Experiences of Integrating Learning and Engagement Strategies (LESs) into Software Engineering Courses

Dr. Peter J Clarke, Florida International University

Peter J. Clarke received his B.Sc. degree in Computer Science and Mathematics from the University of the West Indies (Cave Hill) in 1987, M.S. degree from SUNY Binghamton University in 1996 and Ph.D. in Computer Science from Clemson University in 2003. His research interests are in the areas of software testing, software metrics, model-driven software development, domain-specific modeling languages, and computer science education. He is currently an associate professor in the School of Computing and Information Sciences at Florida International University. He is a member of the ACM (SIGSOFT, SIGCSE, and SIGAPP); IEEE Computer Society; and a member of the Association for Software Testing (AST).

Dr. Mandayam Thirunarayanan, Florida International University

Mandayam Osuri Thirunarayanan is an associate professor in the School of Education and Human Development (SEHD), in the College of Arts, Sciences, and Education (CASE) at Florida International University, in Miami, Florida. He teaches courses in learning technologies at the undergraduate, masters and doctoral levels. He also supervises doctoral dissertations. His research interests include distance education, cyber-learning, and videogames in education. He has proposed innovative and futuristic educational applications.

Ms. Sai Chaithra Allala, Florida International University

Sai Chaithra Allala received her Master's degree from Florida International University in Computer Science and is currently pursuing a Ph.D. in Computer Science. Her research areas of interest include software engineering, Software Testing, and computer science education. Currently, the focus of her research is using model-driven engineering, natural language processing, and machine learning to automate the generation of test cases from user requirements.

Juan Pablo Sotomayor, Florida International University

Juan Pablo Sotomayor received his Master's degree from Florida International University currently pursuing a Ph.D. in Computer Science. He is currently focused on the research area of Software Testing including MicroServices architecture.

Dr. Monique S Ross, Florida International University

Monique Ross earned a doctoral degree in Engineering Education from Purdue University. She has a Bachelor's degree in Computer Engineering from Elizabethtown College, a Master's degree in Computer Science and Software Engineering from Auburn University, eleven years of experience in industry as a software engineer, and three years as a full-time faculty in the departments of computer science and engineering. Her interests focus on broadening participation in engineering through the exploration of: 1) race, gender, and identity in the engineering workplace; 2) discipline-based education research (with a focus on computer science and computer engineering courses) in order to inform pedagogical practices that garner interest and retain women and minorities in computer-related engineering fields.

Experiences of Integrating Learning and Engagement Strategies (LESs) into Software Engineering Courses

Abstract

The increase in job opportunities for computing professionals in the global community has resulted in a dramatic surge in the enrollment numbers in computer science (CS) departments in many academic institutions. Many of these jobs are in software engineering and require the use of a wide range of skills, including both technical and non-technical skills. The technical skills needed for effective software engineering include: knowledge of the software process and use of a wide array of tools to support development. The non-technical (soft) skills include: effective communication and team management skills. Pedagogy in software engineering courses continue to evolve as new evidence-based approaches become more widespread. However, this evolution tends to lag behind the pedagogical changes in other STEM disciplines.

In this paper we present our experiences of integrating learning and engagement strategies (LESs) into face-to-face (F2F) learning environments for both software engineering and software testing undergraduate classes. The LESs used in our pedagogical approach include: *collaborative learning*, *gamification*, *problem-based learning*, and *social interaction*. Our approach is guided by the LES Integration Model (LESIM) that has the objective of increasing student learning and engagement in both F2F and online learning environments. We conducted a quasi-experimental quantitative study to determine the students' perception of using LESs in a F2F learning environment for two software engineering classes. Based on the results of the study and our experience using LESIM in the classroom we present lessons learned.

1 Introduction

Pedagogy in undergraduate education continues to evolve as new evidence-based approaches to teaching become more widespread in the STEM community. Although many of these techniques have been gaining traction in most STEM disciplines [1, 2], the rate of adoption in the areas of Computer Science (CS), Information Technology (IT), and Software Engineering (SE) is less than expected [3]. The increasing number of students entering undergraduate programs in CS/IT/SE [4] requires that the introduction of these evidence-based approaches be adopted at a faster rate. This is particularly true for students entering SE programs given that the effective development of software applications requires the use of a wide range of skills, including both technical and non-technical skills [5, 6]. The technical skills needed for effective software engineering include knowledge of the: software process – requirements, design, validation and evolution; and tools and techniques (a) to model various artifacts in the requirements and design phases, (b) support verification and validation, and (c) maintenance activities post software deployment. The non-technical (soft) skills include effective: communication, team management and participation, and time management, among others.

In this paper we present our experiences of integrating learning and engagement strategies (LESs) into face-to-face (F2F) learning environments with the expectation of improving student learning and engagement for both software engineering and software testing undergraduate classes. The

experiences reported are based mainly from the students' perspective focusing on the satisfaction of the LESs used in Fall 2019 classes as compared to a class previously taken in their program of study. The LESs use in our pedagogical approach include: *collaborative learning*, *gamification*, *problem-based learning*, and *social interaction*. Our approach is guided by the LES Integration Model (LESIM) and supported by SEP-CyLE[a] (Software Engineering and Programming Cyber-learning Environment). SEP-CyLE [7], an instance of STEM-CyLE, contains vetted learning content in the form of learning objects and tutorials, and can be configured to use various combinations of LESs. LESIM was introduced in the work by Clarke et al. [8], however it is included in this paper for completeness.

To determine the students' perspective of student learning and engagement using LESIM, we conducted a quasi-experimental quantitative study. The main research question focuses on students' perception of student learning and engagement when using LESs in the classroom as compared to the approach used in another class previously taken in their program of study. Preliminary studies have indicated an improvement in student learning when there is an increase of class time dedicated to LESs and a reduction in time for the traditional lecture style of teaching. This improvement is reflected in the midterm examination results for control and treatment groups in a software testing class [8]. The main contributions of this work are:

1. The results of a study showing students' satisfaction when using LESs in both software engineering and software testing undergraduate classes, as compared to the approach used in a previous class in their program of study.
2. A summary of our experiences when using LESIM in two different software engineering classes, and how LESIM maybe use more effectively in future software engineering classes.

Section 2 presents the work most closely related to our work. Section 3 briefly describes LESIM, and Section 4 provides an overview of the software engineering courses and how LESIM was used in those courses. Section 5 describes our study, and Section 6 presents a summary of our experiences using LESIM in F2F classes. We conclude and present future work in Section 7.

