Information and Software Technology 158 (2023) 107189

Contents lists available at ScienceDirect

Information and Software Technology

journal homepage: www.elsevier.com/locate/infsof

Application of Project-Based Learning to a Software Engineering course in a hybrid class environment

Edgar Ceh-Varela [a,*], Carlos Canto-Bonilla [b], Dhimitraq Duni [a]

[a] Department of Mathematical Sciences, Eastern New Mexico University, Portales, NM, USA
[b] Department of Information Technologies, Universidad Tecnologica Metropolitana, Merida, Yucatan, Mexico

ARTICLE INFO

Keywords:
Project-Based Leaning
Software Engineering
Hybrid environment

ABSTRACT

Context: This paper centers on Project-Based Learning (PBL). In PBL, the student is now the center of the whole teaching and learning process, while the instructor‘s role is now of a facilitator presenting to the students the resources and guidance to solve the given problem. Most existing studies, apply PBL to courses having in-person students.
Objective: The paper presents the application of a PBL approach to a Software Engineering (SE) course having a hybrid class environment (i.e., online and in-person students). The main objective of this paper is to analyze the students' attitudes after experiencing working on a real-life problem as part of our PBL approach in a hybrid class environment.
Methods: We propose a *relaxed plan-based* software development model as basis for guiding the project execution. At the end of the course, we applied a survey to the students to evaluate their experience in the course.
Results: We obtained the answers of 70.8% of students taking a SE course. With these answers, we could measure the students' perception of using PBL in a SE course and how this strategy helped them to gain soft and hard skills in software development. We divided the answers for their analysis into different categories: soft skills, technical skills, learning experience, and other results. Moreover, we compare the performance of the teams and students based on their type (i.e., online and in-person).
Conclusion: We found qualitative differences in the experience of online and in-person students. Based on our experience with this study, we provide guidelines for applying PBL in a hybrid environment. Overall, our study has demonstrated a positive contribution in supporting teaching SE using a PBL in a hybrid class environment.

1. Introduction

Courses in universities are traditionally developed using lectures and following the syllabus for the course. Moreover, students must take several assignments and tests to evaluate their learning. Therefore, the instructor is the center of the process by organizing and planning the students' learning, while the students are always on a second plane [1].

New student-centered teaching methodologies are being used in higher education. This paper is centered on a project-based learning (PBL) methodology [1–3]. As its name implies, PBL uses projects related to real-world problems to learn specific subjects and attain some generic skills to solve those problems. Instead of only lectures, like in traditional learning approaches, the PBL learning process begins with the problem. The objective is to develop a strategy for teaching students how to learn rather than specific abilities and emphasize relevant skills such as problem-solving, communication, teamwork, leadership, critical thinking, creativity, and risk assessment [4–6]. Therefore, in PBL, the student is the center of the whole teaching and learning process, while the instructor's role is now of a facilitator presenting the resources and guidance to solve the given problem.

PBL has been used to teach courses in multiple disciplines [7–10]. PBL is an essential component in teaching engineering courses [11,12]. For example, during the COVID-19 pandemic, many universities were forced to implement new PBL approaches to provide remote laboratories to their students [13,14]. The results of applying PBL in different engineering fields have been proven to be positive [15,16].

Similarly, PBL has been used in Software Engineering (SE) courses. A SE course covers the complete software development lifecycle. It involves learning computer science's analytical and descriptive tools (e.g., algorithms, debugging tools, modeling, and simulation tools) and

* Corresponding author.
E-mail addresses: eduardo.ceh@enmu.edu (E. Ceh-Varela), carlos.canto@utmetropolitana.edu.mx (C. Canto-Bonilla), dhimitraq.duni@enmu.edu (D. Duni).

https://doi.org/10.1016/j.infsof.2023.107189
Received 10 November 2022; Received in revised form 21 February 2023; Accepted 23 February 2023
Available online 6 March 2023
0950-5849/Published by Elsevier B.V.

the rigor that the engineering domain provides to software development, all while working in a collaborative context.

Several studies [17–19] have shown the efficacy of using a PBL approach in a SE course. However, these studies show the results of courses with in-person students. This paper presents the application of a PBL approach to a SE course having a hybrid class environment (i.e., online and in-person students). The main objective of this paper is to analyze the students' attitudes after experiencing working on a real-life problem as part of our PBL approach in a hybrid class environment.

The remainder of this paper is organized as follows: Section 2 presents the background literature related to PBL and Software Engineering. Section 3 reviews several works related to applying PBL to SE courses. Section 4 details the proposed method. The results of applying a PBL approach to a hybrid SE class are in Section 5. Finally, our conclusions are in Section 6.

2. Background

2.1. Project-based learning

Project-Based Learning (PBL) is a learning model in which students plan, implement, and evaluate projects that have real-world applications beyond the classroom [20]. The PBL model has been greatly accepted in higher education [2,3]. PBL involves forming teams of students with different profiles, with different hard and soft skills, who work together to carry out projects to solve real problems. There are several hard and soft skills students can learn to solve problems. For example, hard (i.e., technical) skills related to a specific job or task, such as programming or data analysis. Similarly, soft skills include communication, teamwork, problem-solving, and critical thinking.

