2022 IEEE/ACM 44th International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET) | 978-1-6654-9592-9/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICSE-SEET55299.2022.9794148

Peer Instruction in Online Software Testing and Continuous Integration – A Replication Study

Bhuvaneswari Gopal
bhuvana.gopal@unl.edu
School of Computing, University of Nebraska-Lincoln
Lincoln, Nebraska, USA

Stephen Cooper
stephen.cooper@unl.edu
School of Computing, University of Nebraska-Lincoln
Lincoln, Nebraska, USA

ABSTRACT

This paper discusses the results of replicating and extending a previous study on the active learning pedagogy of Peer Instruction (PI) in the topics of unit testing, integration testing and continuous integration. The original paper studied the efficacy of PI as a pedagogy for an in-person classroom with honors students from various academic majors. In this replication study we focus on a fully virtual, synchronous online classroom consisting of computing related majors. Our findings reinforce the results in Gopal and Cooper's original study. Cognitively, we found a correlation between PI and student learning, by observing encouraging increases in levels of success as measured through cognitive pre- and post-course instrument for the topics we studied. In addition, we also found that PI had a statistically significant impact on student attitudes in the constructs of interest, gender, usefulness, confidence and professionalism.

CCS CONCEPTS

• **Software and its engineering**; • **Social and professional topics** → **Computing education**; • **Human-centered computing** → *Empirical studies in collaborative and social computing*;

KEYWORDS

Software Engineering, Software Engineering Education, Unit Testing, Integration Testing, Continuous Integration, DevOps, Sofware Testing, Peer Instruction, Active Learning, Online Learning

ACM Reference Format:
Bhuvaneswari Gopal and Stephen Cooper. 2022. Peer Instruction in Online Software Testing and Continuous Integration – A Replication Study. In *44nd International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET '22), May 21–29, 2022, Pittsburgh, PA, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3510456.3514168

1 INTRODUCTION

This paper presents the design of and results from a replication study utilizing Peer Instruction (PI) [24], in comparison with purely lecture-based content delivery, for an online Software Engineering (SE) class. Our specific interest was in how PI "worked" in an online SE course. While we hope that, in a post Covid-19 pandemic world, we will be returning to in-person classes, we recognize that online instruction is here to stay. We use the opportunity of Covid-19 to explore the extent to which PI can be successfully used in a mid-level online CS class. The organization of this paper is as follows: Section 2 contains an overview of related literature. Section 3 poses research questions we set out to answer. Survey design and methodology are introduced in Section 4 and results presented in Section 5. Results are analyzed in Section 6. Section 7 explores threats to the validity of our study, and Section 8 concludes.

ICSE-SEET '22, May 21–29, 2022, Pittsburgh, PA, USA

ACM ISBN 978-1-6654-9592-9/22/05…$15.00
https://doi.org/10.1145/3510456.3514168

2 RELATED LITERATURE

Our work combines two distinct topics: software testing education and online education. To that end, we have classified relevant literature in software testing education and PI into two subsections. We present each separately.

2.1 Studies on teaching software testing and continuous integration

The ACM CS 2013 report [20] describes the need for software testing to be an integral part of the undergraduate CS curriculum. Students with well-developed testing skills become better software engineers as the act of testing also forces the integration, application, and enhancement of the other critical software development skills such as analysis, design and implementation. But software testers of the future may not be adequately prepared for the task at hand, as observed by Buffardi, due to the lack of testing education in our colleges [4], and echoed by Wong [32]. Garousi [14] observed that studies around the globe have shown that developers understand the importance of software testing education. Yet many of them still lack formal testing related education [5]. A few studies present difficulties, opportunities and challenges with software testing education [1, 3, 6, 8, 10, 21, 26], with potential remedies proposed by [2, 11]. In DevOps, particularly continuous integration, Eddy [9], Greising [17] and Süß [31] have explored how to successfully integrate continuous integration as a topic into SE classes. However, there is little focus on how well students learned the topic.