2 Related Work

In this section we briefly introduce Learning and Engagement Strategies (LESs) then describe the work cited in the literature that is most closely related to our work. As previously stated the LESs include collaborative learning, gamification, problem-based learning, and social interaction. *Collaborative learning* is where two or more people work in groups mutually searching for understanding, solutions, or meanings, or creating a product [9, 10]. *Gamification* uses game design elements and game mechanics to improve user experience and engagement with a system [11], these design elements and mechanics can be applied to pedagogy [12, 13]. *Problem-based learning* (PBL) is an approach to learning and instruction in which students tackle problems in small groups under the supervision of a tutor [14, 15]. *Social Interaction* is an approach that enhances knowledge acquisition through social activities, such as students establishing meaningful dialogue within student groups and with teachers [16, 17]. Additional details on LESs may be found in Clarke et al. [8].

Connolly et al. [18] examine how the use of computer games can be used to teach software engi-

[a]https://stem-cyle.cis.fiu.edu/

neering concepts, and describe how a computer game they were developing is used to teach such concepts. The authors did a literature review and found twenty-three papers that focused on game-based learning in computer science, software engineering and information systems. Nine of the papers reported evaluation results, the results reported in the papers are mixed. The authors describe the SDSim game which uses a multi-client/single server architecture. The game comprises one or more players that are expected to manage and deliver a number of software development projects. No results of studies were presented in the paper.

Hainey et al. [19] describe an approach to teaching requirements collection and analysis at the tertiary level that uses game-based learning (GBL). The authors developed a GBL application to support the teaching of requirements collection and analysis that is highly motivating and uses an engaging form of media. The argument made is that typical software engineering courses fail to teach the students the skills they require to be professional software developers. The authors conducted a study that compared using GBL and the traditional approach to teaching software engineering. They used a pre/posttest with control group/experimental group design at both a Higher Education (HE) - university level, and a Further Education (FE) - (community) college level. Overall the studies showed that GBL can be a suitable approach to teach requirements collection and analysis at a supplementary level in tertiary education.

Manohar et al. [20] describe the implementation of a federally supported project with the objectives of enhancing the quality of software education via student engagement, and by bridging the gap between the basic principles presented in the classroom and the complexity of real world problems. The project transformed the pedagogical approach by increasing the levels of student engagement and learning through interactive, hands-on exercises, case studies and discussions. The paper describes in some detail how the case studies used in the classes were developed, with an example for software testing being provided. To show the consistency with the institutions' curriculum, the education outcomes of the case study based approach for two courses were mapped to the ABET criteria, as defined by the academic institution. A survey was conducted to assess the effectiveness of the active learning tools developed and used in the classroom. Ten students participated in the study and the results showed that student learning in the areas of problem identification, formulation and solving significantly improved with the use of case studies.

Clarke et al. [8] introduced the LES Integration Model (LESIM), briefly described in the next section, that forms the foundation of the work presented in this paper with respect to the students' perspective of using LESs in F2F classes. Clarke et al. conducted a study to determine the impact on student learning when the time dedicated to the traditional style of teaching is reduced and the time for LESs were increased in F2F software testing classes. The results showed that reducing Lecture Style (traditional) activities by 29% and increasing collaborative learning, gamification, problem-based learning and social interaction activities each by 6%, 5%, 17% and 1% (respectively) can show an improvement in student learning as reflected in the midterm exam grades. Although the improvement in this study was not statistically significant, a follow up study do show statistical significance.

3 LES Integration Model (LESIM)

Figure 1 shows the Learning and Engagement Strategy Integration Model (LESIM) described in [8]. The top of the figure shows the different pedagogical approaches including the LESs - collaborative learning (CL) gamification (GA), problem-based learning (PBL) and social interaction

(SI); and the traditional approach - lecture style (LS). These approaches are used in the context of F2F and online learning environments with access to both F2F and online learning content, shown in the second level of Figure 1.

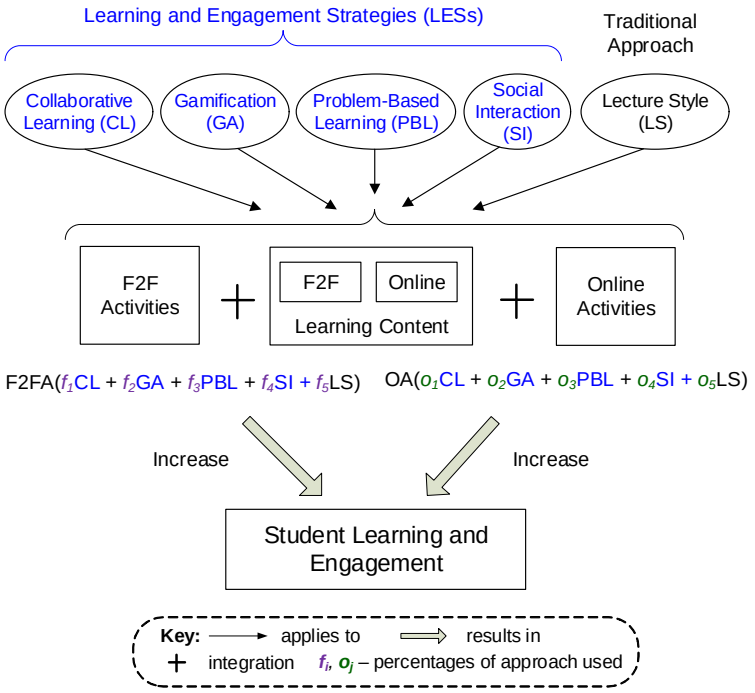

Figure 1: LES integration model (LESIM).

As stated in [8], the main objective of using LESIM is to identify the values of f_i and o_j in the following equations to maximize student learning and engagement.

$$\text{F2FA}(f_1\text{CL} + f_2\text{GA} + f_3\text{PBL} + f_4\text{SI} + f_5\text{LS}) \tag{1}$$

$$\text{OA}(o_1\text{CL} + o_2\text{GA} + o_3\text{PBL} + o_4\text{SI} + o_5\text{LS}) \tag{2}$$

Where F2FA and OA represent the context in which the face-to-face and online activities are realized, respectively. You may think of f_i and o_j as the percentage of the respective pedagogical approach being used. For F2F classes these variables could represent the percentage of class time dedicated to the pedagogical approach, or the number of tasks associated with the approach. It is expected that along with the percentages there will be a context in which the LESs are used (F2FA and OA). For example, CL in the context of F2F activities (F2FA) will most probably be different to the CL in the context of the online activities (OA), examples of the context are provided in Section 4. It is worth stressing that the LESIM model shown in Figure 1 is *adaptable* and *extensible*. It is possible that the four LESs identified in the model may not be the optimal set to maximize student learning and engagement.