These differences in profiles, offer excellent learning opportunities and prepare students to work in a diverse global environment. Similarly, PBL demands in-depth learning, focusing on real-world problems and challenges, and relying on problem-solving, decision-making, and investigative skills. In PBL, hard and soft skills can be assessed through various methods, such as written or oral reports, group presentations, and peer evaluations [2].

There are many benefits that PBL offers to the learning process since it urges students to think and act based on the design of a project, develop a plan with defined strategies, provide a solution to a question and not just meet curricular objectives. Students learn different techniques for solving problems by contacting people from different points of view and skills. For example, people with different coding skills and personal attributes not directly related to a specific job or task. They learn to learn from each other and learn how to help their peers learn. Moreover, students can use various resources and have more extended autonomy than in a conventional class.

In PBL, the instructor is a facilitator, presenting students with resources and directions to conduct their projects. The instructor is not the primary source of access to information. Educators must act as learning guides and allow students to acquire autonomy and responsibility. Consequently, students who perceive greater autonomy tend to have more positive experiences and perceive this methodological approach as more beneficial [21] and more engaging than conventional instructional practices [19].

2.1.1. PBL main characteristics

PBL is a teaching approach that involves having students work on real or simulated problems to learn new concepts and skills [22]. Some of the main characteristics of PBL include the following:

- *Real-world relevance*: PBL projects should have a real-world application and be related to the workplace tasks that professionals would perform. This helps students see the relevance of the material they are learning and makes it more meaningful and motivating. Moreover, the final solutions' evaluation must also be integrated into learning to match the professional world's quality assessment (see Section 4.7).
- *Broad scope*: PBL projects should be broad in scope, with minimal specifics, allowing students to work on establishing the project's list of tasks and subtasks. This encourages students to think critically and creatively and to develop problem-solving skills.
- *Collaboration and interdisciplinary perspectives*: PBL often involves students working in groups to solve problems, which can help develop collaboration and communication skills. Additionally, PBL projects should require interdisciplinary perspectives, allowing students to take on various roles and develop insights beyond a single subject or domain.
- *Long duration*: Completing PBL projects takes significant time and effort, and the students cannot complete them in a single class session. This allows students to delve deeply into the material and apply their knowledge and skills to a more complex and realistic problem.
- *Practical utility*: The final products (e.g., software, websites) produced through PBL should have practical utility rather than just formative value. This helps students see the real-world impact of their work and allows them to develop skills that are relevant and applicable in the professional world.

Overall, PBL is an effective teaching approach that encourages active learning, collaboration, problem-solving, and real-world skills development.

2.2. Project-based learning and software engineering

PBL has been adapted for different instructional domains, such as medicine [7,8], electronic engineering [9], environmental engineering [6], and industrial engineering [10]. Similarly, PBL can be an effective way to teach engineering concepts and abilities as it helps students develop essential skills such as problem-solving, collaboration, and communication and encourages active learning and creativity. Moreover, different works show the benefits of using PBL in remote learning environments [13,14].

PBL has also been used and adapted for SE courses [17–19]. In this paper, we are interested in applying PBL to SE. Software Engineering can be defined as:

"The disciplined application of engineering, scientific, and mathematical principles and methods to the economical production of quality software" [23].

A course of SE is complex to teach. It mainly combines non-technical topics (e.g., interviewing, management, quality) that students do not relate to as quickly as with the other technical subjects from engineering [19,24]. Therefore, a SE course should prepare students with technical skills related to software development and, at the same time, help them to develop soft skills.

Although several studies exist on how a PBL approach is applied to a SE course (please refer to Section 3), one characteristic in common is that the majority of them form teams with students taking the course in person. Only few studies work with teams formed by remote students. Our scenario is different. The primary objective of this study is to assess the efficacy of a PBL approach in a SE course that includes both in-person and online students. Therefore, our scenario presents more challenges to the in-person and online students working on their projects as a team. Some of these challenges are:

1. Communication barriers: In-person and online students may have different communication styles and preferences, which can make it difficult for them to communicate effectively with one another. Additionally, online students may have difficulty participating in group discussions or real-time communication, which can lead to misunderstandings inside the team.
2. Time zone differences: If in-person and online students are in different time zones, it can be difficult for them to coordinate meetings or work on group projects together.

3. Different learning styles: In-person and online students may have different learning styles and prefer different learning materials and activities. This can make it difficult for them to work together effectively on group projects or assignments.
4. Different levels of engagement: In-person and online students may have different levels of engagement with the course material, which can lead to misunderstandings or conflicts when working together.
5. Different technological abilities: In-person and online students may have different levels of familiarity and comfort with technology, which can make it difficult for them to work together effectively on projects that require the use of technology.

3. Related work

As we have mentioned, Project-based Learning (PBL) has gained traction in education. However, with the growing prevalence of remote learning, PBL environments have become increasingly diverse, with some classes consisting of both in-person and remote students. This scenario has presented new challenges and opportunities for instructors, as they must now design PBL activities that can accommodate both types of students and facilitate effective collaboration. This section will review the related work on applying PBL to SE courses in traditional in-person and remote learning environments.

