2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET)

How Much Authenticity can be Achieved in Software Engineering Project Based Courses?

Zahra Shakeri Hossein Abad*, Muneera Bano†, Didar Zowghi‡
* Department of Computer Science, University of Calgary, Calgary, Canada, zshakeri@ucalgary.ca
†Swinburne University of Technology, Melbourne, Australia, mbano@swin.edu.au
‡School of Software and Electrical Engineering, University of Technology Sydney, Australia, didar.zowghi@uts.edu.au

***Abstract*—Software engineering (SE) students not only need sufficient technical knowledge and problem solving ability but also social and interpersonal skills in order to be industry ready. To prepare the students for the *'real world'* the SE educators frequently use *'Authentic Assessment'* and *'Project Based Learning (PBL)'* approaches in their curricula. However, the level of *'authenticity'* should vary within PBL courses offered in different years of a degree program. In this paper, we present and discuss the results of the data collected and analyzed from the first SE course offered to the students. The aim of our research is to explore how much authenticity can be achieved in the first SE course. Our study was conducted at the University of Calgary with 64 software development project teams, totaling 229 undergraduate students. The data is collected from three semesters (2016-2018) in order to assess and monitor students performance. The course design used seven authentic assessments that focused on students skills while covering a complete software development lifecycle. The results from data analysis show that students made progress in some areas of problem solving skills, however, they struggled in their social skills (e.g. people handling skills, negotiations skills and organizational skills), understanding software quality and adaptability.**

***Index Terms*—Software Engineering Education, Authentic Assessment, Project-based Learning, Collaborative Learning**

I. Introduction

The software and IT industry are evolving at a rapid pace and to make the software engineering (SE) graduates ready for industry, the SE curricula and pedagogies need to reflect current professional requirements [1]. The basis of any engineering discipline is learning problem solving skills and creating pragmatic and cost-effective solutions for real world problems by the application of scientific knowledge [2]. A common approach in the SE discipline for making graduates ready for real world is industry-based learning which is achieved by placing students in an actual work environment. However, if the students do not acquire the right skill set prior to this work placement, the industry-based projects may not provide the desired learning outcomes [3].

For the current job market, SE graduates require problem solving skills, social skills required for teamwork [4], and adaptability that makes them self-learners for technological paradigm shifts [5]. The *'engineering'* focus in software engineering (SE) education can confuse the instructor and the students by misleading them about the SEs essential human and social dimensions [6]. Besides the engineering principles, software development knowledge and practice, the students in SE need to understand the *socialness of software* that distinguishes SE from other engineering disciplines [6], [7]. A recent empirical study has shown that SE students lack critical social and communication skills when interacting with customers [8]. Bastarrica et al. [9] have reported that students consider teamwork and planning more challenging than technical problem solving. Hence software engineering education should strike a balance between technical, problem solving, social and adaptability skills [10].

Over the years, software engineering education researchers have proposed alternative approaches to industry-based learning, by designing curriculum and task activities based on project-based learning and authentic assessment principles [11]–[13]. These approaches stress the need for the design of activities to be based on 'realistic' problems that students have to solve in a collaborative environment, thus simulating the real world environment within the classroom. There is still the challenge of how far the educators can go in providing 'authenticity' in their curriculum and tasks, considering that the students are not yet fully trained to embrace all the challenges of the real world. To prepare the students for the real working environment, Dawson [14] proposed twenty 'dirty tricks' to be simulated in project-based learning (e.g. conflicting requirements, hardware crash, or uncertain customers). However, considering the students are yet under training, without proper guidance and supervision these tricks may not be all helpful for students [6].

In this paper, we present the results from a longitudinal study conducted in a software engineering course at the University of Calgary, where the data was collected from three semesters (2016-2018). The project based SE course is offered to students in their second year and it is their very first SE course within their degree structure. The curriculum was designed to utilize authentic assessment in project based learning to assess students for problem solving and social skills. Our research objective was also to analyze the outcomes from the course assessments and evaluate the design of the course. We were interested to explore whether using authentic assessment tasks in the first SE course would be helpful in improving students' learning.

The main contributions of this study are as follows:

- We have designed a set of tasks and activities for a com-

TABLE I: The details of our authentic assessment model, including tasks, activities, and (problem solving/social) skills under study

	Problem Solving skills				Social Skills			
Authentic Activities	Problem Understanding	Planning Skills	Quality Understanding	Adaptability	People Handling Skills	Negotiation Skills	Organizational Skills	**Authentic Assessment Tasks**
A_1 [Inadequate project specification]	●				●	●	●	T_1: Providing a vague description of the product by clients
A_2 [Requirements change]	●	●		●		●		T_{2-1}: Frequent meetings with clients T_{2-2}: Implementing the interactive WoZ technique T_{2-3}: Implementing various prototyping techniques
A_3 [Working with Different Personalities]	●			●	●	●		T_3: Random client/product selection
A_4 [Uncertainty in Time and Effort Estimation]	●	●						T_{4-1}: Implementing the Poker game planning T_{4-2}: Implementing the Silent Group planning
A_5 [Flexible Development Process Model]				●	●		●	T_5: Ask students to define their process model
A_6 [Quality Inspections]		●	●				●	T_{6-1}: Modeling and documentation during the course of the project T_{6-2}: Applying design patterns during the development process
A_7 [Change Programming Paradigm]		●		●				T_7: Switch to the TDD approach in the middle of the development process

plete software development life cycle following authentic assessment principles, that incorporate both problem solving and social skills assessment within teamwork.

- We have conducted analysis both quantitatively (on students marks) and qualitatively (on students and instructors reflections) across three semesters, to explore how helpful is authentic assessment in the first SE course.

The rest of this paper is structured as follows: Section II provides the Background and Research Motivation. Section III gives details of the Study Design by discussing both Pedagogical design and Research design. Section IV provides Results and Discussions and Section V outlines the implications of these results. Finally, Section VI discusses the findings further, draws Conclusions, and provides recommendations for the future research.

II. Background and Research Motivation

Software engineering discipline is required to produce industry-ready graduates. Therefore, the curricula need to prepare students not only with the current technical knowledge but also with self-learning and soft skills. Software Engineering educationists have been employing a combination of *'learning theories'* that have their roots in educational philosophies, most of which fall under a constructivist paradigm of learning [15]. Some of the most widely used *'learning theories'* by software engineering educators in their curricula design are *'Learning by Doing'* [16], *'Situated Learning'* [17], *'Discovery Learning'* [18], *'Learning through Failure'* [19], and *'Learning through Reflection'* [20]. These learning theories provide the foundations of *project-based*, *collaborative*, and *authentic* learning. SE educators have been using these theories in designing curricula for decades, for example, Tvedt et al. [21] combined traditional computer science coursework with experience-based learning (i.e. the software factory) to help students experience the development of a real-world software project. In the same vein, Moore and Potts [22], and Germain et. al [23] showed that how SE education can be improved through project-based courses (i.e. reflection-in-action and Studio in Software Engineering projects).