2.2 Peer Instruction (PI) - in CS and online

PI was originally developed by Mazur [24] in 1991, as an evidence-based pedagogical practice. In PI, students actively participate in their own learning by discussing questions with small groups of peers during the class session, providing real-time feedback to the instructor. PI is relatively new to computer science education and has been studied since at least 2000 [33]. Studies have shown that PI is well-liked by both students and instructors [27, 29]. Several studies [7, 23] have shown that students learn from the discussion

phase during PI. In SE, PI has been studied in the context of student discussions and risk management [13]. Gopal and Cooper [16] examined the efficacy of PI in a sophomore software engineering class conducted in person, in unit testing (UT), integration testing (IT) and continuous integration (CI) and found significant score increases. In their work employing a fine grained analysis of clicker data, Gopal and Cooper [15] classified students' response patterns for each SE topic into various bins and reported overall correctness on each type of vote. They also tracked these patterns from in-class to the exam, and quantified absolute percentages of students that demonstrate longer-term learning from the PI process.

PI in online classes is a relatively new topic without much established research. Nerantzi [25] provided practical advice for course designers, module and program leaders on how they can utilize PI to maximize student engagement and learning during the Covid-19 pandemic [25]. However, their broad recommendations were based on previous literature, rather than on actual study data, and were not specific to CS. A study by Englund [12] described the design and evaluation of an online, asynchronous tool that mirrors the PI process in three large-scale courses in physics. As we can see above, PI in online computer science (CS) classes has yet to be explored. The unique contribution of our work is in studying the effectiveness of PI in a completely online, synchronous software engineering (SE) class.

3 RESEARCH QUESTIONS

This paper discuses our attempt to replicate Gopal and Cooper's [16] results with a completely synchronous online sophomore software engineering class with non-honors students belonging to computing majors. Our research questions for this study were:

RQ1: Does PI help students in an undergraduate, synchronous, online SE class learn the concepts of unit testing, integration testing and continuous integration better than students in an online, synchronous, purely lecture format class?

RQ2: Does PI increase the enjoyment of students in learning computer science concepts in an undergraduate, synchronous, online SE class compared to students in an online, synchronous lecture classroom?

4 SURVEY DESIGN AND METHODOLOGY

4.1 Replication study design

This research project was determined to be exempt by our university's Institutional Review Board. We targeted students enrolled in a semester long (15 week) sophomore/junior level software engineering course at a large R1 university, mandatory for sophomores and juniors before they progressed to the senior capstone. The entire course was conducted synchronously online through a remote video-conferencing tool, Zoom. We used a quasi-experimental pretest-post test, hybrid, between groups and within groups design for this study. The control and treatment groups consisted of successive cohorts of students. In Gopal and Cooper's original study [16], students belonged to several academic majors such as CS, SE, Computer Engineering (CE), Actuarial Science, Marketing, Finance, Electrical Engineering and Business. In our study, students primarily belonged to CS or CE majors. Tables 1 and 2 show the distribution of students for treatment and control groups based on academic majors and college standing. Our treatment group was run in Fall 2020 whereas our control group was run in Spring 2021.

Table 1: Treatment Group Student Distribution

Major	sophomore	junior	senior	Total	%
CS	8	25	9	42	70
CE	2	12	4	18	30
Total	10	37	13	60	100
%	17	62	22	100	

Table 2: Control Group Student Distribution

Major	sophomore	junior	senior	Total	%
CS	3	16	17	36	72
CE	0	8	6	14	28
Total	3	24	23	50	100
%	6	48	46	100	

Treatment group students were instructed using PI in a completely online synchronous Zoom classroom, while control group students received online synchronous instruction through traditional lectures. Both groups were instructed in consecutive semesters by the same instructor with the same content. Students typically take this software engineering course their sophomore or junior year, as it is a prerequisite to the year-long senior capstone. Students who take our SE course as seniors have typically failed one or more previous CS courses, and will not graduate at the end of the senior year, since they will take SE during the senior year and not sophomore/junior year. This indicates that they could be academically weaker students. There are more than double the percentage of seniors in our control group, and we therefore broke our control and treatment groups into two subgroups: sophomores+juniors (the group that is taking the SE course "on time"), versus seniors (who are behind). We hoped that these subgroups would help us more honestly measure the effect of PI on our online students. It meant that the data for our subgroups did not end up being normally distributed, while our overall data were normally distributed.