3.1. Traditional in-person environment

In [25], the authors use PBL as a teaching method for SE students, with the aim of improving their understanding of concepts. The authors introduced factors that should be present in ‘‘pure’’ PBL and described an implementation of PBL in a class focused on understanding information flows in SE. The paper analyzes the strengths and weaknesses of this implementation and suggests modifications for future classes to enhance students' learning and experience. Researchers in another study [26], used PBL in a software quality module within a SE course. The paper presents the positive outcomes of using PBL and highlights the need for reproducibility and generalization of PBL in SE education. The authors argue that the success of this approach depends on the development of PBL problems and concepts specific to the discipline and the usefulness of the outcome in the industry. A seven-year case study is presented in [27]. The researchers analyzed SE education using a blended approach of traditional lectures, textbooks, lecture notes, and a web-based platform to support a cooperative PBL environment. The researchers found that the main issue for the students was the novelty of the educational approach. Students required assistance in enhancing their projects with distinct educational objectives and favored the widely favored group work methodology

The study presented in [28], describes a survey that investigates students' perceptions of using PBL in an introductory SE course. The study compared the responses of 32 undergraduate students enrolled in a PBL course to 17 students in a traditional teacher-centered course with a software development project. The results showed a positive reception of PBL and an increased perception of the contribution of practical software development assignments to learning specific SE topics in the PBL course compared to the traditional course. In [29], the researchers present a framework for the implementation of PBL in SE education, with industry collaboration, to bridge the gap between academic and professional software development. The paper proposes establishing a SE lab in academic institutions with the help of industries. Students can gain complete knowledge of concepts, skills, and principles through real-time project execution by universities and industries. Similarly, in [30], researchers investigate the impact of adding the final stage of the experiential learning cycle to a PBL approach in SE courses. A quasi-experiment was conducted on three courses, and results showed that the modified PBL approach improved students' motivation and experience but had a limited impact on their learning outcomes. Likewise, in [19], PBL was part of an active learning strategy. This strategy was used to improve students' learning experiences in three Computer Science courses (i.e., Information Systems Development, Data Structures, and Web Languages and Technologies) that address subjects related to SE. Furthermore, [31] assesses the PBL approach's efficacy in improving SE students' skills in three essential domains: practical, knowledge, and soft skills. Their research looks at project planning, requirement analysis, software design, development, presentation, and other parts of SE.

In [17], SE is taught using a game-centric PBL approach. The students apply the concepts from SE to create different modules for a computer game. A similar study is conducted in [32]. The authors present their experience incorporating PBL and gamification elements into an introductory SE course. They identified 17 guidelines for instructors to follow. The paper provides insights and guidelines that can be useful for educators. Likewise, [18] presents a methodology based on PBL to enhance the learning effectiveness in SE by implementing software factories. The methodology allows students to get immersed in practical software development projects from actual clients, supported by procedures, roles, and metrics to manage the outcomes.

All the studies presented above involved students in a traditional, in-person classroom setting. The instructors formed teams among these students to tackle the projects presented.

3.2. Remote learning environment

This section highlights the application of PBL in SE courses with students participating remotely. In [33], the researchers report the results of applying a PBL approach to a joint Mongolian-German team-teaching project aimed at teaching global SE in a remote environment. The project involved students with no prior experience in international projects, who were initially confused and frustrated by cultural differences. Through their involvement in the project, the students gained problem-solving skills and an appreciation for dealing with cultural assumptions. A similar project is presented in [34]. The researchers describe a remote, team-teaching experience between universities in Mexico and Germany. Students from each university worked together on a group project using video conferences, chats, and cloud-based project management tools. The project aimed to compare the effectiveness of PBL versus traditional instructor-based lectures in teaching soft intercultural skills. The results showed that PBL was much more challenging but positively impacted the students' performance and learning success compared to traditional lectures. Likewise, in [35], the paper presents a global SE class taught by Ritsumeikan University in Japan and the Technical University of Nuremberg in Germany. The course uses a PBL approach to provide students with experience working in international software development teams. The instructors found regular discussion and counseling important, as was enforcing deadlines and reconciling individual perceptions of team members. The class faced challenges with video conferencing software and coordinating the classes, but these challenges were overcome. The researchers identified trust as the most crucial factor for the success of the course.

Our research objective, as stated in Section 2, is to assess the success of using a Problem-Based Learning method in a Software Engineering course that brings together traditional and online students in collaborative teams. This scenario presents unique challenges, as it requires both in-person and online team members to work together despite differences in their level of interaction with the instructor. While in-person students benefit from face-to-face interaction with the instructor, their online counterparts rely primarily on video lectures and collaboration with their team members for their learning experience.

4. Methodology of our PBL approach

4.1. Software development model selection

Several software development models can be used to teach a SE course [36], such as Cascade [37], Spiral [38], Prototype [39], V-model [40], and Agile [41]. The best model for a given project will depend on the specific needs and goals of the project. Familiarity with a range of software development models allows students to develop a broad understanding of the different approaches that can be taken to software development. It is important for students to understand the strengths and limitations of each model and to be able to choose the appropriate model for a given project.