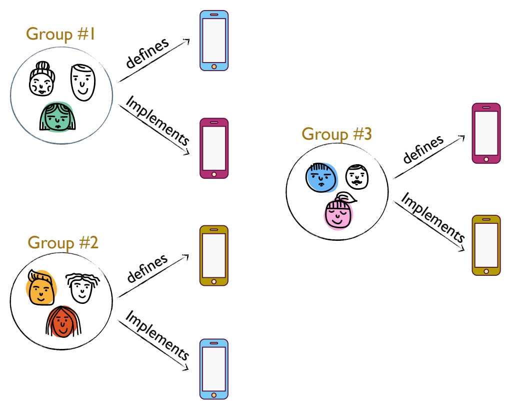


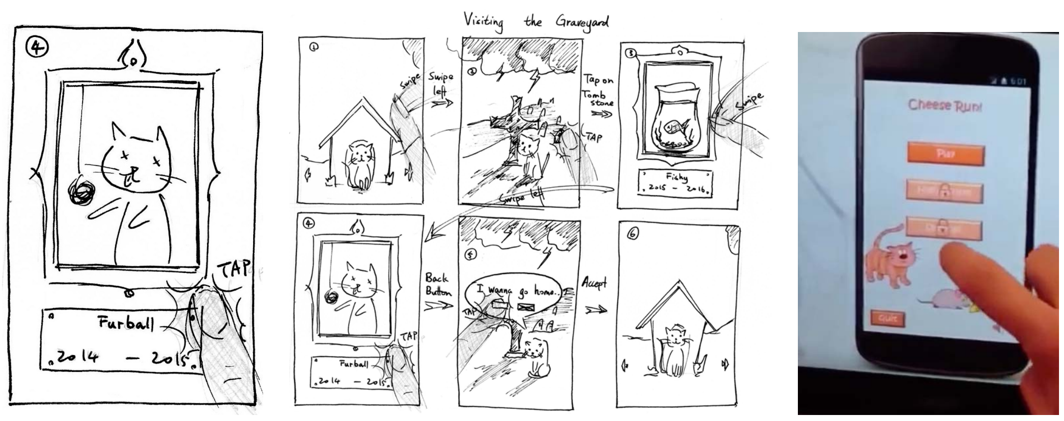
Fig. 1: The project definition/selection process for the study

Capstone projects within the SE program are designed to give students preparation for professional practice and to help them integrate knowledge learned over their program [24]. Before the students reach their capstone project stage, where they can actually be given a chance to face real world environment, they need to develop problem solving and social skills besides technical knowledge. Therefore, most of the SE courses are designed based on project-based learning and collaborative learning paradigms. Authentic assessment in collaborative learning paradigm provides students with the simulation of real-life challenges where they have to focus on problem solving skills based on their previously gained knowledge and the management practices [11], [25]. In authentic assessment, students demonstrate their competencies of knowledge, skills, and attitudes in a professional context [26]. The 'context' is the base planned by the educators to provide the real world setting for learning outcomes and aims for the industry-readiness of the students [11]. The real challenge that SE educators face is that of bringing the right balance of 'realism' and the control of classroom environment for the students (specifically when in the first year of their studies/program) in their curricula and assessments.

The twenty tricks by Dawson [14] provides insights for the

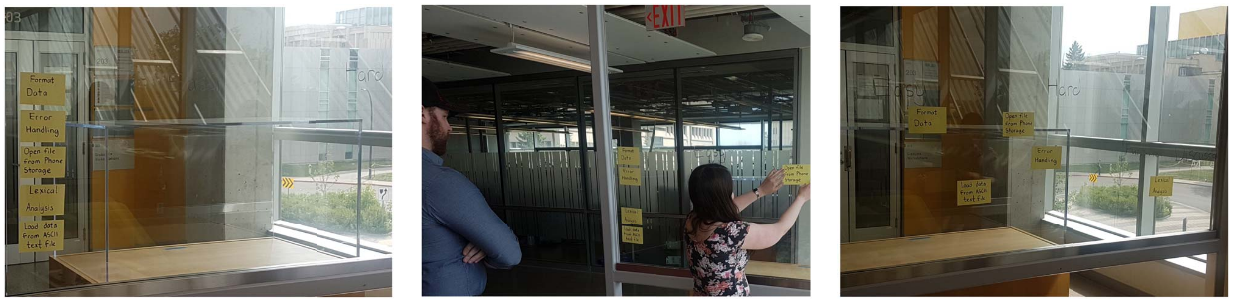
TABLE II: The details of the authentic assessment model used in our longitudinal study, including the *why*, *what* (i.e. corresponding problem solving ans social skills), and *how* (i.e. corresponding tasks) of each authentic activity

