2023 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW) | 979-8-3503-3335-0/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICSTW58534.2023.00080

Process Oriented Guided Inquiry-based learning -like pedagogy (POGIL-like) in Online Software Testing and DevOps – A Replication Study

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

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

***Abstract*—We present in this paper our findings from replicating our previously published study of a process oriented guided inquiry based learning approach called POGIL-like in the topics of unit testing, integration testing and continuous integration taught in an undergraduate, sophomore/junior level software engineering course. In our original paper we employed POGIL-like in an in-person classroom. In this replication study we discuss in detail the implementation of POGIL-like in a fully virtual, synchronous online classroom. We found a positive correlation between POGIL-like and student learning. We measured student learning outcomes using a cognitive pre- and post- survey instrument and observed statistically significant increases in learning outcomes. Our results reinforce the findings from our original study, and establish POGIL-like as a suitable pedagogical approach for teaching software testing and DevOps online at the undergraduate level.**

***Index Terms*—Software Testing Education, DevOps Education, Unit Testing Education, Integration Testing Education, Continuous Integration Education, POGIL-like, guided inquiry based learning, Active Learning, Online Learning**

I. Introduction

The importance of teaching software testing has been well understood by and established in literature. Unit testing, integration testing and continuous integration are typically some of the first topics that novice software engineers are introduced to in industry jobs. Active learning is fast gaining popularity among researchers and educators alike, to help students better understand the content they are taught. While there are several active learning pedagogies being employed in CS education, our specific interest in this study was to study how a process oriented, guided, inquiry based learning approach based on POGIL [1] affected cognitive and affective learning gains in the topics of unit testing, integration testing and continuous integration, which were taught as part of an undergraduate, mid level software engineering course. We recognize and wish to capitalize on the paradigm shift that the Covid-19 pandemic has brought about with online instruction, and use the opportunity of the pandemic to investigate if online POGIL-like helped students learn better than a traditional online lecture class.

Undergraduate SE courses are great opportunities to introduce to and train students in these topics before these students transform into industry professionals. However, teaching software testing is not an easy task [2]. Learning software testing presents some challenges to students, which we explored in a previous study [3]. Students often struggle with identifying where to start and finish testing, recognizing what to test, and correlating requirements to test cases [3].

To help mitigate some of these difficulties, educators have started exploring non-transmissionist pedagogies with the focus being on the student and their participation during learning, playing an active, real-time role in the classroom. POGIL [4] is a teaching and learning approach involving students working together in small teams, with a specific process and roles. POGIL studies in entry level CS courses show an improvement in student learning [4]. ”POGIL” is a trademarked term, only used by the POGIL project and forbidden for general use [5]. We use the term ”POGIL-like” to denote our implementation of POGIL in this paper.

We organized the paper as follows: Section 2 explains prior work, Section 3 poses the research question that we investigated. In Section 4 we introduce the survey design and methodology. We highlight the salient features of the POGIL-like pedagogy in Section 5. Results follow in Section 6, which we discuss in Section 7, and explore potential threats to the validity of our study in Section 8. Section 9 concludes the paper.

II. Related literature

Student-centric cooperative and active learning approaches to teaching have shown to have advantages over traditional lecture [6]. Pair programming [7], [8], ”flipped classroom”, and Peer Instruction [9]–[12] are all approaches that are now gaining popularity in CS education.

A. Teaching software testing and DevOps using active and collaborative learning other than POGIL

Specifically in teaching software testing using active learning approaches, we have published our work utilizing Peer

979-8-3503-3335-0/23/$31.00 ©2023 IEEE
DOI 10.1109/ICSTW58534.2023.00080

Instruction to teach unit testing, integration testing and continuous integration [13] both online and in regular in person classrooms. Non-lecture based approaches to teaching software testing have been gaining popularity. Problem-Based Learning (PBL) and Just-in-time-teaching (JiTT) were some of the approaches used [14] along with peer review [15] and game-based software testing [16]. These approaches show that researchers have begun focusing on evidence-based ways to teach STEM courses [17], [18].

B. Prior work in POGIL in STEM including CS

POGIL has its origins in chemistry and biomechanics [19]–[23]. College general chemistry courses have adopted POGIL to improve student learning outcomes and to reduce drop out and failure rates [19], [24], [25]. Studies on POGIL explore various aspects such as student grades, student confidence [26], how to facilitate POGIL classes [21], large classrooms and efficiency of POGIL [20], and the role of instructor intervention [27]. POGIL in CS has been introduced to the scientific community through important work by Kussmaul [4] and Hu [28].

There are unique challenges to using POGIL in CS [4]. Significant time has to be invested by faculty wishing to adopt POGIL. CS curricular content undergoes rapid change, and the POGIL activities have to keep up with it, thus making dissemination and sharing difficult. The CS-POGIL project has published activities for CS1 and CS2 classes [4], [4], [28]–[30]. POGIL activities for project management and other non-programming related aspects of SE are also available on the POGIL website [31].