4.2 Implementation of PI in the online software engineering classroom

Our implementation of the PI cycle [27] involved peer discussion sandwiched between two votes, and was identical to Gopal and Cooper's original study [16], except that the entire cycle was conducted synchronously online through Zoom. At multiple points during the lecture, students engaged with these multiple-choice questions designed to help them confront and explore challenging concepts [15, 23], and answered them using clickers [19]. The specific sequence of steps for a PI cycle/clicker question episode in class was as follows:

(a) **Initial vote (V1)**: Multiple choice question was presented, and students answered individually (results are withheld from the class). If the number of students who answered the question correctly fell below 30%, as per Mazur's guidelines [24], the instructor facilitated a breakout session over Zoom.

If the threshold was not met, the instructor proceeded with the next portion of the lecture.

(b) **Zoom breakout room discussion in small groups (3-5 students)**: Students discussed their answers over Zoom in pre-assigned breakout rooms and shared their reasoning with each other, with a chance to convince their Zoom groupmate(s) to change their responses for the upcoming second vote.

(c) **Second vote (V2)**: Students answered a second time, perhaps changing their answer based on their peers having persuaded them during group discussion. The duration of V1 and V2 can each range between 45 seconds and 2.5 minutes, depending on the difficulty of the PI question being answered.

(d) **Instructor led discussion**: After V2, the instructor would lead a class-wide discussion by asking students to share explanations and discussions they had in their group and providing clarification of how the question can be analyzed. The correct answer was clearly indicated. The results of student responses were displayed at this point [15]. Similar to Gopal and Cooper's original study, we put forth significant effort in the classroom to explain to students why PI was being used and why it could be beneficial for them.

4.3 Instruments used

We utilized the cognitive and affective questionnaires published in Gopal and Cooper's original study [16], in the same order, for both the treatment and control groups, both pre- and post- intervention. There were 3 topics in the cognitive portion for a total of 16 questions – 6 in unit testing, 3 in integration testing and 7 in continuous integration. All questions were multiple choice with one single correct answer. The validated instrument utilized to measure student attitudes [18] included five constructs - confidence, interest, gender, usefulness and professionalism. During class time, we utilized the PI questions published in Gopal and Cooper's original study [16]. Students were not compensated for their participation in the surveys.

5 RESULTS

5.1 Cognitive - Overall results

We found that students in the treatment group had larger increases in scores. There were 60 students in the treatment group and 50 students in the control group. Table 3 shows what percentage of students answered each question correctly for the pre and post surveys. For example, if 10 students got a Q2 right in the treatment group for the pre survey, the value would be 0.16, 10/60. In our table, TG and CG denote the treatment group and control group respectively.

The response scores data from each question were analyzed for normality using the Shapiro-Wilk test, and homoscedasticity was determined using Levene's test for homogeneity of variance. Based on the results of these tests, we employed the appropriate ANOVA techniques, with a between groups (control vs treatment) and within groups (pre vs post survey) design, based on the groups we were analyzing. We utilized a regular one-way ANOVA or Welch's ANOVA for normally distributed data, and the non-parameteric Kruskal-Wallis test for data that was not normally distributed [28]. We ran the test on the four different score distributions – pre-control vs pre-treatment; pre-control vs post-control; pre-treatment vs post-treatment; and post-control vs post-treatment. We report all our results at a significance level of $\alpha = 0.05$.