	A1 [Inadequate project specification]
Why?	To ensure students experience the uncertainty and ambiguity of specifications in real world projects.
What?	As students need to have close liaison with their client to clarify the requirements of an inadequate specification, this activity will help students improve their *problem understanding, people handling skills, and organizational skills*
How?	• [T_1]: Figure 1 illustrates the process that students followed to select their course project in this course. As the description of the course project was provided by an *unknown* client, all teams were asked to provide a specification of no more than two paragraphs long. We specifically asked clients to provide ambiguous statements and omit some aspects of the product, such as the details of the functionalists and the requirements of the system.
	A2 [Requirements Change]
Why?	To prepare students for the real world in which requirements and their priorities are changing all the time.
What?	By understanding that requirements in a real world change inevitably, students need to plan for these changes by flexible planning, adaptable designs, and strong negotiation skills. Thus, the successful implementation of this authentic activity helps students improve their *problem understanding, planning skills, adaptability*, and their *negotiation skills*.
How?	• [T_{2-1}]: Following the fact that students need to perform all the project tasks based on the initial description of the system that has been provided by their client, they needed to contact their customers frequently and to document their collaboration in their report. • [T_{2-2}]: In the early stages of the development process, students were asked to create a video showing a user interacting with their system. Although their functionality was not complete at that point, the system responses could be simulated using another person. They needed to ask their clients to think aloud when interacting with the system. During this phase, clients were encouraged to support their suggestions by hand-drawn sketches (see Figure 2c). • [T_{2-3}]: Students needed to create 5 overview sketches showing individual snapshots of their system's interface. Each sketch should represent a different idea for the interface. They were to choose one of the overview sketches and create five more detailed sketches elaborating on it and to create a storyboard showing a user performing a task with their system (see Figure 2a,b).
	A3 [Working with Different Personalities]
Why?	To help students understand the integral role of communication and the importance of client satisfaction during a software development project.
What?	By conducting this activity and it corresponding tasks, students will be able to improve the *problem understanding*, and *adaptability* skills required for their problem solving tasks as well as their social skills including *people handling* and *negotiation* skills.
How?	• [T_3]: Ass illustrated in Figure 1, teams will not implement the product that they will define. When a team defines a product, the team members will act only as the customers of that product and another team (who is interested in the defined product) will implement it. As the teams have no choice of choosing their clients, they need to collaborate with different personalities with different expectations during the course.
	A4 [Uncertainty in Time and Effort Estimation]
Why?	To help students exercise the dynamics of planning and estimations and to help them realize the time overhead of activities not related to their principal tasks in real software development projects.
What?	By practicing this activity, in addition to improving their *planning skills*, students become to improve their *problem understanding* skills, as a required skill for more accurate and realistic estimates.
How?	• [T_{4-1}]: Students were asked to perform a planning poker session with their teammates. They needed to break down one of their functional requirements into a set of tasks and provide a set of cards to each team member. Have each team member pick a card for each task, then reveal their cards and compare estimates. Repeat this process with each task. In several paragraphs, discuss their planning session. • [T_{4-2}]: Students were asked to perform a silent group effort estimation session with their teammates. They were asked to create a set of features from the functional requirement defined in earlier stages of their development process, and label a wall with EASY on one side and HARD on the other side. Have their team members form a line and silently place the features on the section of wall where they think the feature fits. The session is finished when the features stop moving around (see Figure 3).
	A5 [Flexible Development Process Model]
Why?	To help students experience changes in working procedures and understand the need for being flexible and adaptable during a development lifecycle.
What?	To be able to manage inevitable changes and disruptions that occur during a development process, and to be able to adapt to externally caused changes, students need to improve their *adaptability, people handling*, as well as their *organizational* skills.
How?	• [T_5]: Before starting the development process, students were asked to choose one of the following software process models: Opportunistic, Waterfall, Spiral, Concurrent, or Scrum and justify why they chose this methodology for developing your system over the others. During the course of the project, they could change the process model if required.
	A6 [Quality Inspections]
Why?	To make students understand the crucial need for quality standards which must be maintained throughout a development project.
What?	As students, by applying this activity, need to apply various quality standards (e.g. code comments, documentations, and code design standards) throughout a project, they become to improve their *planning, quality understanding*, and *organizational* skills.
How?	• [T_{6-1}]: During the course of the project, students were asked to create structural and behavioral models using standard tools following the Unified Modeling Language (UML) notations. To keep the documentation process efficient, we asked students to model only the essential features of their system and abstract away unnecessary details without oversimplifying. • [T_{6-2}]: During the development process, among a list of design patterns (i.e. Facade, Strategy, Adapter, Singleton, and Observer) students were asked to pick one design pattern and apply it in their project.
	A7 [Change Programming Paradigm]
Why?	To help students understand the dynamics of software development projects and learn that changing the programming paradigm/language/environment is commonplace in real world, so they must be more flexible and realistic with their plannings and collaborations.
What?	To accommodate this type of unplanned changes during a development process, students must improve their *planning* and *adaptability* skills.
How?	• [T_7]: In the middle of their development process, students were asked to implement one of their functional requirements using a test-driven development (TDD) approach which needed them to change their estimations/planning as well as their team structures (i.e. job roles).



(a) Sketching (b) Story Boarding (c) Wizard of OZ

Fig. 2: The prototyping techniques applied in authentic task T_{2-3}



(a) Initial Layout (b) Alternating tasks by team members (c) Final Layout

Fig. 3: Silent grouping effort estimation

educators on how to introduce authenticity in their assessment tasks of problem solving and social skill learning activities. For example, to help students improve their *problem understanding* and *people handling* skills, he suggests that the role play in the course project should include uncertain and naive customers, who are not sure of what they want and have no technical background so that they cannot even imagine what form the possible solution should take. Santos et al. [27] have proposed a PBL teaching and learning methodology called xPBL that adopts the principles of authentic assessment along with a flexible curriculum which closely monitored by professional practitioner tutors. These studies do not differentiate the state of knowledge and skills students have at various levels of their degree program. Tai and Yuen [13] have defined authentic assessment in the context of PBL from three perspectives: Content (the knowledge of the students), Process (application of the knowledge in problem solving scenario), and the Output (the product and artifacts produced as a result of process and contents). However, these three perspectives do not assess the social and interpersonal skills required by the students within a collaborative environment of PBL [11].

Although the principles and learning objectives behind PBL and Authentic Assessment are fully aligned, we do not have any well-defined guidance on how to apply these approaches in the most effective way and how to monitor the learning outcomes over a period of time [28]. How far a real world environment can be simulated within software engineering classroom environment is a challenge, and every university may have their own way of interpreting the context, process and the learning outcomes of SE courses.

A systematic review [29], on studies reporting on the application of PBL in computing subjects from 1997-2011 has shown that even though it is preferred pedagogy for improving teaching and learning, there are only few example papers explicitly demonstrating how to implement PBL and authentic assessment in a systematic way. Also, there is a lack of longitudinal empirical studies to analyze whether any of the recommended guidelines work in different contexts and a cohort of students.

III. Course Design

Our study is based on the data collected from the first software engineering course offered to undergraduate students in their second year of a software engineering degree. In the following section, we will describe the pedagogical design of the course and research design of our study.

A. Pedagogical Design

The course in this study was designed based on the following pedagogies:

a) Collaborative Learning: The study was conducted from the data collected from 64 software development project teams, totaling 229 undergraduate students in their first SE subject. Implementing the prototyping techniques, such as sketch, storyboard, and Wizard of Oz (e.g. Figure 2) to clarify and explore the requirements of a project [30], [31], as well as the planning and effort estimation techniques, such as poker effort estimation, and silent grouping estimation (e.g. Figure 3) are two samples of the tasks for supporting collaborative learning in this study).

b) Project-based Learning and Role-Playing: As illustrated in Figure 1, in this course, the students were asked to propose a mobile app and act as the client for another team, while at the same time developing an app based on the idea proposed by a different team.

c) Authentic Assessment: Table I and Table II provide the details of the authentic tasks that correspond to both problems solving and social skills. The tasks were inspired by the insights of Dawson [14] to deliberately create a challenging real world environment. Figure 4 presents the mapping between these tasks and each of the social and problem solving skills during each iteration (i.e. assignment) of the course project.

d) Learning Through Reflection: The students were asked for reflections on their tasks after performing each of the authentic tasks listed in Table II. For example, to collect student reflections on T_1, at the end of the Assignment #1, we asked students to: *"Provide details of your collaboration (e.g. meeting, email, stand- up meetings, etc) with your customer. How it helped you to do this assignment!"*.