4 Using LESIM in SE Courses

In this section we describe the software engineering courses used in the study and how the LESs were integrated into these courses.

4.1 Software Engineering Courses

LESIM has been used in two undergraduate courses at Florida International University (FIU), these courses are CEN4010 Software Engineering 1 and CEN4072 Fundamentals of Software Testing. A brief description of these courses are provided in the text below.

CEN4010 Software Engineering 1: The catalog description for the course: *software process model, software analysis and specification, software design, and testing.* The prerequisite are: CGS3095 - Technology in the Global Arena and COP3337 - Computer Programming II. The grading policy is based on 2 midterm examinations (25% each), a group project (25%), attendance and class participation (5%) and a final examination (20%). The course is taught as a lecture section with no lab component. The required textbook is "Object-Oriented Software Engineering: Using UML, Patterns, and Java" by Bruegge and Dutoit [21], other reading material includes class notes and relevant papers from conferences and journals.

The project for the class used in this study, consists of three (3) deliverables: Software Requirements deliverable that includes, software requirements document (SRD), an in-class presentation, and Unified Modeling Language (UML) [22] diagrams to support requirements analysis; the Software Design deliverable that includes the software design document (SDD), and UML diagrams for system and detailed design; and the Final System deliverable that includes the final system documentation (FSD) - requirements model, design model, implementation model and validation model, in-class presentation, and all UML diagrams developed during the project (requirements model - use case diagrams, object diagrams, sequence diagrams, class diagram; design model - package diagrams (architecture), deployment diagram, state machines, refined sequence diagrams, and class diagrams (using 4 design patterns)). Student teams are required to demonstrate their UML diagrams and execute the test cases in one-on-one sessions with the instructor.

CEN4072 Fundamentals of Software Testing: The catalog description for the course: *test plan creation, test case generation, program inspections, black-box testing, white-box testing, GUI testing, and the use of testing tools.* The prerequisite is: COP3550 - Data Structures. The grading policy is based on 2 midterm examinations (25% each), a group project (25%), attendance and class participation (5%) and a final examination (20%). The course is taught as a lecture section with no lab component. The required textbook is "Foundations of Software Testing"' by Mathur [23], other reading material includes class notes, tutorials on testing tools, and relevant papers from conferences and journals.

The class project consists of two (2) deliverables: Specification-Based Test Deliverable (SBTD) that include a specification-based test document, an in-class presentation, and a demonstration using testing tools to execute test cases (unit, subsystem and system); and Implementation-Based Test Deliverable (IBTD) that includes an implementation-based test document, an in-class presentation, and a demonstration using testing tools to execute test cases and the generation of code coverage (statement and branch). Additional details on the class project, including the testing tools and testing techniques used, are presented in Clarke et al. [24].

4.2 Integrating LESs into SE Courses

A similar approach is used to integrate LESs into both the Software Engineering 1 (SE1) and the Fundamentals of Software Testing (FST) courses. Although Figure 1 shows the integration of LESs into both F2F and online learning environments, we will focus mainly on the integration into

the F2F learning environments. The LESs used in the online learning environment were similar in both classes, a further description will be provided later in the paper. Both classes were taught in an active learning classroom with movable chairs, a wireless connection to the instructor's projector, and at least 5 whiteboards. The context in which the LESs were integrated into the F2F learning environments (F2FA, see Equation 1) for the SE1 and FST classes are as follows:

- *Collaborative Learning* (CL) - is integrated at the class level and at the project team level. Most of the class level collaboration is exhibited by students presenting solutions and answering questions during class sessions. We also consider class project presentations and in-class demonstrations as collaborative learning at the class level. Similarly, working in teams on the class project outside of class is collaborative learning at the team level. Additional collaborative learning activities at the team level include teams solving problems in class using the whiteboards, and other teams checking the correctness of the solutions given.
- *Gamification* (GA) - include several activities which are: (1) students are awarded midterm exam points for correctly answering questions posed by the instructor, note every student in the class is given an opportunity to answer such questions; (2) all students in a team are awarded midterm exam points if the team can correctly answer a question using the whiteboards, the correctness of the solution is determined by the other students in the class; (3) if more than one team produces the correct solution then the number of points allocated for solving the problem is split evenly between the teams; (4) when reviewing a solution on the whiteboard, every student in the class that identifies an error with the solution is awarded a midterm point; and (5) midterm exam points are also awarded to a student if they provide significant assistance to another student in helping them understanding a concept or some other practical aspect of the course. There is a maximum of 10 midterm points that a student can use towards the next midterm exam or final exam, and points cannot be carried forward to the next exam.
- *Problem-Bases Learning* (PBL) - involves students working to solve problems presented in class and in the team projects. Note that the SE1 and FST team projects are formulated to be as close as possible to real world problems faced by software engineers in industry. These problems are open-ended, e.g., determining when a software design is complete, or when should you stop testing (these problems are for medium-sized systems). Additional details on how the problems are introduced in the F2F learning environments can be gleaned from the CL and GA description provided above.
- *Social Interaction* (SI) - students interact socially in the class in a number of ways including: (1) working in teams on the class project, and each member of a team being assigned an administrative role (e.g., team leader, minute taker and time keeper) and a technology-specific role (e.g., requirements engineer, software developer or system tester); (2) all teams participate in the project presentations by being required to ask questions of the presenting team and grading the presenting team; (3) working in teams to solve problems on the whiteboards and checking the solutions to the problems on the boards; and (4) posting and responding to questions on the messaging system provided by the learning management system, e.g., Canvas [25].

It should be noted that there is some overlap between the LESs describe above. For example,

Table 1: Samples used in the study in Fall 2019.

Course	Enrollment	No. Teams	LES Survey	
			Participation	%
CEN4010 Software Engineering 1 (SE1)	18	4	15	83.3%
CEN4072 Fundamentals of Software Testing (FST)	16	4	11	68.8%

working on problems on the whiteboards involve CL, PBL and SI, and if points are awarded for the correct solution or finding an error then GA is included as well.

5 Experimental Study

The study presented in this section provides the students' perspective of student learning and engagement using LESIM in F2F learning environments for two software engineering classes (*SE1* - CEN4010 Software Engineering 1 and *FST* - CEN 4072 Fundamentals of Software Testing). The students' perspective is captured using a questionnaire with both closed-ended and open-ended questions. The main research question being investigated in the study is: *What are students' perception of using LESs in the classroom as compared to the approach used in a class previously taken in their program of study?*

5.1 Method

In this section we describe the following aspects of the study: sample used, data collection approach, and the details of the design.