Furman University professors used inquiry-based learning, with a slight modification from POGIL, in five introductory CS classes [32]. Medley [33] compared inquiry based learning with lectures, again with a slight variation in classic POGIL. The majority of studies about POGIL in CS are in CS1 [28], [34], [35]. All the above studies found higher pass rates or improved student grades for the inquiry-based group compared to lectures. There have been POGIL activities developed for specific subtopics in cybersecurity [36]–[38] such as access control, software defined network data plane flooding attacks,firewall and IPsec. These studies noted improved student performance. There is also one study on faculty affect, and difficulties [39].

C. POGIL in SE

Kussmaul has developed POGIL activities for software engineering [4], with ”structured like POGIL” activities by other researchers [40]. The focus of these studies has been on helping students learn process skills like problem-solving, communication and teamwork [41]. There has also been one study on utilizing POGIL to teach design patterns [42].

D. Online collaborative and inquiry based learning

Online POGIL is a nascent topic. Reynders and Ruder [43] studied how to move a large organic chemistry course from in-person to online POGIL instruction. Their emphasis was on altering the support system for the course structure that involved teaching assistants and training them.

Most of the SE topics studied with POGIL are mainly process based - project scheduling, software development and release life cycles, story point estimation, and version control, to name a few. When considering how POGIL is used to teach technical content, we published our study of POGIL-like in unit testing, integration testing and continuous integration in an in-person, sophomore/junior SE classroom [44]. We examined the efficacy of POGIL-like by utilizing pre- and post- survey questionnaires consisting of cognitive and affective components.

Barr [45] studied POGIL in an online software engineering course, and specifically found that group activities did not go well over breakout rooms on Zoom. Students suffered from missed opportunities for learning from each other since many students worked on their own instead of working with their assigned group. A study by Trevathan [46] investigated how to implement a condensed version of online POGIL in IT courses. They proposed a preliminary teaching framework using social networking, blogs and a wiki.

As we can see above, there are some POGIL and active learning studies published in other areas of computer science and software engineering, but there is a dearth of research utilizing POGIL in the areas of software testing and DevOps. Online implementation of POGIL, in particular, is a new and yet largely unexplored area of study. Our work in this paper aims to combine the relative scarcity of active learning studies in software testing and DevOps, with the much needed online education paradigm which has shaped how we teaching during and post the Covid-19 pandemic.

III. Research Question

Our primary objective in this online replication study was to investigate whether POGIL-like influenced student learning outcomes for unit testing, integration testing and continuous integration. Our research question is:

RQ: Does the POGIL-like pedagogy help students learn the topics of unit testing, integration testing and continuous integration better than students taught through lecture in a synchronous, online class?

IV. Methods

A. Study Context

The instructor of the course was trained in POGIL activity writing and facilitation having participated in several POGIL workshops conducted through the POGIL project [47], including an activity writing workshop. Our original in person classroom study, utilized 4 POGIL-like activities we created - 2 in unit testing, and 1 each in integration testing and continuous integration [44], and we utilized the same activities for this replication study. Our study was reviewed and granted exempt certification through the Institutional Review Board of our university.

At the beginning of the semester, we administered a survey questionnaire to the students in both the online lecture and

online POGIL-like groups using a cognitive questionnaire with 16 multiple choice questions that we published in our previous study [44]. The questionnaire contained 6 questions in unit testing, 3 in integration testing and 7 in continuous integration. At the end of the semester, students took the survey again, after undergoing instruction online using either POGIL-like or lecture, depending on their section. Student participation in the surveys was purely optional and no compensation was provided.

B. Student and instructor roles

Student roles: Each student played a specific role in the group for each POGIL-like online session [28]: Manager, Recorder, Presenter and Reflector. More details on the specific functions of each role can be found in our previous study detailing an in person implementation of POGIL-like [44]. Roles were rotates through sessions to give each student experience with each role. We based our implementation of POGIL on the details in earlier POGIL CS studies [4], [24], [34], but adapted to the online environment. In our online classroom the instructor spent two minutes rotating through each breakout room, to listen in and be available for questions from the group. Students raised the hand using the online feature on Zoom when they needed the instructor's attention.

Experiment groups: Our study consisted of two sections of students in a sophomore/junior level SE course at a large R1 university. The control group (CG), was taught online using lectures. There were 59 students in this section, taught in the Fall of 2021. The treatment group (TG), utilized the online POGIL-like pedagogy. There were 54 students in this section, taught in the Spring of 2022. There was no difference in the content order or instructor for either of the sections. We utilized a quasi-experimental, pre- post- test study design. We analyzed the variance in our data within and between both groups. The majority of our students were Computer Science and Computer Engineering majors.