Likewise, to record student reflections on the planning tasks T_{4-1} and T_{4-2}, we asked students to *If you notice any huge gap between the efforts estimated with the two approaches, try to identify and explain the factors that caused that difference. List also the difficulties that you faced while performing estimation with the two approaches"*.

B. Research Design

Our research was longitudinal and exploratory in nature and was motivated by the fact that as educators we aimed to improve the design of our curricula based on systematic assessment of students performance. We were interested to explore the extent to which authentic assessment could be used in the first SE course. We collected the data from three semesters for all the tasks and assessments.

1) Research Questions: Our study aims to uncover whether and how the application of authentic assessment activities in a software engineering course impacts student problem solving and social skills. This main goal led us to the following research questions:

RQ1 [*To what extent does Authentic Assessment help students improve their problem solving and social skills in a collaborative learning environment?*

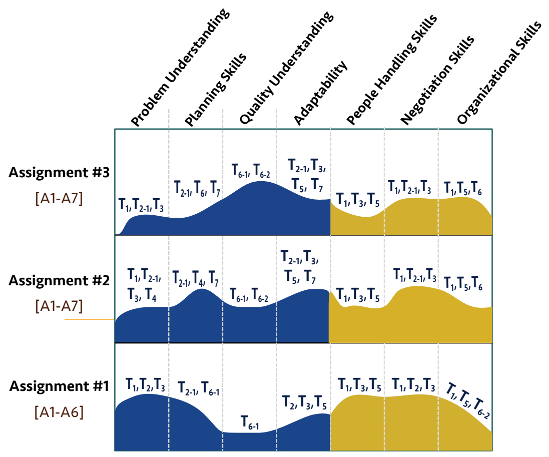


Fig. 4: The distribution of authentic tasks during different iterations of the course project, including the mapping between tasks and each of the problem solving and social skills. The main focus of each assignment is indicated by the height of the tasks.

RQ2 How does Authentic Assessment help SE students in improving their problem solving and social skills in a collaborative learning environment?

2) Data Collection: Both qualitative and quantitative data were collected, from the three assignments that students submitted during the four-month period of the course. Sixty-four teams (229 students) participated in this study over three consecutive years. Each student was enrolled in the course for only one semester and was assigned to just one team.

Teams were asked to submit three assignments, which were designed to help students connect course material to real world experiences, using a set of authentic assessment activities and tasks (see Table I for more details about these tasks). Figure 4 illustrates the distribution of the authentic assessment activities and tasks as well as the mapping between these tasks and each of the problem solving and social skills during each iteration of the course project.

Moreover, for the purpose of implementing this study, the first author of the paper manually analyzed each of the submitted assignments and, on a scale of 1 to 10, rated students problem solving and social skills, using the assessment criteria listed in Table III. The quantitative data that were collected during this step was used to investigate the usefulness of the applied authentic activities and their corresponding tasks for improving students problem solving and social skills. The data collected for this study is kept anonymized, and teams were identified using a numeric Team-ID.

3) Data Analysis: To evaluate the impact of authentic assessment tasks and activities on student social and problem solving skills (RQ1), as we could not confirm that the underlying populations follow a normal distribution (using *Q-Q* plots [34]), we used the non-parametric Kruskal-Wallis test. To determine the statistical significance we used the *p*-

TABLE III: Assessment criteria for measuring the extent to which authentic assessment help students improve their problem solving and social skills. These criteria were inspired by the in-sights of the previous works presented in [14], [32], [33].

Skills	Assessment Criteria
Problem Understanding [PU]	• The ability to understand, articulate, and solve complex problems and make sensible decisions based on available information [32]
Planning Skills [PS]	• The ability to effectively determine goals and priorities, deadlines and resources required to achieve them [33].
Quality Understanding Skills [QUS]	• Understanding the importance of quality in both product and development process [14]. • Considering standards that must be maintained throughout a project and not handled as an afterthought [14].
Adaptability [A]	• The ability to adapt to changes when performing a task without showing resistance [14], [32]. • The ability to participate actively in achieving a common goal, even when cooperation leads to a goal that is not directly related to ones own interests [33]. • The ability to learn new concepts, methodologies, and technologies in a reasonable time-frame [32].
People Handling Skills [PHS]	• The ability to deal with other people through social communication and interactions under favorable and easily make contacts and engage in social activities. [32], [33]. • The ability to work effectively in a team environment and contribute toward the desired goal [32].
Negotiation Skills [NS]	• The ability to identify ones own and other peoples positions in a negotiation, exchanging concessions and reaching satisfactory agreements on the basis of a win/win philosophy [33].
Organizational Skills [OS]	• The ability to efficiently manage various tasks and to remain on schedule without wasting resources [32]. • Following organizational policies and procedures [33].

TABLE IV: RQ1- The results of the statistical analysis to investigate the usefulness of AA activities in improving student problem solving and social skills (on a scale of 1-10)

Skills	Assignment #1		Assignment #2		Assignment#3		A1-A2	A2-A3	A1-A3
	Mean	SD	Mean	SD	Mean	SD			
Problem Understanding [PU]	3.48	1.61	7.32	1.72	9.06	.79	.00038	.00027	2.1e-6
Planning Skills [PS]	2.89	1.51	4.50	2.19	6.70	2.08	.002	2.2e-8	9.9e-3
Quality Understanding Skills [QUS]	2.15	1.44	3.37	1.78	5.25	2.83	0.05*	3.2e-5	4.3e-8
Adaptability [A]	4.01	2.34	3.67	2.11	6.09	1.61	.64*	7.4e-7	1.3e-4
People Handling Skills [PHS]	2.37	1.18	3.12	1.45	3.84	2.33	.02*	.62*	.0006
Negotiation Skills [NS]	4.00	1.98	7.29	1.80	7.57	1.77	2.6e-9	.75*	5.4e-10
Organizational Skills [OS]	4.04	1.17	7.53	1.85	8.43	1.30	4.5e-10	.11*	2.4e-10

*: The impact of AA activities on the corresponding skill is not significant (*p-value*> 0.01).

values (< 0.01), and report as significant, differences at 99% confidence interval.

To answer RQ2, all the collected qualitative data, including students answers to open-ended questions in their assignments as well as the written feedback they have received from their clients were analyzed by qualitative coding. This open coding required us to analyze each development team response line-by-line, to extract meaningful results. To implement this process and to coalesce all of the extracted concepts, we used NVivo [35], a qualitative analysis software tool to code and synthesize qualitative data.