Some traditional models have been replaced by more flexible and iterative approaches, such as Agile methodology. However, while the Agile model can be an effective approach to software development, new engineers may need to gain the necessary experience to apply the Agile model effectively. They may benefit from being introduced to more traditional models before transitioning to Agile.

Although students will learn about all the most common types of models during the SE course, including Agile, we designed a *relaxed plan-based model* (see Fig. 1). This model uses the following traditional models as its basis:

- Cascade model. This model is simple and easy to understand and use for the students.
- Spiral model. It is a flexible model. Changes in requirements and functionality additions can be made at later phases and incorporated accurately.
- Prototype model. Ideal for an online system. Incremental deliverables are presented to the customer. Customer feedback is essential for creating the final product.

More conventional plan-based strategies promote knowledge exchange mainly through documentation [42]. The proposed model (i.e., relaxed plan-based) focuses on combining theory and practice of Software Engineering and Project Management in various activities. These activities include project estimates, requirements elicitation, modeling, design, and software development. Therefore, the main goal of our model is to expose SE students to the realities of working on a software project in a corporate setting.

4.2. Project description

Asking the students to work on the same problem helps with comparison and consistency. As in other works [19,43], in our study, the students worked on solving a particular problem. For a real-life experience, the PBL project consisted of creating a "Theatre Management System". A web system to help a theatre with the sale of tickets. The web system requires the following modules:

- Client Management
- Theatre Management
- Ticket Selling

The course instructor played the role of the theatre owner and presented a general description of the requirements. The length of the SE course is 16 weeks. Therefore, the project was divided into five stages.

Table 1 presents the different stages of the project. This table also shows the activities and the products derived from each stage.

Table 1 also indicates the products related to each base model used in our *relaxed plan-based model*.

In the following sections, we present the description of these stages.

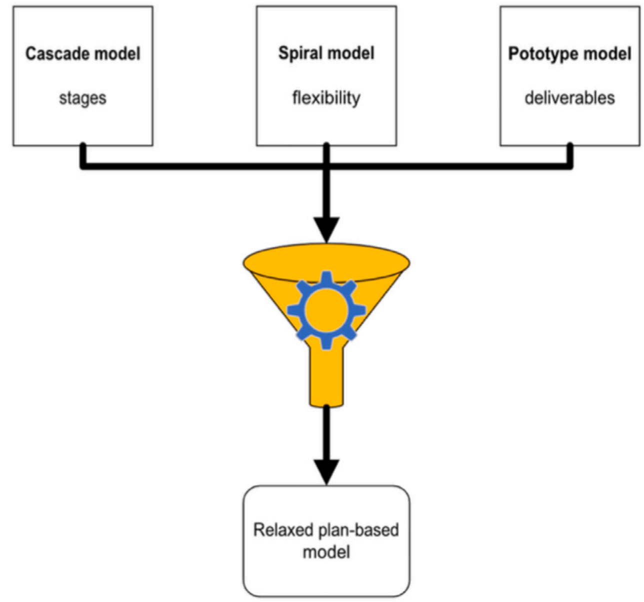


Fig. 1. The relaxed plan-based software development model used for the project.

4.3. Stage 0: Team formation

In general, teams can be homogeneous or diverse in composition. Homogeneous suggests that the team members have similar profiles, whereas diverse means that the team members have considerable differences in their profiles [44].

In most works using PBL [17–19,24,31], the instructors determine the number of teams and assign the students to these teams. These instructors have the advantage of having students physically in the classroom. However, our context is different. The SE course has a section for asynchronous online students (n = 21) and a section for in-person students (n = 5). Therefore, this situation makes the team formation a more challenging tasks than in previous works (see Section 2.2). Given the distribution of our students in two sections, we decided that teams could be formed with students from the in-person section, the online section, or a mix of both.

The instructor explained the project requirements and expected outcomes on the first day of the course. Then, we let students form their teams, allowing them to pick team members from any of the two sections (i.e., online and in-person). There are many factors that students might consider when choosing team members for a project or assignment. For example, skills and expertise, compatibility, availability, and trustworthiness. Ultimately, the process of choosing team members may involve a combination of these and other factors and will depend on the specific needs and goals of the project. We consider that having this informative session before forming teams helped them measure their strengths and look for teammates that can complement their skills.

The size of each team was restricted to up to four members. In total, seven teams were formed. Each team worked independently to solve the given problem. Similarly, as a requirement for each team, a member must play the role of team leader for each project stage. In this way, the team leader could have real-life experience managing a team of programmers and being responsible for keeping the project on track. Moreover, each team needed to set up a GitHub[1] repository for a real-life experience working on a collaborative software development platform.

Table 1
Course project schedule.