Sample: Two software engineering classes at Florida International University (FIU) participated in the study, these classes were SE1 and FST. Table 1 shows a summary of the samples used in the study. Columns from left to right in the table show the course number and name, enrollment at the end of the course, number of project teams, and LES survey participation rate. These classes were taught in the evening, 5:00pm (SE1) and 6:25pm (FST) on Tuesday and Thursday, and consisted of both full-time and part-time students.

Data Collection: The data collection for the study involved the use of a survey instrument focusing on the use of LESs in the classroom. The LES survey instrument consisted of 35 questions: 6 questions were related to the overall reaction to the class; 6 questions on the collaborative learning approach; 6 on gamification; 6 on problem-based learning; 6 on social interaction; and 5 open-ended questions. The first 30 questions on the instrument used a Likert scale for the answers, where 1 = *Strongly Disagree*, 2 = *Disagree*, 3 = *Neither Agree nor Disagree*, 4 = *Agree*, 5 = *Strongly Agree*, and NA = *Not Applicable*. The NA response should be selected if the student does not think the LES approach was used in the class. Appendix A shows an abbreviated form of the survey, only the questions reported on in this paper are shown. Two answers were required for each of the first 30 questions, one for a class previously taken in the students' program of study (Data Structures) and the other for the current class that uses LESIM (SE1 or FST).

We decided to use the Data Structures (DS) class since *DS was taken by all students in the SE1 and FST classes* in Fall 2019. In addition, the DS class did not explicitly use LESs during F2F instruction as stated by the instructor. The catalog description for the COP3550 Data Structures (DS) class is: *Basic concepts of data organization, running time of a program, abstract types, data structures including linked lists, n-ary trees, sets and graphs, internal sorting.* Based on

Table 2: Total time (estimated) dedicated to each LES in each class during Fall 2019.

	Learning and Engagement Strategies (LESs)				
	Collaborative Learning (CL)	Gamification (GA)	Problem-Based Learning (PBL)	Social Interaction (SI)	Lecture Style (LS)
Time (mins)	443	155	310	152	815
Percentage	23.6%	8.3%	16.5%	8.1%	43.5%

the syllabus from the instructor that taught most of the course sections in the recent past, the requirements of the course are as follows. The course consist of 7 modules, 4 assignments (25%), one midterm exam (35%), and on final cumulative exam (40%). In addition, the students are required to read chapters from the text book and work on other hands-on exercises (without a grade). The assignments are expected to be completed on an individual basis and are due every 2-4 weeks. The textbook used in the DS class is "Data Structures and Algorithm Analysis in Java" by Mark Weiss [26]. The course was taught as both a F2F class and an online class. Additional student expectations of the class include: taking practice quizzes in Canvas [25], interacting during class time with the instructor and peers, responding to emails within 2 days, and submitting assignments by the corresponding deadlines.

The instrument used in the study was created using questions adapted from other validated instruments [27, 28]. We are currently validating the instrument, the results of this study will help us with the validation process. Prior to this study we received feedback from several experts in the field and updated the instrument to reflect their suggestions, prior to obtaining Internal Review Board (IRB) approval.

Design: We conducted a quasi-experimental quantitative study [29] that involved a survey instrument administered during the final week of the semester. As previously stated, the study was conducted in Fall 2019 at FIU in two SE classes SE1 and FST. The students in each class were informed of the study at the beginning of the semester and were informed that they could opt out of the study if they so choose, without penalty. In addition, it was stated that there was no extra credit for participating in the study. The study was conducted in accordance with the guidelines specified in the Institutional Review Board (IRB) approval for an ongoing NSF Improving Undergraduate Student Education (IUSE) project at FIU.

An estimate of the time dedicated to each LES in each of the Fall 2019 classes is shown in Table 2. The total time used in each class for the entire semester was 1875 minutes. The time dedicated to the Lecture Style (LS) approach during the semester was 815 minutes (43.3%), Collaborative Learning (CL) 443 minutes (23.6%), Gamification (GA) 155 (8.3%), Problem-Based Learning (PBL) 310 (16.5%) and Social Interaction (SI) 152 (8.1%). The class times in the table were computed based on the number of slides presented during class, the lesson plan for each class and the class sessions dedicated to discussing the project and solving in-class problems. LS consisted mainly of the instructor presenting the basic concepts underlying specific topics in each course. For example, a basic concept in each of the classes is as follows: SE1 - the process of performing requirements elicitation; and FST - the approach used to generate test cases using equivalence partitioning. Using these times in Equation 1 we get `F2FA(0.24CL + 0.08GA + 0.17PBL + 0.08SI + 0.43LS)` where `CL` - collaborative learning, `GA` - gamification, `PBL` - problem-

Table 3: Results from the survey for questions 6, 12, 18, 24 and 30 comparing the LESs in the Data Structures (DS) and the Software Engineering 1 (SE1) courses. SD - Strongly Disagree, D - Disagree, N - Neither Disagree or Agree, A - Agree, SA - Strongly Agree, NA - Not applicable, NR - No Response. * - High high number of NA and/or NR responses.

Q.	Item	C	SD (%)	D (%)	N (%)	A (%)	SA (%)	NA (%)	NR (%)	M (SD)
6	Overall, I am satisfied with the class	DS	0 (0)	1 (6.7)	3 (20.0)	6 (40.0)	4 (26.7)	0 (0)	1 (6.7)	3.93 (1.77)
		SE1	1 (6.7)	1 (6.7)	3 (20.0)	5 (33.3)	5 (33.3)	0 (0)	0 (0)	3.80 (1.21)
12	Overall, I am satisfied with my *collaborative learning* experience in the class	DS	1 (6.7)	2 (13.3)	1 (6.7)	3 (20.0)	1 (6.7)	7 (46.7)	0 (0)	3.13* (2.43)
		SE1	0 (0)	0 (0)	2 (13.3)	9 (60.0)	3 (20.0)	0 (0)	1 (6.7)	4.07 (1.68)
18	Overall, I am satisfied with the *gamification* experience in the course	DS	0 (0)	0 (0)	2 (13.3)	1 (6.7)	0 (0)	12 (80.0)	0 (0.0)	3.33* (1.81)
		SE1	1 (6.7)	0 (0)	1 (6.7)	9 (60.0)	4 (26.7)	0 (0)	0 (0)	4.00 (1.00)
24	Overall, I am satisfied with the *problem-based learning* experience in the course	DS	0 (0)	1 (6.7)	0 (0.0)	8 (53.3)	3 (20.0)	3 (20.0)	0 (0)	4.08 (2.22)
		SE1	1 (6.7)	0 (0)	1 (6.7)	8 (53.3)	5 (33.3)	0 (0)	0 (0)	4.07 (1.03)
30	Overall, I am satisfied with the *social interaction* experience in the course	DS	0 (0)	1 (6.7)	7 (46.7)	3 (20.0)	2 (13.3)	2 (13.3)	0 (0)	3.46 (1.77)
		SE1	0 (0)	0 (0)	3 (20.0)	6 (40.0)	6 (40.0)	0 (0)	0 (0)	4.20 (0.77)

based learning, `SI` - social interaction, and `LS` - lecture style. The context in which the LESs are used, `F2FA`, was described in Section 4.2.