IV. Results and Discussion

A. RQ1- The Extent to which Authentic Assessment Help Students Improve Their Problem Solving and Social Skills

To answer this research question we posed the following null hypothesis which tests whether the applied authentic assessment method, including the AA activities and tasks, affect student problem solving and social skills.

Null Hypothesis: The effect of applying AA tasks and activities on students' s_i skill, during the course of the project, is not significant.

Where $s_i \in \mathcal{S} = \{$PU, PS, QUS, A, PHS, NS, OS$\}$ and presents different problem solving and social skills studied in this paper and listed in Table I.

Table IV summarizes the means and standard deviations of the scores that teams have received for each skill at the end of each iteration (i.e. assignment). Moreover, this table lists the results of our hypothesis testing, using the Kruskal-Wallis tests and presents the p-value for each of these tests. In the rest of this section, we look in detail at the effects of AA on students score for each of the *problem solving* and *social* skills and discuss these effects over the project timeline.

1) RQ1-1: Problem Solving Skills: Recall from Table IV, *problem understanding [PU]* skill showed more improvement over the course of the project, compared to other skills of this category (i.e. from mean=3.48 in Assignment#1 to mean=9.06 in Assignment #3). Figure 5a presents the details of the distribution of scores for this skill after each iteration. One reason for the low average score for this skill after the first iteration is the inadequate specification that students received for their course project. Practicing different prototyping techniques (T_{2-2} and T_{2-3}) as well as a mandatory high level of collaboration during each step of the project helped students obtain the requirements of their system and obtain detailed agreement with their client. Moreover, looking at the results of the Kruskal-Wallis tests (Table IV), the 99% confidence analysis show that the effects of applying the AA tasks (i.e. T_1, T_2, T_3, and T_4) on improving students PU skill achieve statistically-significant improvement during the course of the project.

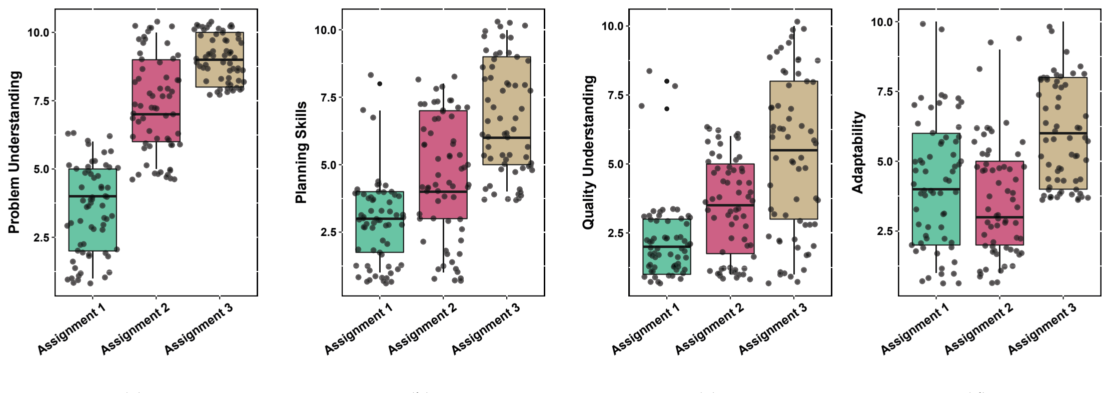


Fig. 5: RQ1- The extent to which authentic assessment help students improve their problem solving skills during the different iterations of the course project.

Planning skills [PS] come next, with the final mean score of 6.70. It can be seen from Figure 5b that there were teams who were still struggling with their planning skills during the second iteration of their course project, but after implementing the *Poker* and the *Silent grouping* time and effort estimation techniques (i.e. T_4) and comparing their plans with their real progress, they found how bad their time estimation and planning had been. Students learned to be more realistic and flexible in their planning for the third iteration and could achieve a higher score for their planning during the third assignment. However, similar to the PU skill, the 99% confidence analysis shows that the effects of applying the AA tasks on improving students planning skills achieve statistically-significant improvement during the course of the project. Recall from Table IV and Figure 5b-c, students found the *quality understanding* and the *adaptability* skills more challenging during the first two iterations of their course project. While students were asked to improve their QU skill through performing modeling and documentation (T_{6-1}) as well as through applying the design patterns (T_{6-2}) tasks, the teams felt vulnerable in terms of the timing of their projects and took quality shortcuts to make their deadlines during the first two iterations. However, following the feedback they received after the second iteration, students learned about the importance of standards which must be maintained throughout the project and integrated these standards into the third iteration of their development process.

To improve students *adaptability* skills during a problem solving task, we applied AA activities which require a high level of flexibility and adaptability (i.e. A2, A3, A5, and A7), enough to ensure a successful implementation of the task (Table I). Students found this skill very challenging, specifically during the second iteration where they were asked to change their programming paradigm to Test Driven Development (TDD). As a result, they failed to perform well in adapting to the new context during the second iteration (Figure 5d and Table IV).

2) RQ1-2: Social Skills: comparing to the *Negotiation* (Figure 6b) and *Organizational* (Figure 6c) skills, students found the AA activities which requires practicing *people handling skills* more challenging, with receiving the mean score of 3.84 after the third iteration (Figure 6a), as stated by:

> *"The greatest challenge was to communicate with our clients and setting meetings for them to check the progress of the app. Basically, it was hard to make appointments, they do not commonly answer emails."*

Moreover, looking at Table IV, the results of the 99% confidence analysis show that the effects of applying AA tasks (i.e. T_1, T_3, and T_5) on improving students *people handling* skills have not achieved statistically-significant improvement until the last iteration of the project.

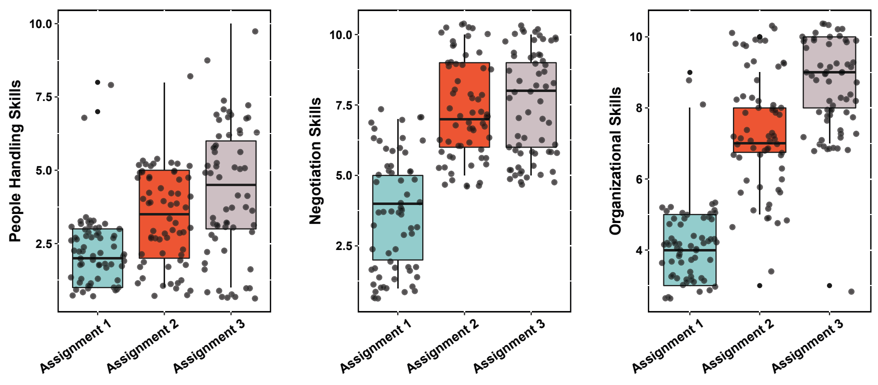
B. RQ2- How Authentic Assessment Helped to Improve Student Problem Solving and Social Skills?