Project	Week	Tasks	Products
Stage 0	1–2	Team formation	Document with the list of team members
Stage 1	3–6	Requirements Specification	Requirement Description Document (RDD)[a] Pert diagram Critical Path Analysis (CPA) Gantt diagram Cost estimation Software Requirement Specification (SRS)[a] Use case diagrams[a] Class diagrams[a] Sequence diagrams[a]
Stage 2	7–10	Design Specification	Software Design Document (SDD)[a] Component diagrams[a] Updated Use case diagrams[b] Updated Class diagrams[b] Updated sequence diagrams[b] Updated Gantt diagrams[b] Screenshots of the system's progress[c]
Stage 3	11–13	Risk Analysis and Test Specification	Software Test Plan[a] Updated Use case diagrams[b] Updated Class diagrams[b] Updated sequence diagrams[b] Updated Gantt diagrams[b] Screenshots of the system's progress[c]
Stage 4	14–16	System demonstration to stakeholders	Final project report Test reports[a] Source code Final prototype[c]

[a]Cascade model.
[b]Spiral model.
[c]Prototype model.

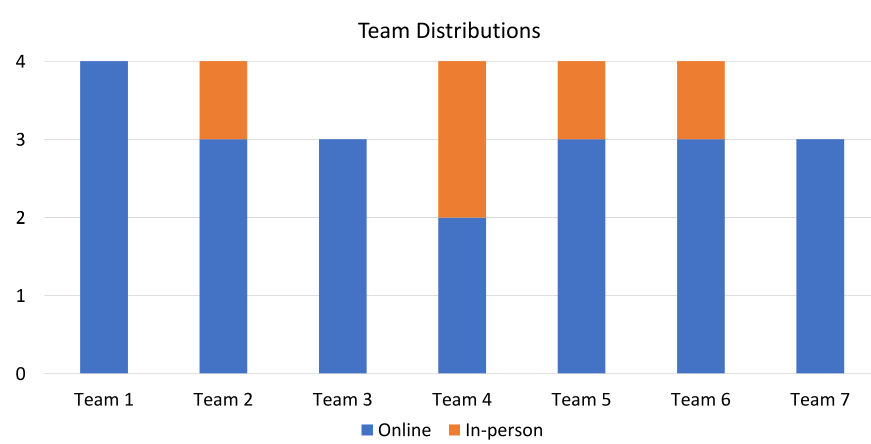


Fig. 2. Distribution of students from different sections in each team.

Fig. 2 shows the distribution of students in those teams. From this Figure, we can observe that more than half of the teams (n = 4) are formed by students from both sections (i.e., in-person and online).

4.4. Stage 1: Requirements specification

For this stage, the teams defined and scheduled their initial tasks for the project. Similarly, the teams prepared estimations for the effort and the costs of working on the project. The teams created different documents and diagrams to document this information. Moreover, in this stage, each team leader met with the customer (i.e., the instructor) to obtain more information about the functional and non-functional project requirements. The preferred method to obtain more information was by interviewing the customer with predefined questions prepared by the team.

At the end of this stage, the teams had more knowledge about the requirements for the project. The output of this stage was a series of documents obtained by the requirements engineering process. During this stage, the students received complimentary lectures, assignments, and quizzes to strengthen their knowledge about the techniques they needed to use for this stage. The topics covered for this stage are software processes, project planning (i.e., Gantt and Pert charts, CPM,

[1] http://www.github.com

COCOMO), system modeling (i.e., UML), and requirements engineering.

4.5. Stage 2: Design specification

In this stage, the teams defined the system's architecture and the interaction of the different components of the system. The teams determined the design for their implementations, such as the database schema, the classes to implement, and the platforms to use. One output for this stage was a document (see Table 1) with all the information regarding the planned architecture and components. Additionally, the team leader was responsible for updating previous documents to reflect the new information from this stage. For example, the project schedule was adjusted to reflect the new activities created during this stage.

Moreover, the team presented to the customer some progress in constructing the system. This activity had two intentions. First, to motivate the teams to work on developing their solution. Second, to simulate presenting the project progress to the client and receiving feedback. As in the previous stage, the information about the activities and tasks for this stage was presented in complementary lectures and assignments. The topics covered in this stage are software design and implementation.

4.6. Stage 3: Risk analysis and test specification

For this section, the teams were involved in the system's coding. Similarly, the teams prepared a test plan for this stage to verify and validate that the system works correctly and covers the functional and non-functional requirements. As in the previous stage, the team leader was responsible for updating previous documents with information from this stage. Likewise, the teams presented new progress to the client to receive feedback. The complementary lectures and assignments covered in this stage are software testing and software maintenance.

4.7. Stage 4: Project demonstration

The last stage of the project consisted of demonstrating the functional system to the client. The team leader guided the client in this demonstration to validate the project's requirements. To present a more real-life experience to the teams, we invited an external instructor to take the role of an additional stakeholder. This instructor has vast experience in web systems development, so the feedback was enriched. The work for this stage was complemented with lectures corresponding to software quality and configuration management.

The team delivered a final project report to finalize this stage, including the results of the different validation and verification tests.

4.8. Evaluation of the PBL approach

We obtained information from students to assess the pertinence of our PBL approach. Based on [24,28,45], we created a survey with *20* mandatory Likert scale questions and *one* mandatory multiple-choice question to better understand their perceptions of learning SE under a PBL paradigm. All students received the survey via e-mail. The questions for this survey are listed in Table 2.