Table 3: Percentage of students who answered each question correctly (n for TG=60, n for CG=50)

Topic	Pre-TG	Post-TG	Pre-CG	Post-CG
Unit Testing				
Q1	0.53	0.91	0.35	0.53
Q2	0.33	0.62	0.31	0.40
Q3	0.48	0.82	0.40	0.45
Q4	0.35	0.88	0.33	0.42
Q5	0.17	0.64	0.20	0.34
Q6	0.17	0.76	0.12	0.30
Q1-Q6 Average	0.35	0.77	0.28	0.36
Integration Testing				
Q7	0.50	0.94	0.40	0.50
Q8	0.33	0.90	0.17	0.35
Q9	0.10	0.75	0.14	0.20
Q7-Q9 Average	0.31	0.86	0.23	0.35
Continuous Integration				
Q10	0.10	0.88	0.05	0.15
Q11	0.12	0.88	0.10	0.15
Q12	0.13	0.76	0.20	0.20
Q13	0.13	0.85	0.10	0.10
Q14	0.10	0.88	0.05	0.20
Q15	0.28	0.76	0.30	0.35
Q16	0.38	0.76	0.35	0.60
Q10-Q16 Average	0.21	0.82	0.15	0.25
Overall Average	0.28	0.82	0.20	0.34

5.1.1 Cognitive pre-survey: Determining starting points for control and treatment groups. In comparing learning outcomes on the basis of the scores on the pre survey for the control group versus the treatment group, we utilized the regular one-way ANOVA test to analyze the variance in the distribution of pre survey scores. We found that the two groups had similar starting points in the topic of unit testing ($p = 0.32$), integration testing ($p = 0.19$) and continuous integration ($p = 0.22$). The knowledge levels where the two groups started off was not statistically different. An overall analysis of the pre survey scores for both groups yielded a similar result ($p = 0.24$).

When the sophomores+juniors subgroup in the control group were compared with sophomores+juniors in the treatment group, the results for analyzing the variance in the distribution of scores were not statistically significant (Kruskal-Wallis p=0.28). Similar results were obtained when we compared the control group seniors with the treatment group seniors (Kruskal-Wallis p=0.31). However, within the control or treatment groups, when we compared juniors+Sophomores with seniors, the results were weak (p=0.068), which, while still technically statistically not significant, warranted further investigation. We found this to be the case across

all 3 cognitive topics (unit testing, integration testing and continuous integration). This seemed to indicate that sophomores+juniors seem to start out somewhat stronger than the seniors, since this is typically a sophomore/junior level course.

5.1.2 Post control vs post treatment. Comparing learning outcomes on the basis of the post survey scores for the control group versus the treatment group, our results showed that the two groups differed significantly on how much they knew, as evidenced by their overall post survey scores: χ^2 = 10.23, p < 0.01. Our results of the Kruskal-Wallis tests for each subgroup based on college standing are: unit testing χ^2 = 9.47, p = 0.0121; integration testing χ^2 = = 15.06, p = 0.0095; and continuous integration χ^2 = 16.63, p = 0.0092. Our results of the ANOVA tests for each topic, for pre- and post- scores of the treatment group are : unit testing χ^2 = 8.39, p < 0.01; integration testing χ^2 = 10.19, p < 0.001; and continuous integration χ^2 = 19.87, p < 0.001. Our results showed that in each of the 3 topics as well as the overall scores, we found p values < 0.05, indicating that Students in the online PI class scored significantly higher than students in the online, purely lecture-based class. Figure 1 shows the distribution of overall scores in the post-treatment group and post- control group.

5.1.3 Pre treatment vs post treatment. Within the treatment group, our data were normally distributed, and we utilized a one-way ANOVA for analyzing the variance in the score distributions pre- and post- test. Based on our results (p < 0.001), we found that students in the PI class (treatment group) scored significantly higher on their post survey than in the pre survey, as evidenced by their scores χ^2 = 55.27, p < 0.0001. Our results of the ANOVA tests for each topic, for pre- and post- scores of the treatment group are: unit testing χ^2 = 8.39, p < 0.01; integration testing χ^2 = 10.19, p < 0.001; and continuous integration χ^2 = 19.87, p < 0.001. Our results reinforce the findings by Gopal and Cooper's original study [16].