To address this RQ, we discuss how each of the AA activities, listed in Table I helped students improve their problem solving and social skills. To this end, we have reviewed and summarized all submitted assignments over the three years of the study duration as below:

1) A1– Inadequate Project Specification: Recall from Table I, the A1 activity aimed to help students improve their *problem understanding, people handling, negotiation*, and *organizational* skills. To achieve this goal, students received a vague description of their project and they were required to seek clarification by having a close liaison with their clients and their teammates.

The majority of teams under study stated that they benefited from the customer's early feedback in different ways. They believed that the most important part was increasing their understanding of the products' requirements significantly.

> *"The collaboration with the client and among ourselves was incredibly helpful in this assignment. Unlike other courses where the assignments are clear, this one is much more open-ended and so there was much more*



(a) (b) (c)

Fig. 6: RQ1- The extent to which authentic assessment help students improve their social skills during the different iterations of the course project.

uncertainty. Having constant communication was essential for developing a solution that everyone was happy with."

We also noticed that, in addition to their *problem understanding* skill, this activity and its corresponding task (i.e. T_1) helped students improve their social skills such as *people handling, negotiation*, and *organizational* skills:

> *"We met with our client once a week since our project began, which has helped to keep the progress on the project moving ahead. Our clients' experience in the problem domain has helped us greatly through keen evaluations of various ways ahead. This helped us to avoid pitfalls such as trying to develop complex mathematical algorithms that would likely have been beyond the computational abilities of most mobile devices. We have also kept him updated on what our current assignments in the course were. All of these deliverables were available for his review if he wanted to. Additionally, if any major issues have come up, we have informed him about it, explained what we plan to do about it, and asked if he had any suggestions for how do resolve it."*

2) A2– Requirements change: After students received a general description of their project from their client, they were asked to practice the requirements change activity (A2) through implementing Tasks T_{2-1}, T_{2-2}, and T_{2-3}. While implementing these tasks triggered a high level of requirements change, many realized that these techniques greatly helped them in the general decision making and clarification of the product. as in:

> *"The prototyping techniques helped us clarify our non-functional requirements and how we generally wanted the app to look alongside explore the different ways we can implement functions. In particular, these techniques helped us clarify the Organized and Streamlined Visuals NFR. We found that we were unsure exactly how we wanted our app to flow for the user/client, and the sketches helped us by visualizing possible app appearances without having to build the app and instead just drawing it on paper to view."*

Another interesting finding of this study was that students had not seen the frequent and inevitable changes to the requirements of their project as an obstacle and had noticed that it was not as difficult as expected.

> *"We found the requirements change activity very manageable during the course of this project. The collaboration with our client was very important and useful because they were able to describe in greater detail and during the prototyping activity, draw out their thoughts for us to see. This is not so evident when it is just a text document that we are working from."*

Many also realized that the authentic tasks that were applied to manage and practice the requirements change activity helped them to ensure all major decisions are approved by their customer so that there will be no discrepancy between what is expected and what is being delivered.

> *"The customer also has a high amount of input to how the product proceeds and evolves as it continues being developed. Any sort of disagreement or confusion between any of our team members about how to proceed can be easily clarified and resolved in this way."*

3) A3– Working with Different Personalities: To implement this authentic activity, which aimed to teach students the importance of human factors and communication in analysis and understanding of requirements, we defined the process of client/product selection to be anonymous and based on students' interest in the products that were suggested by various clients. Recall from Table IV, the extent of the improvement of students' organizational skills at the end of the course, comparing to the first assignment, is statistically significant.

However, a few stated that the biggest obstacle in communicating with their customer was their clients' negotiation style, which had made it difficult to establish many full group meetings.

"The struggles of meeting with our customers is, of course, trying to come up with an agreement during our discussions. This could not happen, so we decided to communicate with only one of our customers at a time, which was still helpful in answering quick questions for us. Nevertheless, meeting as a group could really help us to see the bigger picture that they were envisioning."

4) A4– Uncertainty in Time and Effort Estimation: Changing the delivery date of a project, or introducing unplanned activities with a time overhead is very commonplace in real software development projects. To help students exercise this authentic activity, we asked students to implement two types of *time* and *effort* estimation techniques, including T_{4-1}: Poker effort estimation [36], [37], and T_{4-2}: Silent grouping effort estimation [36] during various stages of their development process. When asked students at the end of the second iteration to compare their estimations with the actual elapsed time and effort, the majority stated that the main reason for the difference between their estimations would be the lack of experience which their group possesses in regards to estimating the time taken for various tasks and some ambiguity in deciding what values they would place on each cost driver.

"Since this is the first time any of us have created an app, it was difficult to give an accurate estimate of how many lines of code/the effort in person-hours each task would take."

However, students found the Silent Grouping Effort Estimation to be an enlightening process that allowed them to plot and emphasize the relative difficulty of their tasks. They realized that this activity promoted silent debate over task estimates that allowed them to gain a feel for each other "standings" with a task before opening the floor to live discussion at the end to address each other's concerns.

Some students revealed that their estimations have been rather inaccurate during the first two iterations (i.e. Assignments #1 and #2) of the project, but during the project, they practiced the planning tasks and ended up with more accurate plans and estimations towards the end of the project.

"The accuracy of the estimates following our team discussions were surprisingly accurate. Our planning experience during the first two assignments contributed to better estimates. We found this to be somewhat surprising, given our limited experience with the Android Studio development environment."

5) A5– Flexible Development Process Model: One of the most interesting findings based on the data collected from students development process is that during the course they had noticed that they need to change their development process model due to changes that had been happened to their requirements and teams plan.

"[...] At first, we considered a waterfall approach in hopes of taking advantage of the nature of our assignment. It was our thoughts that we could obtain a concrete understanding of our project which would satisfy the requirements of a successful waterfall approach. It was our belief that with the specifications already laid out that the waterfalls carry forward approach would allow us to maximize our efforts and allow us to complete the product in the most timely matter. Unfortunately, it was during the requirements gathering procedure that we gained a sense of the potential of the product. The products capacity for improvement also meant that such improvements would require an, as of the moment, an indeterminate amount of time. The potential for prototyping coupled with the considerable potential for improvements to our product we decided that to maximize our time and efforts, the Scrum model would be the most effective."

6) A6– Quality Inspections: To ensure students will not postpone the documentation and quality inspection to the end of the project, we asked them to consider the maintenance of quality standers from the early stages of their development process (e.g. T_{6-1}, T_{6-2}).