5.2 Results and Analysis

Tables 3 and 4 show the results of Questions 6, 12, 18, 24 and 30 from the survey instrument shown in Appendix A. The table consists of eleven columns, which are from left to right: *Q* - identifies the question number from the survey; *Item* - the question as stated in the survey; *C* - the two courses being compared Data Structures (DS) and Software Engineering 1 (SE1) (or DS and Fundamentals of Software Testing (FST)); Columns 4 through 8 - values of the Likert scale starting at Strongly Disagree (SD), Disagree (D), Neither Disagree or Agree (N), Agree (A), and Strongly Agree (SA); Column 9, *NA* - represents Not Applicable; *NR* - No Response; and *M (SD)* - mean and standard deviation for the Likert scale responses. NA was selected by the student if in their opinion the LES was not used in the class. A "*" shown in the rightmost column of the tables identifies a question that had a relatively high number of NA and/or NR responses, see the rows for Questions 12 and 18 in Tables 3 and 4. It should be noted that due to space limitations in the paper we do not provide an analysis for all the items in the survey.

The results in Tables 3 and 4 are shown as a bar chart in Figure 2. The bars are shown as groups of four bars. Each group of bars from left to right are: *DS-SE1* - the bars representing the mean

Table 4: Results from the survey for questions 6, 12, 18, 24 and 30 comparing the LESs in the Data Structures (DS) and the Fundamentals of Software Testing (FST) courses. SD - Strongly Disagree, D - Disagree, N - Neither Disagree or Agree, A - Agree, SA - Strongly Agree, NA - Not applicable, NR - No Response. * - High high number of NA and/or NR responses.

Q.	Item	C	SD (%)	D (%)	N (%)	A (%)	SA (%)	NA (%)	NR (%)	M (SD)
6	Overall, I am satisfied with the class	DS	1 (9.1)	2 (18.2)	0 (0)	4 (36.4)	4 (36.4)	0 (0)	0 (0)	3.73 (1.42)
		FST	0 (6.7)	2 (6.7)	2 (20.0)	4 (33.3)	3 (33.3)	0 (0)	0 (0)	3.73 (1.10)
12	Overall, I am satisfied with my *collaborative learning* experience in the class	DS	1 (9.1)	1 (9.1)	2 (18.2)	0 (0)	1 (9.1)	6 (54.5)	0 (0)	2.80* (2.20)
		FST	0 (0)	0 (0)	1 (9.1)	7 (63.6)	3 (27.3)	0 (0)	0 (0)	4.18 (0.60)
18	Overall, I am satisfied with the *gamification* experience in the course	DS	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	11 (100)	0 (0.0)	-* (-)
		FST	0 (0)	2 (18.2)	1 (9.1)	4 (36.4)	4 (36.4)	0 (0)	0 (0)	3.91 (1.14)
24	Overall, I am satisfied with the *problem-based learning* experience in the course	DS	1 (9.1)	1 (9.1)	0 (0)	6 (54.5)	1 (9.1)	2 (18.2)	0 (0)	3.56 (2.15)
		FST	0 (0)	0 (0)	1 (9.1)	7 (63.6)	3 (27.3)	0 (0)	0 (0)	4.18 (0.60)
30	Overall, I am satisfied with the *social interaction* experience in the course	DS	3 (27.3)	1 (9.1)	3 (27.3)	2 (18.2)	1 (9.1)	1 (9.1)	0 (0)	2.70 (1.75)
		FST	0 (0)	0 (0)	2 (18.2)	6 (54.5)	3 (27.3)	0 (0)	0 (0)	4.09 (0.70)

value of the ratings the students in the SE1 class gave for the DS class they took; *SE1* - the bar representing the mean value of the ratings for the SE1 class; *DS-FST* - the bar representing the mean value of the ratings the students in the FST class gave for the DS class they took; and, *FST* - the bar representing the mean value of the ratings for the FST class.

Some of the highlights from Tables 3 and 4, and shown in Figure 2 are as follows. The average ratings for Question (Q.) 6 appear to be very close showing that students were similarly satisfied with both the DS and the SE1 classes, and the DS and the FST classes. Similarly, the average ratings for the Q.24 related to problem-based learning (PBL) for the DS and the SE1 classes were very close, while there was a bigger difference for the DS and the FST classes. Unlike the SE1 and FST classes, the difference for the average ratings for Q.30 related to social interaction (SI) was greater and may even be statistically significant. The average ratings for Q.12 - collaborative learning (CL), and Q.18 - gamification (GA), although shown in Figure 2, will not be further analyzed due to the high number of missing values.

Further statistical analysis provided additional insights into the questions with few missing values. The average ratings for Q.6, Q.24 and Q. 30 in general are not normally distributed as specified by the Shapiro-Wilk test for normality, we therefore use the Wilcoxon signed-rank test to analyze the

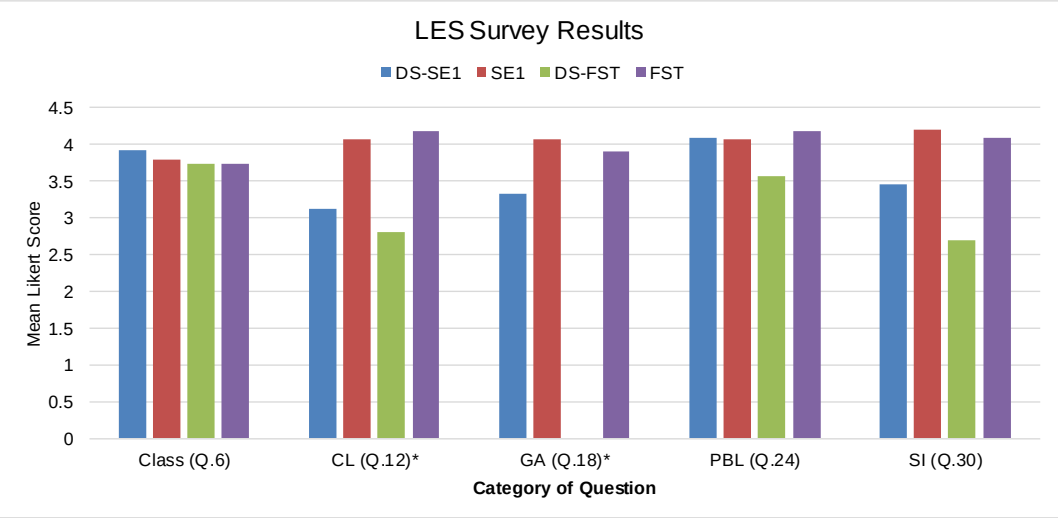

Figure 2: Bar chart showing a summary of the results for the LES survey.