Online Class sessions: Each iteration of the course consisted of 16 weeks, with 3 days of class sessions each 50 minutes long, all conducted synchronously online through Zoom. We implemented the POGIL-like pedagogy identical to what we describe in our original study [44] with 2 sessions in unit testing, 1 for integration testing, and 1 for continuous integration. All classes were conducted synchronously online. Attendance was taken during each class session. Students were awarded 1% of their total grade based on their attendance and engagement in the POGIL-like sessions. Engagement was gauged by the artifacts produced by the reflector, as well as how students actively participated during discussions in breakout rooms.

Before each class session: In keeping with the constructivist roots of POGIL-like, we ensured that students were not given any prior preparatory material to work with. We hoped that this would make them focused on exploring and learning with a fresh perspective on the topic being studied.

During class: Students were assigned in groups of 4, to breakout rooms over Zoom, and worked on POGIL-like activities crafted according to specific guidelines based on E-I-A cycles and D/C/V questions [4]. Students studied the model together, worked on activities, discussing their reasoning for the answers as they went along, arguing for or against specific opinions and answer choices, recording their observations and ideas, and finally sharing them with the instructor.

V. Salient features of the online POGIL-like approach

POGIL-like consists of classroom activities that are aimed at aligning with how students learn. Bauer [48] emphasizes that learning is a constructive process. POGIL-like facilitates knowledge co-construction by understanding how people learn. This approach is supported by theories rooted in learning science [49], just like traditional POGIL. The basic premise is that skills and knowledge are developed and constructed by the learner, not by simple transfer of knowledge from the teacher to the student. POGIL-like combines learning theories into a small group setting with specific and well defined processes and roles, to facilitate cooperative learning.

POGIL-like is built on collaborative learning by taking advantage of student strengths in a small group setting, by guiding them through a process heavy, prescribed set of roles and steps [4]. The full details of the in person implementation of POGIL-like can be found in our study [44] on software testing and DevOps education.

In our online implementation POGIL-like pedagogy, students are organized into small teams and each team is assigned to a breakout room on Zoom [50]. All other aspects of the implementation, regarding the enforcement of a "clean slate" with little to no prior knowledge of the topic, to team sizes of 4 students each, to the various POGIL-like student roles, and the POGIL-like models and activities, remain identical to the original, in person implementation [44]. The various POGIL roles we used were rotated through each student in each group in the breakout rooms. For example, student A might have been a reflector for session 1, but will assume a different role, perhaps that of a manager, in session 2. We preserved the same breakout rooms for all 4 POGIL-like sessions we studied, so that students could rotate roles with ease. In creating models and activities, we took care to follow guidelines of how to incorporate E-I-A cycles and D/C/V questions according to the guidelines provided by previous POGIL studies [4], [28], [41], [44].

A. Developing POGIL-like Activities

There are three pillars of POGIL-like activity development - D/C/V question types, E-I-A cycles, and Models. Models are typically the equivalent of lecture slides, but are presented with less text and more graphics/figures/tables and non-text content. The goal in developing models is to avoid large blocks of pure textual content. Each model would be followed by one or more activities, that are exercises that student groups would work on together in breakout rooms. Each activity would contain a series of questions. Questions used for POGIL-like activities would ideally advance student understanding through

exploration (E), concept invention (I) and application (A), fulfilling one or more E-I-A cycles [51]. These questions could be simple, direct knowledge questions, with answers obvious from the models (directed questions, denoted by D). They could combine two or more concepts (convergent questions, denoted by C); or they could challenge the students to visualize the concept in a context completely different from where it was originally presented (divergent questions, denoted by V). We skilfully combined D,C and V question types to create our POGIL-like activities.

Each online POGIL-like session was designed to achieve a set number of process skills (also called process learning objectives or PLOs), and a well defined set of content skills (content learning objectives or CLOs). Each session was designed to be completed in 35-40 minutes, so that in a 50 minute class session, students could have enough time for the report-out process in the general Zoom room. Two example POGIL-like model/ activity combinations (instructor's version, with the solution) are shown in the next section. Student versions would be identical except for the solutions and instructor's notes.

B. POGIL-like in the online classroom: an example activity

Figure 1 shows the set of instructions the students are given before they begin their online POGIL session. The content and process learning objectives are clearly delineated, along with roles.

Continuous Integration *July 2021* *Bhuvana Gopal*

CONTINUOUS INTEGRATION POGIL SESSION

CLO: Identify causes for merge conflicts, merge hell/integration hell; compare and contrast traditional CI vs Gated CI; identify reasons for failed CI builds PLO: Information processing, Problem Solving, Teamwork, Critical Thinking, Communication	start time:

Prerequisites: Successful completion of all questions in Unit Testing POGIL sessions 1 and 2, Integration Testing POGIL session.