Some teams realized that the standard documentation that they have used during the early stages of their development, can be used as a mean for clarifying and understanding the requirements of their product.

"We are assuming that this arrangement of classes will translate well into an app, though given our lack of experience with Android Studio and developing apps in general, this may not end up being the best arrangement of classes. Also, after collaborating this design with our client, we are assuming that these classes and methods will be all that is required to produce the appropriate functionality. "

Interestingly, while this was the first time that students in this course understood the whole life cycle of a software development project, some realized that the implementation of this activity positively impacts the maintainability of the system.

"[...] This also makes maintainability easier. Any changes to how saving files is done for different situations are handled can be done by changing one class only. The strategy design pattern also helps make the project more cohesive as everything associated with saving files is associated with these classes, not scattered throughout the entire project."

7) A7– Change Programming Paradigm: This activity aimed to challenge students' *planning* and *adaptability* skills through introducing new tasks (e.g. changing the programming paradigm to TDD) in the middle of their development process. As illustrated in Figure 5d and Table IV, based on the results of analyzing the first and the second assignments, many found this activity very challenging which negatively impacted their planning and time management.

"We did not know how some features will be implemented using the TDD approach. One implementation could take only a few minutes and another might take many hours."

They also noticed that their experience level was an influential factor in dealing with the challenges caused by the newly introduced task. As stated by:

"We have minimal coding experience and no prior experience estimating the TDD approach. We tried to give conservative estimates but even these were likely not accurate because of our lack of domain knowledge and prior experience."

However, after two iterations, students learned that introducing additional tasks and activities, which might not be directly related to their on-going tasks, are commonplace in the real world and they should always consider the time overhead of these activities in their planning and team management. As illustrated in Figure 5, the average score of the teams for both *planning* and *adaptability* skills were improved in the last iteration (Assignment #3).

V. Threats to Validity

We followed and adapted the empirical research guidelines in software engineering provided by Wohlin et al. [38] to mitigate the limitations of empirical research. To evaluate the reliability of our coding process (for analyzing the open-ended questions), we used the Cohen's Kappa statistics to measure the degree of agreement and consistency between the extracted codes by the first author and a domain expert, who were involved in the coding process. The calculated Kappa value was 0.91, which shows significant agreement according to Landis and Koch [39]. Moreover, the results of our assignments evaluation rely on self-assessment by the first author. These results were triangulated in multiple ways and the evaluation form was pilot tested with 10 teams under study to mitigate the risk of subjective bias, however, we cannot control all biases.

Another threat to the validity of our study is its generalizability to a broad student population. Our population under study might not be representative of the entire software engineering students, our study results might thus suffer from a lack of generalizability. We mitigated this threat by implementing our study on a relatively large population through three semesters (206-2018). However, this paper is entirely based on students' experience of real-world software development and, consequently, their team structure (e.g. software development background, client's characteristics, or projects' characteristics) can affect the results and interpretation of the main findings of this study.

VI. Conclusion

In this paper, we have reported on the effects of applying the authentic assessment tasks in the first software engineering course offered to students. We assessed and measured the usefulness of the applied activities and tasks on students learning by using qualitative and qualitative analysis of the data that was collected from three semesters (2016-2018).

From the details of the analysis provided in previous sections, we learned the following lessons:

- With the exception of few teams [1], there was overall an improvement students performance specifically in problem solving and social skills, required to perform all the designed authentic tasks and activities (listed in Table I). Students learned to understand the importance of quality inspection, the flexibility of planning, and effective collaboration through practical experience.
- within the context of first SE course, among the problem solving skills, students struggled most with authentic tasks which required a high level of change in their working environment/process as well the tasks that required attention throughout the project lifecycle, i.e. quality inspection tasks. Recall from Figure 5d and Table IV, teams' quality understanding skills has received the lowest score in this category with *mean=5.25*. The score of teams' adaptability to new working/social environments comes as the next lowest score with *mean=6.09*.
- Among the authentic tasks and activities that have been applied in this study (for a first SE course), we believe that $A7$, i.e. changing the programming paradigm, may require the more traditional approach of guidance and tutoring rather than 'real world' experience. Given that the students have limited experience with programming (i.e. specifically android programming in this course), changing the programming environment might negatively impact students learning experience and is, contrary to our expectation, less useful in teaching *adaptability* and *planning* skills.
- We applied *"Learning by reflection"* approach in this course to help students improve their learning experience, and asked all the students at the end of each semester to indicate their level of agreement about their perception of learning, we asked them the question: *" I learned a lot in this course"*. On a scale of 1 to 10, students scored their learning experience as: (1) first semester *Mean=6.22, SD=0.99*, (2) second semester *Mean=6.64, SD=0.77*, and (3) third semester *Mean=6.53, SD=0.92*. This evaluation ran by the university and the collected data was anonymous.
- Providing students with the vague and inadequate specification about the course project, made them realize that they have to continuously communicate and negotiate with the clients and that eventually pushed them to struggle and try to improve their social skills.

These results show that the application of the AA model did help students in improving their problem solving and social skills required for software engineering projects.

For future work, we encourage other instructors to explore the different levels of authenticity that should be applied in courses offered in either the initial semesters or final semester of the degree programs. Following the results of this longitudinal study, the design and implementation of authentic assessment requires more flexibility and supervision in the first SE courses, and cannot follow the pattern similar to Capstone projects or industry-based learning units.

[1] Teams whose members faced with intra-team communication problems

References

[1] M. Shaw, “Software engineering education: A roadmap,” in *Proceedings of the conference on The future of Software Engineering*, ACM, 2000, pp. 371–380.

[2] ——, “Prospects for an engineering discipline of software,” *IEEE Software*, vol. 7, no. 6, pp. 15–24, 1990.

[3] C.-Y. Chen and P. P. Chong, “Software engineering education: A study on conducting collaborative senior project development,” *Journal of systems and Software*, vol. 84, no. 3, pp. 479–491, 2011.

[4] D. W. Johnson, R. T. Johnson, and K. A. Smith, “Cooperative learning returns to college what evidence is there that it works?” *Change: the magazine of higher learning*, vol. 30, no. 4, pp. 26–35, 1998.

[5] J. Kay, M. Barg, A. Fekete, T. Greening, O. Hollands, J. H. Kingston, and K. Crawford, “Problem-based learning for foundation computer science courses,” *Computer Science Education*, vol. 10, no. 2, pp. 109–128, 2000.

[6] H Van Vilet, “Reflections on software engineering education,” *IEEE software*, vol. 23, no. 3, pp. 55–61, 2006.