5. Results

We sent the survey described in the previous section to all the students taking the course (n = 26). We collected the answers anonymously using an online survey platform. We followed a survey protocol approved by the University's "Human Subjects Committee". The students could accept or decline their participation in the survey. Two of the students withdrew from the course for different reasons. Only 17 students out of the remaining 24 answered the survey (70.8%). We consider this participation as high, given that in general, participation in surveys is lower among college students [46]. For this study, some of the reasons not all 24 students participated could be because the survey was sent the last week of the semester. This is the week when the students are busy with their final exams and projects. Similarly, as it was an anonymous survey, we could not incentivize students to participate. If there is no tangible reward or incentive for participating in the survey, students may be less likely to take the time to do it.

As mentioned in Section 2.2, a SE course should prepare students to acquire soft and hard skills. Therefore, we used our survey to measure if our PBL strategy helped them with this preparation. The following sections describe the results obtained from our survey. We separate these results into different subsections: Soft Skills, Technical Skills, Learning Experience, and Other Results.

5.1. Soft skills

During the sessions with the students, we clarified the differences between a project manager and a team leader. The project manager is responsible for effective planning and ensuring that the team functions at its best. A team leader is in charge of the day-to-day operations of the development effort, emphasizing the team's efforts and core focus.

We evaluated how the students perceived the role of the Team Leader and Project Manager. These roles are essential in the project because the student must apply different soft skills within a software development team [47]. For example, the Team Leader should have interpersonal, motivational, and conflict resolution skills to maintain communication within the team and with the customer. We put more emphasis on the role of Team Leader given each student needed to play that role for each stage.

Fig. 3(a) shows that 77% of the students agreed and strongly agreed that the role of the Team Leader important to the project. Interestingly, 100% of the in-person students strongly agree with the importance of this role. Similarly, Fig. 3(b) shows a similar percentage of agreement (76%) for the role of Project Manager. As in Fig. 3(a), a high percentage of the in-person students (75%) strongly agree with the question asked.

Effective communication is one of the team leader's most important soft skills [48]. *Communication* is a crucial ability that students must learn and practice [49]. Therefore, we analyze in Fig. 4 the students' perceptions. Fig. 4(a), Fig. 4(b), and Fig. 4(c) show that 95%, 94%, and 95% of the students agreed and strongly agreed that communication skills are essential for the Team Leader. These qualities must be used in communication within the team, especially with project stakeholders, to identify their requirements. From Fig. 4, we can see that none of the students disagrees with the importance of having these communication skills. Moreover, Fig. 4(a) shows that 100% of all in-person students strongly agreed with this statement.

Overall, we can conclude that assigning the role of "Team Leader" in a real-life project helped the students to (i) understand the implications and responsibilities of the role and (ii) see the importance of soft skills (i.e., communication skills) in a project.

5.2. Technical skills

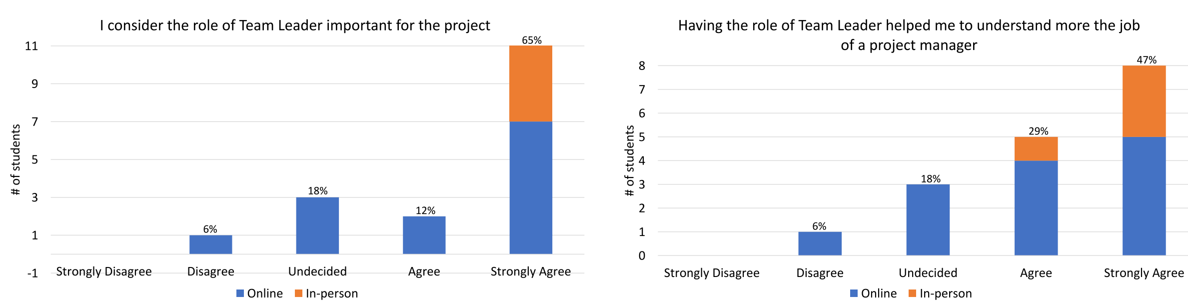
A course about SE needs to provide some technical skills regarding different aspects of software development. Therefore, a PBL approach must allow the students to work on real-life tasks.

Fig. 5(a) shows that 94% of the students agreed that working on a real-life project helped them understand the tasks needed. Only one student from the in-person section disagreed with this point.

Distributed projects face numerous challenges. For example, distributed team members have limited or no face-to-face interactions [50]. For our SE course, students formed teams with students not present in the classroom (see Fig. 2) and with different profiles. Therefore, the students needed to adapt their work practices to a remote and distributed working environment. Fig. 5(b), 5(c), and 5(d) show

Table 2
Survey questions.