Table 5: The comparison of the ratings for the DS and SE1 classes using the Wilcoxon signed-rank test.

Q.	Item	C	Median	N	z	p
6	Overall, I am satisfied with the class	DS	4	14	-0.54	0.59
		SE1	4			
24	Overall, I am satisfied with the *problem-based learning* experience in the course	DS	4	12	-1.08	0.91
		SE1	4			
30	Overall, I am satisfied with the *social interaction* experience in the course	DS	3	13	2.31	0.02
		SE1	4			

data from the survey. Tables 5 and 6 show the comparison of the ratings from the survey using the Wilcoxon signed-rank test for the DS and SE1 classes, and the DS and FST classes, respectively. The tables consists of seven columns, with the first three columns being similar to Tables 3 and 4. Columns 4 through 7 from left to right show: *Median* - the median values of the ratings; *N* the number of values excluding the missing values; z z-score for the test statistic; and p - p-value representing the significance of the result.

Table 5 shows that there is no statistical significant difference between the ratings for Q.6 and Q. 24 between DS and SE1 classes. However, there is a statistical significant difference between the ratings for Q.30 that focuses on the students' social interaction experience in the two classes. Similar results are shown in Table 6 for the analysis of the ratings for the DS and FST classes.

5.3 Discussion

The research question being investigated in this study attempts to provide insights into the students' perception of using LESs in the classroom, as compared to another class previously taken in their program of study. Based on the results shown in Tables 3 and 4, and shown in Figure 2, the use of LESs in both the SE1 and FST classes were consistently rated 4.00 or higher on the Likert scale. The one LES below 4.00 was gamification in the SE1 class with a score of 3.91. The social interaction LES in the SE1 class scored the highest with a rating of 4.20. Note that the ratings

Table 6: The comparison of the ratings for the DS and FST classes using the Wilcoxon signed-rank test.

Q.	Item	C	Median	N	z	p
6	Overall, I am satisfied with the class	DS	4	10	0	1
		FST	4			
24	Overall, I am satisfied with the *problem-based learning* experience in the course	DS	4	9	1.86	0.06
		FST	4			
30	Overall, I am satisfied with the *social interaction* experience in the course	DS	3	10	2.56	0.01
		FST	4			

for the LESs in both classes were higher than the ratings for Q. 6 the overall satisfaction for these classes. These results would suggest that the students were satisfied with the way the LESs were implemented in the SE1 and FST classes.

When the use of LESs in the SE1 and FST classes are compared to the DS class, the students were slightly more satisfied with the implementation of problem-based learning in the DS class than in the SE1 class. For the other LESs the students were more satisfied in the SE1 and FST classes than the DS class. There was a statistical significant difference with the students' satisfaction with how the social interaction LES was implemented in both the SE1 and FST classes as compared to the DS class, see Tables 5 and 6. This is reflected in the following comment from one of the students in the SE1 class:

Very little social interaction in the other class (Data Structures) as compared to this one (Software Engineering), the social interaction was fun and useful at times and other times it was stressful and time consuming.

Although not statistically significant, the students in the FST class found the problem-based learning to be more satisfying when compared to the DS class, with a z-value of 1.86 and a p-value of 0.06. The following comment related to problem-based learning was provided by a student in the FST class:

The PBL in the Data Structures class and the Fall 2019 course (Software Testing) were about the same. The only difference is the groups forced interaction in the course using LESs (Software Testing).

The results in Tables 3 and 4, and shown in Figure 2, support the fact that students were not sure if collaborative learning took place in the DS class, since 7 students (46.7%) from the SE1 class rated this LES as NA (Not Applicable) in the DS class they took, and 6 students (54.5%) from the FST class gave a similar rating for the DS class. It was clear from the results in the tables and the figure that gamification did not occur in the DS class, with 12 students (80%) in the SE1 class and 11 students (100%) in the FST class stating that the gamification LES was not applicable to the DS class they took. The following are comments from students in the SE1 and FST classes related to collaborative learning:

SE1 class: *As there were no team projects in my Data Structures course, collaborative learning came through communicating with other students in an ad-hoc manner. This course involved an ongoing group project, which called for a more constant and formal collaborative effort.*

FST class: *This is not applicable for Data Structures, but collaborative learning in the course using LESs helped me grasp material faster rather than reading/studying by myself.*

Threats to Validity: One of the main threats to validity was the fact that students did not have a clear understanding of what each LES entailed. That is, what are the essential components of a given LES in the context of a F2F learning environment. This is specifically true for the collaborative learning LES, which is reflected in the fact that 46.5% (SE class) and 54.5% (ST class) of the students thought there was no collaborative learning in the Data Structures class. Note however that this could be explained since some students may have taken the DS class with different instructors. To minimize this threat the SE1 and FST classes were taught by the same instructor with experience using LESIM in F2F learning environments.

Another threat to validity could be the small sample size used in the study, particularly the sample size for the FST class. This sample size is further reduced during analysis when the missing values are removed while performing the pairwise comparison using the Wilcoxon signed-rank test, as shown in Tables 5 and 6. We are currently conducting additional studies to increase the overall sample size and providing a better explanation on how LESs are implemented in the face-to-face learning environment. There is a threat of estimating the effort dedicated to each LES and lecture style approaches, since using class time dedicated to each approach may not be accurate. The contents in Table 4 were estimated after the class was taught using various artifacts. We are currently looking into developing a mobile app to support keeping track of time dynamically.

6 Experiences using LESIM

In this section we present the lessons learned when using LESIM in the F2F learning environment for the SE1 and FST classes, and describe the approach going forward. There are a number of issues with the way LESIM was realized in the SE1 and FST classes, some of these issues were highlighted in the open-ended questions in the survey instrument. Below is a list of lessons learned while conducting the study and reviewing the data collected.