[7] W. Maalej and D. Pagano, “On the socialness of software,” in *Dependable, Autonomic and Secure Computing (DASC), 2011 IEEE Ninth International Conference on*, IEEE, 2011, pp. 864–871.

[8] M Bano, D Zowghi, A Ferrari, P Spoletini, and B Donati, “Learning from mistakes: An empirical study of elicitation interviews performed by novices,” in *International Requirements Engineering Conference*, 2018.

[9] M. C. Bastarrica, D. Perovich, and M. M. Samary, “What can students get from a software engineering capstone course?” In *Software Engineering: Software Engineering Education and Training Track (ICSE-SEET), 2017 IEEE/ACM 39th International Conference on*, IEEE, 2017, pp. 137–145.

[10] N. R. Mead, “Software engineering education: How far we’ve come and how far we have to go,” *Journal of Systems and Software*, vol. 82, no. 4, pp. 571–575, 2009.

[11] S. C. dos Santos and F. S. F. Soares, “Authentic assessment in software engineering education based on PBL principles a case study in the telecom market,” in *2013 35th International Conference on Software Engineering (ICSE)*, 2013, pp. 1055–1062.

[12] J. R. Savery and T. M. Duffy, “Problem based learning: An instructional model and its constructivist framework,” *Educational technology*, vol. 35, no. 5, pp. 31–38, 1995.

[13] G. X.-L. Tai and M. C. Yuen, “Authentic assessment strategies in problem based learning,” in *Proceedings of ASCILITE*, Citeseer, 2007.

[14] R. Dawson, “Twenty dirty tricks to train software engineers,” in *Proceedings of the 22nd international conference on Software engineering*, ACM, 2000, pp. 209–218.

[15] E. O. Navarro, “On the role of learning theories in furthering software engineering education,” in *Instructional design: Concepts, methodologies, tools and applications*, IGI Global, 2011, pp. 1645–1666.

[16] J. Dewey, “Democracy and education, new york,” *Macmillan). If you see john Dewey, tell him we did it. Educational Theory*, vol. 37, no. 2, pp. 145–152, 1916.

[17] J. Lave, *Cognition in practice: Mind, mathematics and culture in everyday life*. Cambridge University Press, 1988.

[18] J. S. Bruner, *On knowing: Essays for the left hand*. Harvard University Press, 1979.

[19] R. Schank, *Virtual Learning. A Revolutionary Approach to Building a Highly Skilled Workforce.* ERIC, 1997.

[20] D. A. Schön, *Educating the reflective practitioner*. Jossey-Bass San Francisco, 1987.

[21] J. D. Tvedt, R. Tesoriero, and K. A. Gary, “The software factory: Combining undergraduate computer science and software engineering education,” in *Proceedings of the 23rd international conference on Software engineering*, IEEE Computer Society, 2001, pp. 633–642.

[22] M. Moore and C. Potts, “Learning by doing: Goals and experiences of two software engineering project courses,” in *Conference on Software Engineering Education*, Springer, 1994, pp. 151–164.

[23] T Germain, P. N. Robillard, and M. Dulipovici, “Process activities in a project based course in software engineering,” in *Frontiers in Education, 2002. FIE 2002. 32nd Annual*, IEEE, vol. 3, 2002, S3G–S3G.

[24] A. Goold and P. Horan, “Foundation software engineering practices for capstone projects and beyond,” in *Proceedings 15th Conference on Software Engineering Education and Training (CSEE T 2002)*, 2002, pp. 140–146.

[25] J. Herrington and A. Herrington, “Authentic assessment and multimedia: How university students respond to a model of authentic assessment,” *Higher Education Research & Development*, vol. 17, no. 3, pp. 305–322, 1998.

[26] J. T. Gulikers, T. J. Bastiaens, and P. A. Kirschner, “A five-dimensional framework for authentic assessment,” *Educational technology research and development*, vol. 52, no. 3, p. 67, 2004.

[27] S. C. dos Santos, F. Furtado, and W. Lins, “Xpbl: A methodology for managing pbl when teaching computing,” in *Frontiers in Education Conference (FIE), 2014 IEEE*, IEEE, 2014, pp. 1–8.

[28] S. C. dos Santos, “Pbl-see: An authentic assessment model for pbl-based software engineering education,” *IEEE Transactions on Education*, vol. 60, no. 2, pp. 120–126, 2017.

[29] A. M.C. A. Oliveira, S. C. dos Santos, and V. C. Garcia, “Pbl in teaching computing: An overview of the last 15 years,” in *Frontiers in Education Conference, 2013 IEEE*, IEEE, 2013, pp. 267–272.

[30] Z. Shakeri Hossein Abad, S. Moazzam, C. Lo, T. Lan, E. Frroku, and H. Kim, “Loud and interactive paper prototyping in requirements elicitation: What is it good for?” In *2018 IEEE 7th International Workshop on Empirical Requirements Engineering (EmpiRE)*, 2018, pp. 16–23.

[31] Z. Shakeri Hossein Abad, S. D. V. Sims, A. Cheema, M. B. Nasir, and P. Harisinghani, “Learn more, pay less! lessons learned from applying the wizard-of-oz technique for exploring mobile app requirements,” in *2017 IEEE 25th International Requirements Engineering Conference Workshops (REW)*, 2017, pp. 132–138.

[32] F. Ahmed, L. F. Capretz, and P. Campbell, “Evaluating the demand for soft skills in software development,” *It Professional*, vol. 14, no. 1, pp. 44–49, 2012.

[33] S. T. Acuña and N. Juristo, “Assigning people to roles in software projects,” *Software: Practice and Experience*, vol. 34, no. 7, pp. 675–696, 2004.

[34] M. B. Wilk and R. Gnanadesikan, “Probability plotting methods for the analysis for the analysis of data,” *Biometrika*, vol. 55, no. 1, pp. 1–17, 1968.

[35] G. R. Gibbs, *Qualitative data analysis: Explorations with NVivo*. Open University, 2002.

[36] M. Usman, E. Mendes, F. Weidt, and R. Britto, “Effort estimation in agile software development: A systematic literature review,” in *Proceedings of the 10th International Conference on Predictive Models in Software Engineering*, ACM, 2014, pp. 82–91.

[37] V. Mahnič and T. Hovelja, “On using planning poker for estimating user stories,” *Journal of Systems and Software*, vol. 85, no. 9, pp. 2086–2095, 2012.

[38] C. Wohlin, M. Höst, and K. Henningsson, “Empirical research methods in software engineering,” in *Empirical methods and studies in software engineering*, Springer, 2003, pp. 7–23.

[39] J. R. Landis and G. G. Koch, “The measurement of observer agreement for categorical data,” *biometrics*, pp. 159–174, 1977.