#	Question	Type
1	I consider the role of Team Leader important for the project	Likert
2	Having the role of Team Leader helped me to understand more the job of a project manager	Likert
3	As a Team Leader, the contact with the client is important	Likert
4	As a Team Leader, the contact with the members of the team is important	Likert
5	Communications skills are essential for a Team Leader	Likert
6	Reviewing the project deliverables with my teammates has helped me improve my understanding about the tasks needed in a real-life software development project	Likert
7	Being part of a team with members working remotely prepared me to be able to participate in real-life software development projects	Likert
8	Using a collaborative platform for software development, like Github, has helped me to understand how development teams work in real-life projects	Likert
9	Having team members with different set of skills helped me to learn new strategies for solving a software development problem	Likert
10	I believe that using a project-based learning approach influenced my learning process in this course	Likert
11	The experience in a real-life project helped me to understand the work done in a software development company	Likert
12	Working in a real-life project is important to understand the concepts of Software Engineering	Likert
13	I feel satisfied with the final product delivered in the Software Engineering course	Likert
14	I distinguish which elements belong to each phase of the software development life cycle	Likert
15	The project has helped me to better understand the professional work of a software engineer	Likert
16	The topic of the project was interesting to me	Likert
17	The feedback and information received from the instructor and teammates have helped me to improve	Likert
18	I believe practical assignments, focused on the development of a software project, are fundamental for developing new skills and learning about Software Engineering	Likert
19	My confidence to develop a software project at the beginning of the course was ______	Lkert
20	My confidence to develop a software project at the end of the course was ______	Likert
21	Rate how much the Project contributed to learning the following topics a. Software requirements b. Software design and analysis c. Software implementation and construction d. Software testing e. Software Quality	Likert



(a) Importance of Team Leader

(b) Understanding the job of Project Manager

Fig. 3. Understanding of Team Leader and Project Manager roles. Results for questions #1 and #2, respectively.

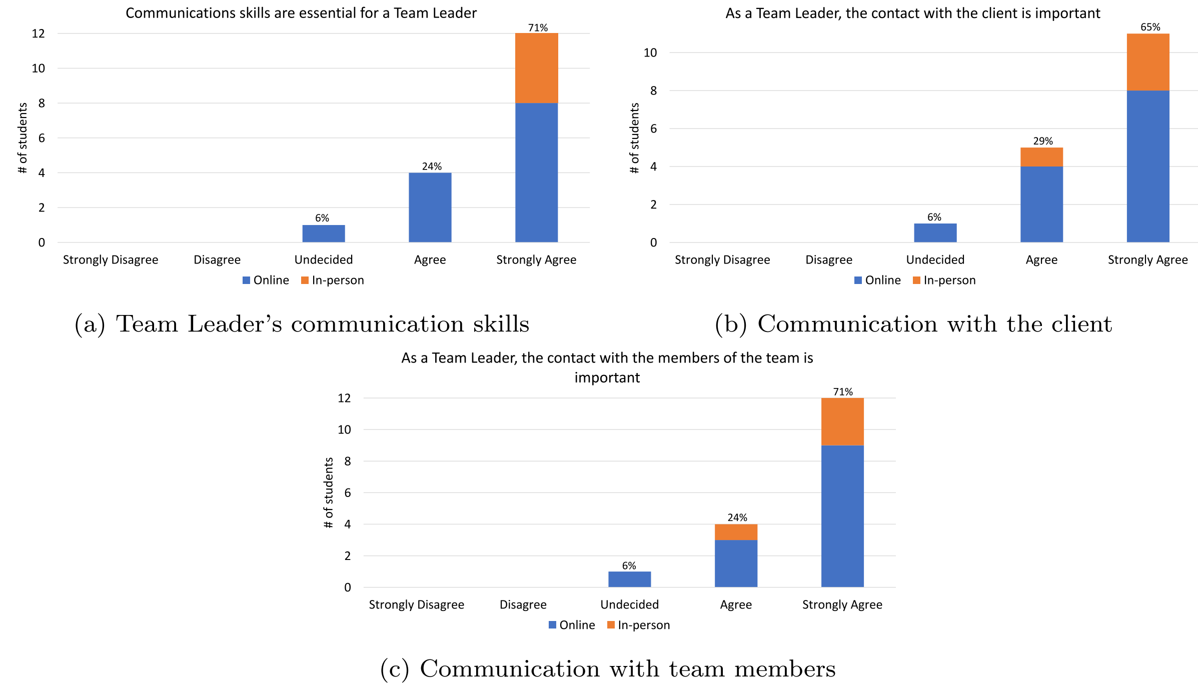
this agreement (88%, 94%, and 82%) regarding the work in a distributed environment. It is interesting to see in Fig. 5(d) that half of the in-person students are undecided about the learning experience of working with other students with different skills. Further research could be done to understand their perception.

As mentioned, using a real-life problem is a significant component of effective student learning throughout the PBL process. Therefore, PBL can support the development of technical skills [51]. Knowing where they stand in mastering these skills is crucial for students, as it gives them more confidence when confronted with problems they must address [52]. Fig. 6 shows how students felt about their confidence in working on a software project *before* this course (Fig. 6(a)) and how they felt *after* it ended (Fig. 6(b)). We can see that only 53% of the students were confident about their skills initially. After the course ended, this percentage increased to 82%. An increment of 29%.

We need to clarify that this course was not about learning how to program but about software engineering. In a PBL approach, students must be responsible for their learning as it is a learner-centered approach. Therefore, learning should be integrated from various disciplines or subjects. The learning of different technical skills can be perceived by the student's confidence levels at the end of the project.