- Awarding points in class should be anonymized as much as possible. Some students find it embarrassing when they do not know the answers to questions and this can deter their participation. A possible solution is making use of in-class technology like clickers. A possible downside is reduced social interaction by the students.
- Implementing collaborative learning, problem-based learning and gamification using the whiteboards to solve problems in class can be time consuming. This requires that the time allocated to these activities should be strictly enforced.
- The team dynamic is very important when using LESIM in the classroom. This is further exacerbated since in most CS/IT/SE programs there are no courses that teach students how to effectively work in teams. Yet this is one of the non-technical (soft) skills [5, 6] required to be an effective software engineer. A student comment from the survey instrument in the SE1 class stated that: *Collaborative learning has its upsides and downsides. Depending on who you are partnering with, collaborative learning can be difficult and leads to larger workloads, in the best scenario it can aid the learning process.*
- Students do not tend to prepare for class by reading the assigned class material, be it notes from the previous classes or other assigned readings. However, if an approach to solving

a problem in presented in class then students do well with in class problem-based learning with these types of problems. Maybe giving the traditional 5-10 minute quiz before class will force the students to better prepare for class.

- When conducting studies using different pedagogical approaches (e.g., as in LESIM) the instructor should spend time in class describing each particular approach. For example, the instructor should describe the nuances between collaborative learning and problem-based learning. Providing such knowledge to the student would improve the accuracy of the data collected. A solution to this problem would be to create learning objects in SEP-CyLE [7] and assign them early in the semester when conducting studies.

One pedagogical approach that would potentially work well with LESIM is using the flipped classroom model where students would be required to do the required reading prior to class [30, 31]. Giannakos et al. [31] identify the benefits of using the flipped classroom model based on a systematic literature review of research papers on the topic. The benefits include: increased learning performance, positive attitudes, increase engagement, more discussions, enforces cooperative learning, and better learning habits. On the other side of the coin the challenges are: high initial cost and very time consuming for the instructor, students are unreceptive with the structure, and decrease student attendance.

Characteristics of LESIM, shown in Figure 1, that can be exploited are its *adaptability* and *extensibility*. We expect that through research this model may be changed by replacing or adding LESs to maximize student learning and engagement. However, we recommend that when using the model it is important to clearly identify the *context* in which the LESs are used in both F2F and online learning environments. In addition, it is very important to explicitly state how the LESs are quantified in the learning environments.

7 Conclusion and Future Work

The use of new evidence-based pedagogical approaches continue to permeate the engineering education space, including software engineering education. In this paper we report on a study that uses the learning and engagement strategies integration model (LESIM) in a face-to-face (F2F) learning environment. The LESs in the model include: *collaborative learning*, *gamification*, *problem-based learning*, and *social interaction*. Unlike previous studies using LESIM the study reports on the students' perspective on their satisfaction with the model in Software Engineering and Software Testing classes as compared to the pedagogical approach used in their Data Structures class.

The results from the study showed that unlike the classes that use LESIM, the Data Structures class used no gamification and little collaborative learning. The students' satisfaction with problem-based learning was similar in both the classes that use LESIM and the Data Structures class that did not formally use LESIM. The students' satisfaction for social interaction was significantly higher for the classes that use LESIM than the Data Structures class. Based on the results of the study and the experience of using LESIM in the F2F learning environment several lessons learned are presented. We are currently performing focus group studies with students to determine qualitatively if the results presented in this paper are supported. The future work involves exploiting LESIM to identify which combinations of learning and engagement strategies most positively impacts student learning and engagement in both F2F and online learning environments.

Acknowledgment

The authors would like to thank the reviewers for their thoughtful and helpful comments. Partial support for this work was provided by the National Science Foundation's Improving Undergraduate STEM Education (IUSE) program under Award Numbers DUE-1525112 and DUE-1712116. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

Appendix A: Abbreviated LES Survey

Q.	**Overall Reaction to Class**	Q.	**Collaborative Learning**
6	Overall, I am satisfied with the class.	12	Overall, I am satisfied with my collaborative learning experience in the class.
	Gamification		**Problem-Based Learning (PBL)**
18	Overall, I am satisfied with the gamification experience in the course	24	Overall, I am satisfied with the PBL experience in the course.
	Social Interaction		
30	Overall, I am satisfied with the social interaction experience in the course.		
	Open-Ended Questions		
31	Please compare your experience using *collaborative learning* in the Data Structures course and the Fall 2019 course using LESs.		
32	Please compare your experience using *gamification* in the Data Structures course and the Fall 2019 course using LESs.		
33	Please compare your experience using *problem-based learning* in the Data Structures course and the Fall 2019 course using LESs		
34	Please compare your experience using *social interaction* in the Data Structures course and the Fall 2019 course using LESs		
35	Any other comments:		

References

[1] M. Borrego and C. Henderson, "Increasing the use of evidence-based teaching in stem higher education: A comparison of eight change strategies," *Journal of Engineering Education*, vol. 103, no. 2, pp. 220–252, 2014. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/jee.20040

[2] C. Henderson, A. Beach, and N. Finkelstein, "Facilitating change in undergraduate STEM instructional practices: An analytic review of the literature," *Journal of Research in Science Teaching*, vol. 48, no. 8, pp. 952–984, 2011. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/tea.20439

[3] M. Stains, J. Harshman, M. K. Barker, S. V. Chasteen, R. Cole, S. E. DeChenne-Peters, M. K. Eagan, J. M. Esson, J. K. Knight, F. A. Laski, M. Levis-Fitzgerald, C. J. Lee, S. M. Lo, L. M. McDonnell, T. A. McKay, N. Michelotti, A. Musgrove, M. S. Palmer, K. M. Plank, T. M. Rodela, E. R. Sanders, N. G. Schimpf, P. M. Schulte, M. K. Smith, M. Stetzer, B. Van Valkenburgh, E. Vinson, L. K. Weir, P. J. Wendel, L. B. Wheeler, and A. M. Young, "Anatomy of STEM teaching in north american universities," *Science*, vol. 359, no. 6383, pp. 1468–1470, 2018. [Online]. Available: http://science.sciencemag.org/content/359/6383/1468

[4] Computing Research Association, "Generation CS: Computer Science Undergraduate Enrollments Surge Since 2006," 2017, https://cra.org/data/Generation-CS/ (retrieved August, 2018).

[5] L. Bender, G. S. Walia, F. Fagerholm, M. Pagels, K. E. Nygard, and J. Münch, "Measurement of the non-technical skills of software professionals: An empirical investigation." in *Software Engineering and Knowledge Engineering (SEKE)*, 2014, pp. 478–483.