Before you start, complete the form below to assign a role to each member.
If you have 3 people, combine Speaker & Reflector.

Team	**Date**
Team Roles	**Team Member**
Recorder: records all answers & questions, and provides copies to team & facilitator.	
Speaker: talks to facilitator and other teams.	
Manager: keeps track of time and makes sure everyone contributes appropriately.	
Reflector: considers how the team could work and learn more effectively.	

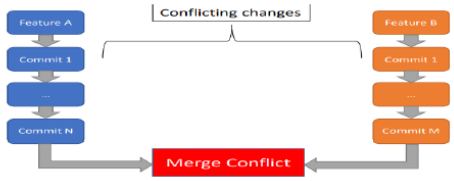
Reminders:

1. *Note the time whenever your team starts a new section or question.*
2. *Write legibly & neatly so that everyone can read & understand your responses.*

Fig. 1. Introductory instructions

Figure 3 and Figure 2 show example excerpts from an E-I-A cycle online for the subtopics of integration hell (also known as merge hell), and gated continuous integration.

MODEL 3:



ACTIVITY:

1) In the model above, which commit has the conflicting changes for features A and B?
   Answer: This is an E/D question, the answer is Commit 1.

2) Look at the model above and identify the scenario where "integration hell" could occur. Specifically, what could go wrong with Commit M and Commit N occurring in the timeline shown?

Answer:
This is a I/C question.

Fig. 2. Model and associated activity for merge hell

In Figure 2 we find the D/C/V progression that leads to an E-I-A cycle. The first question is a D question as part of the E portion of the learning cycle, and simply asks the student group to identify a specific commit from the diagram. The second question, however, asks the student group to reason further and explain potential pitfalls in the commit pipeline within the given timeline.

The above model/activities show how D/C/V questions were used to achieve the E-I-A POGIL-like cycle. We see that students had to build up from simple questions with answers easily identifiable from the model (D questions), to questions that required increasing levels of synthesis and analysis (using C/V questions). We moved through the idea that code does not get committed in a gated CI right away, through concept invention (a reason why one type of CI might be better than the other), implementing the E-I portion of the E-I-A cycle, with D and V type questions from the D/C/V types.

VI. Results

A. Data Analysis Techniques

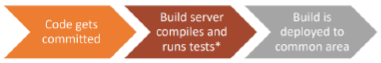
Our primary data points were the pre- and post- cognitive surveys. We analyzed these data to see if the control and treatment groups had statistically significant score differences (between groups set up). We also analyzed the treatment group by itself (within groups set up). Normalcy of data was tested using the Shapiro-Wilk test, combining skew and kurtosis. We used Levene's test to check for homoscedasticity and homogeneity of variance across all values of the independent variables. We used appropriate ANOVA tests (regular or Welch's) and p-value to determine if the results were statistically significant [52]. We use a p-value of less than 0.05 to indicate statistically significant results. There were 59 students in the treatment group (online POGIL-like) and 54

MODEL 4:

Take a look at the diagrams below:

Continuous Integration *July 2021* *Bhuvana Gopal*

TRADITIONAL CI:

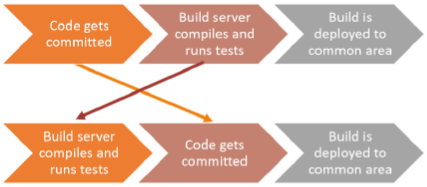


- Build is deployed regardless if tests pass or fail

*integration tests only

GATED CI:

- Run the tests first



Stop the code from being committed to the release branch, verify it offline on a copy of the most recent release code, and pass it back only if the verification is successful. This type of solution is called gated check-in, and it comes in several different flavors and implementations. You can find existing gated check-in solutions out there: Gerrit, Microsoft's Team Foundation Server (TFS); Zuul; and Electric Cloud's Preflight solution are all examples, and each has its pros and cons. For example, Gerrit is widely used in the open source community, as it integrates into Git and Jenkins, and Microsoft's TFS is built in with the entire development environment.

ACTIVITY:

1) In gated CI, does code get committed first?

Answer: This is a E/D question, and no.

2) Identify one reason why gated builds might be better than traditional CI.
Answer:
This is a I/V question.

Continuous Integration *July 2021* *Bhuvana Gopal*

3)
What types of tests do you think would be best suited for CI? Unit tests, or integration tests, or both? In what order?

Answer:
This is also an I/V question.

Fig. 3. Model and associated activity for gated CI

students in the control (online lecture) group. Our data were normally distributed (p value = 0.13).

B. Starting points

Both the online lecture class and the online POGIL-like class had starting points that were not dissimilar, as evidenced by their pre-test ANOVA p values - Unit testing (pre-test: χ^2 = 0.54, p = 0.15), integration testing (χ^2 = 0.53, p = 0.21) and continuous integration (χ^2 = 0.21, p = 0.16). The pre-survey scores for all three topics combined, for both groups (including all topics) was similar as well (χ^2 = 0.28, p = 0.12).