When broken down by college standing, our data were not normally distributed, and we utilized the non-parameteric Kruskal-Wallis test. We found statistically significant results overall for all topics combined, as well as for unit testing, integration testing and continuous integration (Kruskal-Wallis p < 0.001), across both college standing subgroups (sophomores+juniors, as well as seniors). We did not find statistically significant differences within the control group for each subgroup, for each of the 3 topics. Figures 1 and 2 show the results for the cognitive instrument between and within groups respectively. Figure 1 shows the distribution of overall scores in the post-treatment group and post- control group. Figure 2 shows the distribution of scores in the pre and post-survey for the treatment group.

5.2 Affective questionnaire results

5.2.1 Affective - starting points based on pre survey. When sophomores+juniors in the control group were compared with sophomores+juniors in the treatment group, the results were not statistically significant (p=0.28). Similar results were obtained when we compared control group seniors with treatment group seniors (p=0.31).

5.2.2 Affective - pre vs post-survey - treatment group. Within the treatment group, we found statistically significant differences for the affective post-survey, indicating that students in the online

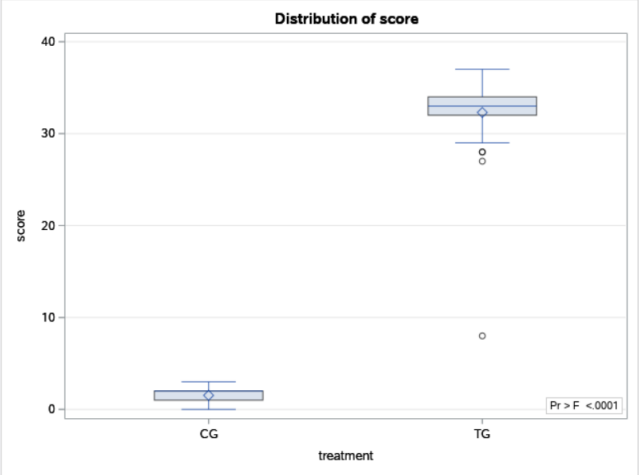

Figure 1: Cognitive post survey- control vs treatment

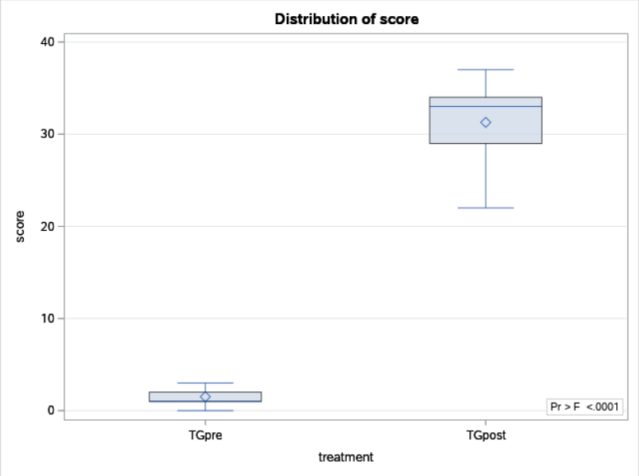

Figure 2: Cognitive pre vs post survey- treatment group

PI class scored higher than those in the online lecture class. Our results are as follows: Overall- χ^2 = 112.27, p <.0001; Interest- χ^2 = 92.94, p <.0001; Gender- χ^2 = 70.41, p <.0001; Usefulness- χ^2 = 72.46, p <.0001; Confidence- χ^2 = 95.27, p <.0001; Professionalism- χ^2 = 67.13, p <.0001. When broken down by college standing, we found statistically significant results overall for all topics combined, as well as for each construct (Kruskal-Wallis p < 0.001), for the sophomores+juniors category, as well as the seniors category. We did not find statistically significant differences within the control group.