[6] A. Radermacher, G. Walia, and D. Knudson, "Investigating the skill gap between graduating students and industry expectations," in *Companion Proceedings of the 36th International Conference on Software Engineering*, ser. ICSE Companion 2014. New York, NY, USA: Association for Computing Machinery, 2014, p. 291–300. [Online]. Available: https://doi.org/10.1145/2591062.2591159

[7] R. Chang-lau and P. J. Clarke, "Software engineering and programming cyberlearning environment (SEP-CyLE)," July 2018, https://stem-cyle.cis.fiu.edu/instances.

[8] P. J. Clarke, D. L. Davis, I. A. Buckley, G. Potvin, M. Thirunarayanan, and E. L. Jones, "An approach to integrating learning and engagement strategies (LESs) into CS class activities," in *126th American Society for Engineering Education (ASEE)*. ASEE, 2019, paper No. 26116.

[9] A. Udvari-Solner, "Collaborative learning," in *Encyclopedia of the Sciences of Learning*, N. M. Seel, Ed. Boston, MA: Springer US, 2012, pp. 631–634. [Online]. Available: https://doi.org/10.1007/978-1-4419-1428-6_817

[10] B. L. Smith and J. T. MacGregor, "What is Collaborative Learning?" in *Collaborative Learning: A Sourcebook for Higher Education.*, A. Goodsell, M. Maher, and V. Tinto, Eds. University Park, Pa.: National Center on Postsecondary Teaching, Learning, and Assessment, 1992.

[11] T. W. Malone, "What makes things fun to learn? heuristics for designing instructional computer games," in *Proceedings of the 3rd ACM SIGSMALL Symposium and the First SIGPC Symposium on Small Systems*, ser. SIGSMALL '80. New York, NY, USA: ACM, 1980, pp. 162–169. [Online]. Available: http://doi.acm.org/10.1145/800088.802839

[12] D. Dicheva, C. Dichev, G. Agre, and G. Angelova, "Gamification in education: A systematic mapping study." *Educational Technology & Society*, vol. 18, no. 3, pp. 75–88, 2015. [Online]. Available: http://dblp.uni-trier.de/db/journals/ets/ets18.html#DichevaDAA15

[13] N. Whitton, *Games-Based Learning*. Boston, MA: Springer US, 2012, pp. 1337–1340. [Online]. Available: https://doi.org/10.1007/978-1-4419-1428-6_437

[14] D. H. Jonassen and W. Hung, *Problem-Based Learning*. Boston, MA: Springer US, 2012, pp. 2687–2690. [Online]. Available: https://doi.org/10.1007/978-1-4419-1428-6_210

[15] H. G. Schmidt, "Foundations of problem-based learning: some explanatory notes," *Medical Education*, vol. 27, no. 5, pp. 422–432, 1993. [Online]. Available: http://dx.doi.org/10.1111/j.1365-2923.1993.tb00296.x

[16] B. Hurst, R. Wallace, and S. B. Nixon, "The impact of social interaction on student learning," *Reading Horizons*, vol. 52, no. 4, pp. 375–398, 2013, https://scholarworks.wmich.edu/reading_horizons/vol52/iss4/5 (retrieved August 2018).

[17] S. Y. Okita, *Social Interactions and Learning*. Boston, MA: Springer US, 2012, pp. 3104–3107. [Online]. Available: https://doi.org/10.1007/978-1-4419-1428-6_1770

[18] T. M. Connolly, M. Stansfield, and T. Hainey, "An application of games-based learning within software engineering," *British Journal of Educational Technology*, vol. 38, no. 3, pp. 416–428, 2007. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-8535.2007.00706.x

[19] T. Hainey, T. M. Connolly, M. Stansfield, and E. A. Boyle, "Evaluation of a game to teach requirements collection and analysis in software engineering at tertiary education level," *Computers & Education*, vol. 56, no. 1, pp. 21 – 35, 2011, serious Games. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0360131510002605

[20] P. A. Manohar, S. Acharya, P. Wu, M. Hansen, A. Ansari, and W. Schilling, "Case studies for enhancing student engagement and active learning in software v&v education." *Journal of Education and Learning*, vol. 4, no. 4, pp. 39–52, 2015.

[21] B. Bruegge and A. H. Dutoit, *Object-Oriented Software Engineering Using UML, Patterns, and Java*, 3rd ed. USA: Prentice Hall Press, 2009.

[22] J. Rumbaugh, I. Jacobson, and G. Booch, *Unified Modeling Language Reference Manual, The (2nd Edition)*. Pearson Higher Education, 2004.

[23] A. P. Mathur, *Foundations of Software Testing*, 2nd ed. Addison-Wesley Professional, 2014.

[24] P. J. Clarke, D. L. Davis, R. Chang-Lau, and T. M. King, "Impact of using tools in an undergraduate software testing course supported by WReSTT," *ACM Trans. Comput. Educ.*, vol. 17, no. 4, pp. 18:1–18:28, Aug. 2017. [Online]. Available: http://doi.acm.org/10.1145/3068324

[25] Instructure, "Canvas," 2019, https://www.instructure.com/canvas (Accessed September 2019).

[26] M. A. Weiss, *Data Structures and Algorithm Analysis in Java*, 3rd ed. Pearson, 2012.

[27] B. Oliver, B. Tucker, R. Gupta, and S. Yeo, "eVALUate: an evaluation instrument for measuring students' perceptions of their engagement and learning outcomes," *Assessment & Evaluation in Higher Education*, vol. 33, no. 6, pp. 619–630, 2008. [Online]. Available: https://doi.org/10.1080/02602930701773034

[28] L. R. Spears, "Social presence, social interaction, collaborative learning, and satisfaction in online and face-to-face courses," PhD Thesis, Iowa State University, 2012.

[29] J. W. Creswell, *Educational research: Planning, conducting, and evaluating quantitative and qualitative research*, 4th ed. Prentice Hall Upper Saddle River, NJ, 2014.

[30] A. Amresh, A. R. Carberry, and J. Femiani, "Evaluating the effectiveness of flipped classrooms for teaching cs1," in *2013 IEEE Frontiers in Education Conference (FIE)*, Oct 2013, pp. 733–735.

[31] M. N. Giannakos, J. Krogstie, and N. Chrisochoides, "Reviewing the flipped classroom research: Reflections for computer science education," in *Proceedings of the Computer Science Education Research Conference*, ser. CSERC '14. New York, NY, USA: Association for Computing Machinery, 2014, p. 23–29. [Online]. Available: https://doi.org/10.1145/2691352.2691354