C. Between groups ANOVA results

The online POGIL-like group had significantly higher post survey scores than those for the online lecture group: (χ^2 = 63.15, p $<$ 0.0001). ANOVA test results for each topic were: unit testing (χ^2 = 34.26, p $<$ 0.001); integration testing (χ^2 = 42.93, p $<$ 0.001); and continuous integration (χ^2 = 41.13, p $<$ 0.001). Figure 4 shows the overall score distribution in the post-online POGIL-like (treatment) group and post- online lecture(control) group.

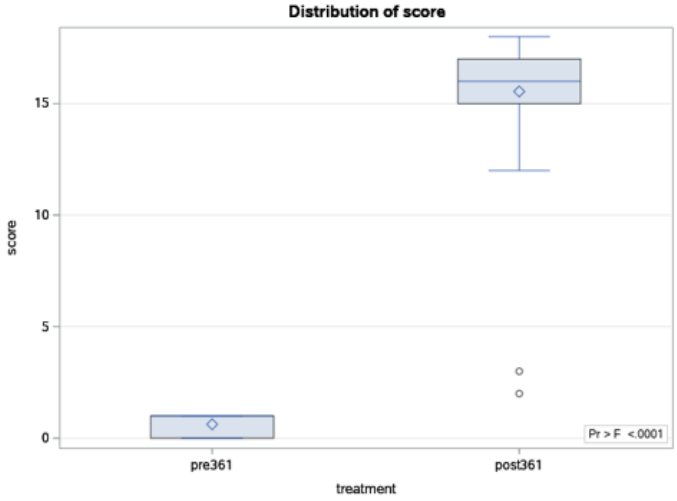


Fig. 4. Online POGIL-like vs online lecture, post survey - cognitive

D. Within groups ANOVA results

The online POGIL-like class (treatment group) exhibited a statistically significant difference in their post survey scores compared to the pre survey, indicating that they learned from the online pedagogical intervention: (χ^2 = 28.97, p $<$ 0.0001). The ANOVA results for each category, at an α level of 0.05 are: unit testing (χ^2 = 40.14, p $<$ 0.001); integration testing (χ^2 = 39.13, p $<$ 0.001); and continuous integration (χ^2 = 34.17, p $<$ 0.001). Figure 5 shows the overall score distribution for the treatment group, pre- and post- survey.

VII. Discussion

Our results indicate that students in the online POGIL-like group learned better than the online lecture group, in all the three topics of unit testing, integration testing and continuous integration. In addition to this being an encouraging result, it prompts us to look deeper into the potential reasons for the success of the online POGIL-like implementations.

A. Group formation and instructor involvement

Some of the key details that we think helped students stay motivated through each online breakout room session are the well defined roles, and the instructor periodically checking in with each room. The clearly defined task set

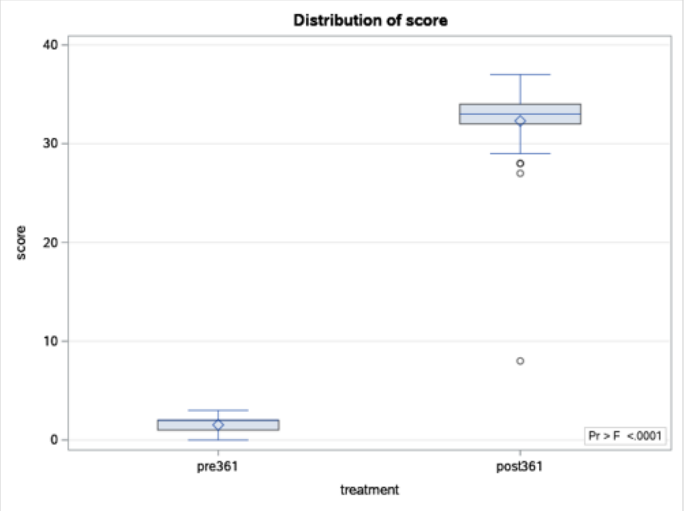


Fig. 5. Online POGIL-like: pre vs post survey - cognitive

before the students with specific activities that followed each model, reinforce the findings by Barr [45], who emphasized clear set goals as a way to combat student isolation online. Students were allowed the freedom to interpret their specific roles individually; reflectors, for example, utilized various techniques to reflect on their group's session - some reflectors took notes, some recorded voice memos, and some created diagrams. All of these were shared with the instructor as she circulated through the breakout rooms periodically. Barr [45] observed in their study that group dynamics are a constant factor in teaching SE. We took particular care to mitigate potential clashes within students in each group through an online poll where we collected student preferences before assigning groups. Our poll specifically asked each student who they preferred to not work with. Based on the results of the poll, we formed our POGIL-like groups. Each POGIL-like group stayed the same throughout the set of our four online POGIL-like class sessions. We think that this setup allowed students to familiarize themselves with the approach better each class session, without having to work through potential group dynamics issues. POGIL-like session attendance and engagement (as determined by the random visits conducted by the instructor through each breakout room) was allotted 1% point of the total grade. We think that this strategy could have motivated students to be present and engaged during POGIL-like discussions in the breakout rooms.

