human Development, George Washington University, 1991.

[3] S. H. Edwards. Using software testing to move students from trial-and-error to reflection-in-action. In *Proceedings of the 35th SIGCSE Technical Symposium on Computer Science Education*, SIGCSE '04, pages 26–30, New York, NY, USA, 2004. ACM.

[4] S. H. Edwards and M. A. Pérez-Quiñones. Experiences using test-driven development with an automated grader. *J. Comput. Sci. Coll.*, 22(3):44–50, Jan. 2007.

[5] S. H. Edwards, J. Snyder, M. A. Pérez-Quiñones, A. Allevato, D. Kim, and B. Tretola. Comparing effective and ineffective behaviors of student programmers. In *Proceedings of the Fifth International Workshop on Computing Education Research*, pages 3–14. ACM, 2009.

[6] R. M. Klassen, L. L. Krawchuk, and S. Rajani. Academic procrastination of undergraduates: Low self-efficacy to self-regulate predicts higher levels of procrastination. *Contemporary Educational Psychology*, 33(4):915 – 931, 2008.

[7] C. H. Lay. At last, my research article on procrastination. *Journal of Research in Personality*, 20(4):474–495, 1986.

[8] J. Spolsky. Painless software schedules. `http://www.joelonsoftware.com/articles/fog0000000245.html`. Accessed: 2015-01-11.

[9] D. Stead. A review of the one-minute paper. *Active Learning in Higher Education*, 6(2):118–131, July 2005.

[10] P. Steel. The nature of procrastination: a meta-analytic and theoretical review of quintessential self-regulatory failure. *Psychological Bulletin*, 133(1):65, 2007.

[11] J. Stodder. Experimental moralities: Ethics in classroom experiments. *The Journal of Economic Education*, 29(2):127–138, 1998.

[12] D. M. Tice and R. F. Baumeister. Longitudinal study of procrastination, performance, stress, and health: The costs and benefits of dawdling. *Psychological Science*, pages 454–458, 1997.

[13] B. W. Tuckman. The development and concurrent validity of the procrastination scale. *Educational and Psychological Measurement*, 51(2):473–480, 1991.

[14] B. W. Tuckman. Relations of academic procrastination, rationalizations, and performance in a web course with deadlines 1. *Psychological Reports*, 96(3c):1015–1021, 2005.

[15] L. Williams. Lessons learned from seven years of pair programming at North Carolina State University. *SIGCSE Bull.*, 39(4):79–83, Dec. 2007.