5.2.3 Affective post-survey: treatment vs control. When we compare affective outcomes on the basis of the post survey scores for treatment vs control group, our results showed that the two groups differed significantly on how they perceived the 5 different constructs pre- and post PI intervention. In each construct as well as the overall scores, we found p values < 0.05, indicating statistically

significant results. Our results were as follows: Overall- χ^2 = 112.27, p <.0001; Interest- χ^2 = 101.92, p <.0001; Gender- χ^2 = 50.23, p <.0001; Usefulness- χ^2 = 65.55, p <.0001; Confidence- χ^2 = 93.78, p <.0001; Professionalism- χ^2 = 69.78, p <.0001. Students in the online PI class had a better perception of all 5 constructs after PI than students in the online, purely lecture-based class.

6 DISCUSSION

6.1 Overall gains

We obtained cognitive survey results similar to Gopal and Cooper's original study [16], as shown in Table 3. We found the largest increases in percentages of students who answered post-survey questions correctly, in the topics of integration testing and continuous integration. These results are similar to Gopal and Cooper's original study (Table 3) [16]. Our results also reinforce the findings from previous studies [22, 27, 30].

In Gopal and Cooper's original study, specific perception shifts for the affective survey were limited to the gender and usefulness constructs. We found significant shifts in student perceptions regarding the gender, usefulness, professionalism, interest and confidence constructs. These are previously unreported results.

6.2 Affective shifts

It is interesting to note that the online PI class exhibited more favorable views on the 5 different constructs in the affective survey. A significant difference on the impact of instructional method was observed when considering student attitude regarding gender, confidence, professional (value), interest in and usefulness in studying CS. Increased post-survey scores in the interest construct denote that students perceived their interest in studying CS increase after PI. More than a third of the students (42% unit testing, 55% integration testing, and 61% continuous integration) gained question correctness through the PI process. We posit that the enjoyment of a lively group discussion that leads to a change of answers, possibly leading to the right answer more often than not, could be a contributor to this shift.

Students' overall confidence levels showed a significant increase as well, and we attribute this to the real time small group communication and feedback mechanism that PI employs, empowering the student with every question. Student perceptions on the gender construct indicated a shift towards a belief of women being as capable and as competent in CS as men. Fruitful PI discussions with groups with students belonging to all genders could be a factor in this perception shift. However, these are new results, and the role of online PI in replicating these shifts needs further research.

6.3 Correctness gains and college standing

Based on their academic progression through the curriculum, the sophomores+juniors group was likely to comprise of students taking this course for the first time, while the seniors group was likely to be weaker students. It is significant that we found correctness increases across all topics and constructs in the cognitive survey, and positive shifts in all constructs of the affective survey, in both groups, after online, synchronous PI. These results reinforce the findings by Zingaro [33] regarding PI with strong and weak students.

7 THREATS TO VALIDITY

It has been established that it might take more than one term to "train students" to be effective learners in a PI classroom [30]. In the context of this study, the students in both groups were not exposed to PI in their previous academic courses. The extent students were able to stay invested in the process of PI during the online course, and the lack of physical presence in a classroom could be factors in utilizing the full potential of PI. Another threat to the validity of this study is the fact that we collected data from a single course, across two iterations, with a moderate number of students in each iteration (treatment n=60, control n=50). Finally, students in both groups could have been exposed any of the topics studied in their prior industry internships. We intend to address these issues in future work.

8 CONCLUSION

While there are studies exploring the relationship between PI in the classroom and student performance in computing, relatively little previous work has been done to evaluate the efficacy of PI in online software engineering classrooms. In this paper, we have presented findings from a quasi-experimental replication study in SE, focused on the application of the PI pedagogy in teaching unit testing, integration testing and continuous integration in an online environment. We saw significant score gains in the PI class in all three areas, and maximum gains in integration testing and continuous integration. The online lecture class did not exhibit learning gains that were statistically significant. In this context, our results strongly suggest that online PI has helped the students learn unit testing, integration testing and continuous integration better than pure online lecture alone. Our results also suggest that students have an improved view of their perceptions about females and males being equally capable of performing well in CS. In addition, their confidence and perception of the usefulness, and professional value of CS improved from before the PI intervention.