B. Developing questions for online POGIL-like sessions

Developing the right number of questions in an appropriate sequence and combination of directed, convergent, and divergent questions is an involved process that takes training and practice. Students may lose interest in the task if there are too many questions. If there are too few, students may miss important connections between concepts. Question difficulty also matters. If the activity questions are too easy, students might become complacent. If questions are too difficult, students might get discouraged and give up. While developing questions, our priority was to help students think deeper, and more critically. We wanted students to progress from simple, obvious answers, to more open ended and involved thinking that would help them reach higher levels of Bloom's Taxonomy [53]. Our activity questions were designed with these factors in mind and comprised of a balance between easy, difficult, direct, and open ended questions.

We surmise that the effort we put into setting up a productive and welcoming online classroom, along with the thought and scholarly approach to group formation and question development, played a part in keeping students engaged during the online POGIL-like sessions, leading to the statistically significant cognitive score gains we observed.

VIII. Threats to Validity

In any study involving students, there are unknown factors that could influence classroom performance. Combining the online environment, which in itself has been shown to be stressful to students [45], with a potentially unfamiliar pedagogical approach such as POGIL-like, could be a factor that affects student motivation [54]. Even with the best, carefully thought out group formation and facilitation efforts by the instructor (who was trained in POGIL facilitation), it is difficult to determine if students committed to adhering to the process oriented elements of the pedagogical approach. Without specifically studying each student group's discussions on each POGIL-like session, it is difficult to know if students did not succumb to a herd mentality, with one or two students leading the discussion and other students not participating. In addition, with the implementation setup of POGIL-like in an online setting, it is difficult for the instructor to be available as much as in an in-person setting. What effect this might have had on student affect, is yet to be understood.

Another threat to the validity of this study is the sample size and monolithic nature of our data, even though our data were normally distributed. Our data were collected from a single course taught by the same instructor over online lectures and online POGIL-like over two semesters. It would be interesting to investigate if our online pedagogical approach is replicable with other courses and instructors.

IX. Conclusion

In this paper, we have presented findings from our study of teaching unit testing, integration testing and continuous integration using a process oriented guided inquiry based learning approach, in a fully synchronous, online software engineering class. We saw significant score gains in the online POGIL-like class in all three areas compared to the online lecture class. We conclude that online POGIL-like could positively impact student learning in these topics.

In response to the RQ: We found that the POGIL-like pedagogy does help students learn the topics of unit testing, integration testing and continuous integration better than students taught through lecture in a synchronous, online class.

We have identified several areas for future work. First, we intend to understand how students perceived online POGIL-like through the affective questionnaire we administered [55] along with the cognitive questionnaire. What were the difficulties faced by students in the online POGIL-like environment? How did student discussions vary based on their potential prior knowledge on the topics studied, perhaps through a previous industry internship? How much of their learning was influenced by the combination of specific student roles and discussions? In our future work, we plan to explore some of these topics.

REFERENCES

[1] C. Kussmaul, "Process oriented guided inquiry learning for soft computing," in *International Conference on Advances in Computing and Communications*. Springer, 2011, pp. 533–542.

[2] P. Clarke, J. Pava, D. Davis, F. Hernandez, and T. King, "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, 2012, p. 307–312.

[3] B. Gopal, S. Cooper, J. Olmanson, and R. Bockmon, "Student difficulties in unit testing, integration testing and continuous integration: An exploratory pilot qualitative study."

[4] C. Kussmaul, "Process oriented guided inquiry learning (POGIL) for Computer Science," in *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education*, 2012, pp. 373–378.

[5] "What is Process-Oriented Guided Inquiry Learning." [Online]. Available: https://serc.carleton.edu/sp/pkal/POGIL/what.html

[6] P. L. Machemer and P. Crawford, "Student perceptions of active learning in a large cross-disciplinary classroom," vol. 8, no. 1. Sage Publications London, Los Angeles, New Delhi and Singapore, 2007, pp. 9–30.

[7] C. McDowell, L. Werner, H. E. Bullock, and J. Fernald, "Pair programming improves student retention, confidence, and program quality," vol. 49, no. 8. ACM New York, NY, USA, 2006, pp. 90–95.

[8] C. A. de Lima Salge and N. Berente, "Pair programming vs. solo programming: What do we know after 15 years of research?" in *2016 49th Hawaii International Conference on System Sciences (HICSS)*. IEEE, 2016, pp. 5398–5406.

[9] B. Simon, D. Bouvier, T.-Y. Chen, G. Lewandowski, R. McCartney, and K. Sanders, "Common sense computing (episode 4): Debugging," vol. 18, no. 2. Taylor & Francis, 2008, pp. 117–133.

[10] L. Porter, C. Bailey Lee, B. Simon, Q. Cutts, and D. Zingaro, "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*, 2011, pp. 138–142.

[11] C. B. Lee, "Experience report: CS1 in matlab for non-majors, with media computation and Peer Instruction," in *Proceeding of the 44th ACM Technical Symposium on Computer Science Education*, 2013, pp. 35–40.

[12] M. L. Maher, C. Latulipe, H. Lipford, and A. Rorrer, "Flipped classroom strategies for cs education," in *Proceedings of the 46th ACM Technical Symposium on Computer Science Education*, 2015, pp. 218–223.

[13] B. Gopal and S. Cooper, "Peer instruction in online software testing and continuous integration: A replication study," in *2022 IEEE/ACM 44th International Conference on Software Engineering: Software Engineering Education and Training Track (ICSE-SEET)*. IEEE, 2022.

[14] J. Coutinho, W. Andrade, and P. Machado, "Teaching exploratory tests through pbl and jitt: An experience report in a context of distributed teams," in *Brazilian Symposium on Software Engineering*, 2021, pp. 205–214.

[15] J. Smith, J. Tessler, E. Kramer, and C. Lin, "Using peer review to teach software testing," in *Proceedings of the Ninth Annual International Conference on International Computing Education Research*, 2012, pp. 93–98.

[16] G. Fraser, A. Gambi, M. Kreis, and J. Rojas, "Gamifying a software testing course with code defenders," in *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, 2019, pp. 571–577.

[17] J. Handelsman, D. Ebert-May, R. Beichner, P. Bruns, A. Chang, R. DeHaan, J. Gentile, S. Lauffer, J. Stewart, S. M. Tilghman *et al.*, "Scientific teaching," 2004.

[18] National Research Council *et al.*, *A framework for K-12 science education: Practices, crosscutting concepts, and core ideas*. National Academies Press, 2012.

[19] R. S. Moog and J. N. Spencer, *Process oriented guided inquiry learning*. American Chemical Society Washington, DC, 2008, vol. 994.

[20] C. P. Bailey, V. Minderhout, and J. Loertscher, "Learning transferable skills in large lecture halls: Implementing a pogil approach in biochemistry," vol. 40, no. 1. Wiley Online Library, 2012, pp. 1–7.

[21] S. M. Hein, "Positive impacts using pogil in organic chemistry," vol. 89, no. 7. ACS Publications, 2012, pp. 860–864.

[22] A. Straumanis and E. A. Simons, "A multi-institutional assessment of the use of pogil in organic chemistry." ACS Publications, 2008.

[23] S. R. Simonson and S. Shadle, "Implementing process oriented guided inquiry learning (pogil) in undergraduate biomechanics: Lessons learned by a novice," vol. 14, no. 1, 2013.

[24] D. M. Hanson, "Designing process-oriented guided-inquiry activities," *Pacific Crest*, pp. 1–6, 2005.

[25] S. Kode and J. Cherukuri, "Creating a learner centric environment through pogil: Our experience in engineering and management education in india," in *2014 IEEE Sixth International Conference on Technology for Education*. IEEE, 2014, pp. 72–75.

[26] S. De Gale and L. N. Boisselle, "The effect of pogil on academic performance and academic confidence." International Council of Associations for Science Education, 2015.

[27] P. L. Daubenmire, D. M. Bunce, C. Draus, M. Frazier, A. Gessell, and M. T. van Opstal, "During pogil implementation the professor still makes a difference," vol. 44, no. 5. JSTOR, 2015, pp. 72–81.

[28] H. H. Hu and T. D. Shepherd, "Teaching cs 1 with pogil activities and roles," in *Proceedings of the 45th ACM technical symposium on Computer science education*, 2014, pp. 127–132.

[29] H. Hu and B. Avery, "Cs principles with pogil activities as a learning community," *Journal of Computing Sciences in Colleges*, vol. 31, no. 2, pp. 79–86, 2015.

[30] H. H. Hu, C. Kussmaul, B. Knaeble, C. Mayfield, and A. Yadav, "Results from a survey of faculty adoption of process oriented guided inquiry learning (pogil) in computer science," in *Proceedings of the 2016 ACM Conference on Innovation and Technology in Computer Science Education*, 2016, pp. 186–191.

[31] NSF TUES 1044679 POGIL in Computer Science. (2021) CS-POGIL. Last Accessed July 2021. [Online]. Available: http://cspogil.org/

[32] K. Abernethy, P. Gabbert, and K. Treu, "Inquiry-based computer science instruction: Some initial experiences," *ACM SIGCSE Bulletin*, vol. 30, no. 3, pp. 14–17, 1998.