In response to RQ1: We found that PI does help undergraduate students in a synchronous online software engineering class learn the concepts of unit testing, integration testing and continuous integration better than their counterparts in a purely lecture format class. This was true for both the sophomores+juniors who were taking the class "on-time" as well as the seniors who were taking the class behind schedule.

In response to RQ2: We found that there was an overall increase in enjoyment of students learning CS, with specific perception shifts observed in all constructs of the affective survey.

Some areas for future work involve addressing questions such as: What are student perceptions regarding online PI? What role did the online environment play in attaining the cognitive and affective gains we found in this study? What role, if any, did industry internships play in these gains? How does PI impact remedial SE education? We hope to address some of these questions in our future work and wish to emphasize the need for extensibility and repeatability to verify our findings from this study.

REFERENCES

[1] R. Agarwal, S.H. Edwards, and M. Pérez-Quiñones. 2006. Designing an adaptive learning module to teach software testing. In *Proceedings of the 37th SIGCSE Technical Symposium on Computer Science Education*. 259–263.

[2] M. Aniche, F. Hermans, and A. Deursen. 2019. Pragmatic software testing education. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education (SIGCSE '19), ACM*. New York, NY, USA, 414–420.

[3] A. Bertolino. 2007. Software testing research: Achievements, challenges, dreams. In *Future of Software Engineering (FOSE'07)*. 85–103.

[4] K. Buffardi and S.H. Edwards. 2014. A formative study of influences on student testing behaviors. In *Proceedings of the 45th ACM Technical Symposium on Computer Science Education*. 597–602.

[5] Z. Chen, J. Zhang, and B. Luo. 2011. Teaching software testing methods based on diversity principles. In *2011 24th IEEE-CS Conference on Software Engineering Education and Training (CSEE T.* Honolulu, HI, USA, 391–395.

[6] P.J. Clarke, J. Pava, D. Davis, F. Hernandez, and T.M. King. 2012. Using WRESTT in SE courses: An empirical study. In *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education (SIGCSE '12), ACM*. New York, NY, USA, 307–312.

[7] Q. Cutts, S. Esper, and B. Simon. 2011. Computing as the 4th "R": A general education approach to computing education. In *Proceedings of the 2011 International Workshop on Computing Education Research*. 133–138.

[8] J. Drake and J. Drake. 2003. Teaching software testing: Lessons learned. *DOI=(http://www. micSymposium. org/mics_2003/Drake.PDF)*.

[9] B.P. Eddy, N. Wilde, N.A. Cooper, B. Mishra, V.S. Gamboa, K.S. Shah, A.M. Deleon, and N.A. Shields. 2017. A pilot study on introducing continuous integration and delivery into undergraduate software engineering courses. In *IEEE 30th Conference on Software Engineering Education and Training (CSEE&T)*. 47–56.

[10] S. H. Edwards and Z. Shams. 2014. Do student programmers all tend to write the same software tests?. In *Proceedings of the 2014 Conference on Innovation & Technology in Computer Science Education*. 171–176.

[11] S. Elbaum, S. Person, J. Dokulil, and M. Jorde. 2007. Bug hunt: Making early software testing lessons engaging and affordable. In *29th International Conference on Software Engineering (ICSE'07)*. IEEE, 688–697.

[12] L. Englund, F. Moosvi, and I. Roll. 2021. Interface and interaction design for an online, asynchronous peer instruction tool. *Interactive Learning Environments*, 1–21.

[13] S. Esper. 2014. A discussion on adopting peer instruction in a course focused on risk management. *Journal of Computing Sciences in Colleges* 29, 4, 175–182.

[14] V. Garousi and J. Zhi. 2013. A survey of software testing practices in Canada. *Journal of Systems Software* 86, 1354–1376.