[33] M. D. Medley, "Inquiry-based learning in cs1," *Journal of Computing Sciences in Colleges*, vol. 23, no. 2, pp. 209–215, 2007.

[34] H. H. Hu and T. D. Shepherd, "Using pogil to help students learn to program," *ACM Transactions on Computing Education (TOCE)*, vol. 13, no. 3, pp. 1–23, 2013.

[35] M. Jonas, "Lessons learned from integrating pogil into a CS1 course," vol. 34, no. 6. Consortium for Computing Sciences in Colleges, 2019, pp. 139–140.

[36] L. Yang, X. Yuan, W. He, J. Ellis, and J. Land, "Cybersecurity education with pogil: Experiences with access control instruction," in *Journal of The Colloquium for Information Systems Security Education*, vol. 6, no. 2, 2019, pp. 14–14.

[37] H. Alshaher, X. Yuan, and S. Khorsandroo, "Teaching flooding attack to the sdn data plane with pogil," in *Proceedings of the 21st Annual Conference on Information Technology Education*, 2020, pp. 194–199.

[38] X. Yuan, L. Yang, W. He, J. Ellis, J. Xu, and C. Waters, "Enhancing cybersecurity education using pogil," in *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education*, 2017, pp. 719–719.

[39] A. Yadav, C. Kussmaul, C. Mayfield, and H. H. Hu, "Pogil in computer science: faculty motivation and challenges," in *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, 2019, pp. 280–285.

[40] B. Coleman and M. Lang, "Collaboration across the curriculum: a disciplined approach todeveloping team skills," in *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education*, 2012, pp. 277–282.

[41] C. Kussmaul, C. Mayfield, and H. Hu, "Process oriented guided inquiry learning in computer science: The cs-pogil & introcs-pogil projects," in *ASEE Annual Conference and Exposition, Conference Proceedings*, 2017, pp. 1–7.

[42] P. C. Lotlikar and R. Wagh, "Using pogil to teach and learn design patterns—a constructionist based incremental, collaborative approach," in *2016 IEEE Eighth International Conference on Technology for Education (T4E)*. IEEE, 2016, pp. 46–49.
[43] G. Reynders and S. M. Ruder, "Moving a large-lecture organic pogil classroom to an online setting," *Journal of Chemical Education*, vol. 97, no. 9, pp. 3182–3187, 2020.
[44] B. Gopal and S. Cooper, "Pogil-like learning in undergraduate software testing and devops-a pilot study," in *Proceedings of the 27th ACM Conference on on Innovation and Technology in Computer Science Education Vol. 1*, 2022, pp. 484–490.
[45] M. Barr, S. W. Nabir, and D. Somerville, "Online delivery of intensive software engineering education during the covid-19 pandemic," in *2020 IEEE 32nd Conference on Software Engineering Education and Training (CSEE&T)*. IEEE, 2020, pp. 1–6.
[46] J. Trevathan and T. Myers, "Towards online delivery of process-oriented guided inquiry learning techniques in information technology courses," *Journal of Learning Design*, vol. 6, pp. 1–11, 2013.
[47] "CS-POGIL | DCV (Directed, Convergent, Divergent) Questions." [Online]. Available: https://csPOGIL.org/DCV+(Directed,+Convergent,+Divergent)+Questions
[48] T. N. Bauer and B. Erdogan, "Organizational socialization: The effective onboarding of new employees." American Psychological Association, 2011.
[49] R. S. Moog and J. N. Spencer, "Pogil: An overview." ACS Publications, 2008.
[50] Zoom. (2021) Video conferencing, web conferencing, webinars, screen sharing. Last Accessed July 2021. [Online]. Available: https://zoom.us/
[51] M. Hu, M. Winikoff, and S. Cranefield, "Teaching novice programming using goals and plans in a visual notation," in *Proceedings of the Fourteenth Australasian Computing Education Conference-Volume 123*, 2012, pp. 43–52.
[52] R. Wasserstein and N. Lazar, "The asa statement on p-values: Context, process, and purpose," vol. 70, 2, p. 129133.
[53] B. S. Bloom, *Taxonomy of educational objectives, Handbook 1: Cognitive domain*, 2nd ed. New York, NY, USA: Addison-Wesley Longman Ltd.
[54] B. Simon, P. Kinnunen, L. Porter, and D. Zazkis, "Experience report: CS1 for majors with media computation," in *Proceedings of the Fifteenth Annual Conference on Innovation and technology in Computer Science Education*, 2010, pp. 214–218.
[55] A. Hoegh and B. Moskal, "Examining science and engineering students attitudes toward computer science," in *2009 39th IEEE Frontiers in Education Conference*, p. 16.