[15] B. Gopal and S. Cooper. 2021. Peer instruction in software engineering – Findings from fine grained clicker data. In *Proceedings of the 52nd SIGCSE Technical Symposium on Computer Science Education*. 476–480.

[16] B. Gopal and S. Cooper. 2021. Peer instruction in software testing and continuous integration. In *Proceedings of the 52nd SIGCSE Technical Symposium on Computer Science Education*. 255–260.

[17] L. Greising, A. Bartel, and G. Hagel. 2018. Introducing a deployment pipeline for continuous delivery in a software architecture course. In *Proceedings of the 3rd European Conference of Software Engineering Education*. 102–107.

[18] A. Hoegh and B.M. Moskal. 2009. Examining science and engineering students' attitudes toward Computer Science. In *2009 39th IEEE Frontiers in Education Conference*. 1–6.

[19] iClicker. 2021. *Student response systems & classroom engagement tools.* https://www.iclicker.com/ Last Accessed July 2021.

[20] Association for Computing Machinery (ACM) Joint Task Force on Computing Curricula and IEEE Computer Society. 2013. *Computer Science Curricula 2013: Curriculum guidelines for undergraduate degree programs in computer science*. Association for Computing Machinery, New York, NY, USA.

[21] E.L. Jones. 2001. Integrating testing into the curriculum — Arsenic in small doses. In *Proceedings of the 32nd SIGCSE Technical Symposium on Computer Science Education*. 337–341.

[22] C.B. Lee, S. Garcia, and L. Porter. 2013. Can peer instruction be effective in upper-division Computer Science courses? *Trans Comput Educ* 12, 1–12.

[23] S. Liao, W. Griswold, and L. Porter. 2017. Impact of class size on student evaluations for traditional and peer instruction classrooms. In *Proceedings of the ACM SIGCSE Technical Symposium on Computer Science Education, ACM*. 375–380.

[24] E. Mazur. 1997. *Peer instruction a user's manual*. Prentice Hall.

[25] C. Nerantzi. 2020. The use of peer instruction and flipped learning to support flexible blended learning during and after the COVID-19 pandemic. *International Journal of Management and Applied Research* 7, 2, 184–195.

[26] A. Orso and G. Rothermel. 2014. Software testing: A research travelogue (2000–2014). In *Proceedings of the on Future of Software Engineering, ACM*. 117–132.

[27] L. Porter, C.B. Lee, B. Simon, Q. Cutts, and D. Zingaro. 2011. Experience report: A multi-classroom report on the value of peer instruction. In *Proceedings of the 16th Annual Joint Conference on Innovation and Technology in Computer Science Education, ACM*. 138–142.

[28] R. Ravid. 2019. *Practical statistics for educators*. Rowman & Littlefield Publishers.

[29] B. Simon, S. Esper, L. Porter, and Quintin Cutts. 2013. Student experience in a student-centered peer instruction classroom. In *Proceedings of the 9th Annual International ACM Conference on International Computing Education Research*. 129–136.

[30] B. Simon, J. Parris, and J. Spacco. 2013. How we teach impacts student learning: Peer instruction vs. lecture in cs0. In *Proceeding of the 44th ACM Technical Symposium on Computer Science Education, ACM*. 41–46.

[31] J. Süß and W. Billingsley. 2012. Using continuous integration of code and content to teach software engineering with limited resources. In *34th International Conference on Software Engineering (ICSE)*. IEEE, 1175–1184.

[32] W. E. Wong, A. Bertolino, V. Debroy, A. Mathur, J. Offutt, and M. Vouk. 2011. Teaching software testing: Experiences, lessons learned and the path forward. In *2011 24th IEEE-CS Conference on Software Engineering Education and Training (CSEE&T)*. IEEE, 530–534.

[33] D. Zingaro. 2010. Experience report: Peer instruction in remedial computer science. In *EdMedia+ Innovate Learning*. Association for the Advancement of Computing in Education (AACE), 5030–5035.