The results show the students' positive satisfaction when working on a real-life project.

Moreover, the project allows them to increase their technical skills, which are needed in real-life jobs concerning software development. This learning is also evidenced in Fig. 7. From Fig. 7(a), we can see that 76% of the students agreed that having real-life experience helped them to understand how a software development company works internally. Similarly, in Fig. 7(b), 89% of the students consider that they have a better understanding of the work done by a software engineer.

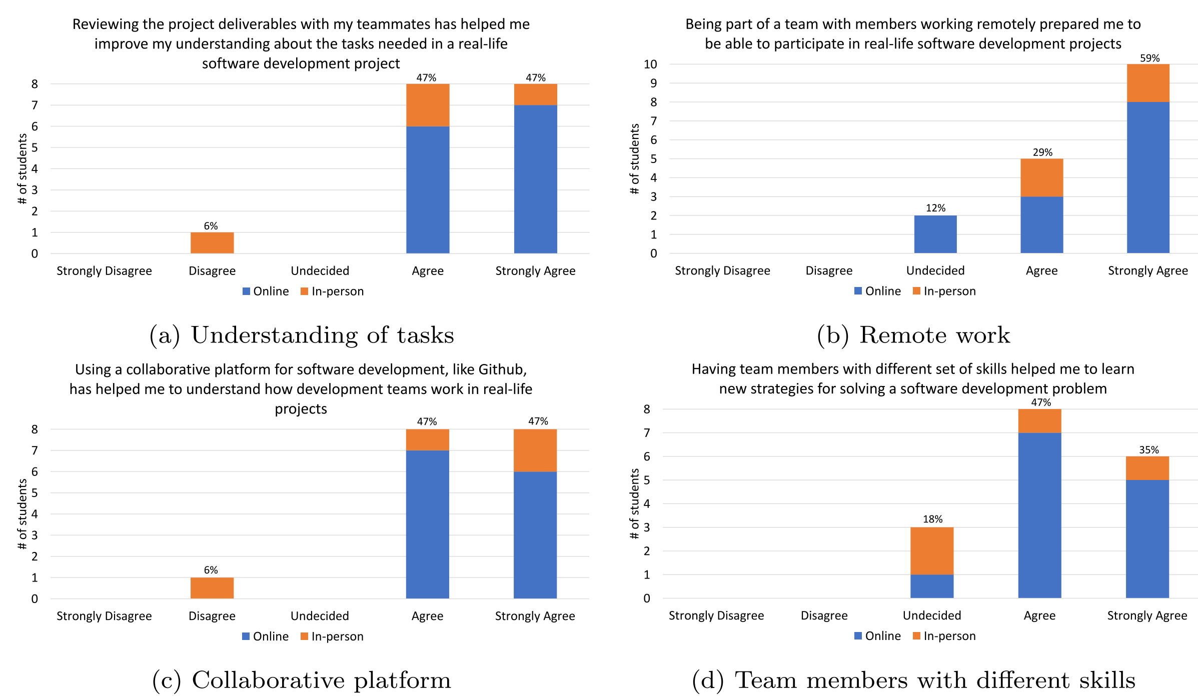


(a) Team Leader's communication skills

(b) Communication with the client

(c) Communication with team members

Fig. 4. Importance of communication skills (i.e., soft skills) for the Team Leader role. Results for questions #5, #3, and #4, respectively.



(a) Understanding of tasks

(b) Remote work

(c) Collaborative platform

(d) Team members with different skills

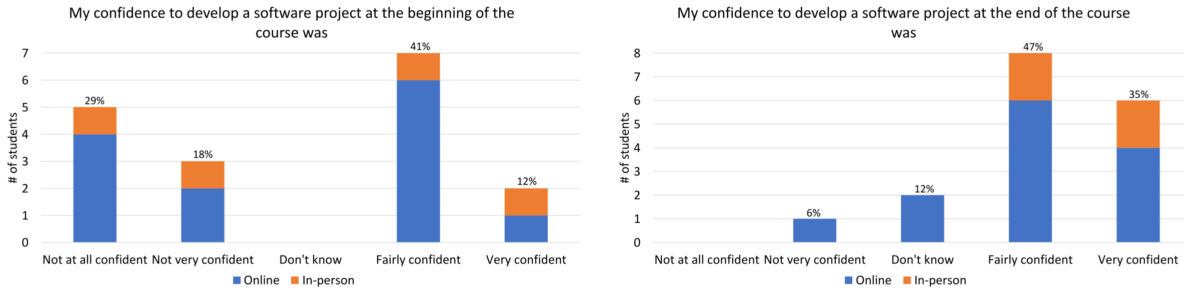
Fig. 5. Importance of PBL to obtain real-life skills. Results for questions #6, #7, #8, and #9, respectively.

5.3. Learning experience

A SE course comprises multiple technical and non-technical concepts such as the software development life cycle (SDLC), requirements engineering, cost estimations, software design, software quality, and maintenance. Therefore, we wanted to assess how a PBL approach influenced students learning of the concepts, strategies, tools, and skills they are supposed to learn on a SE course. Fig. 8(a) shows that 94% of the students consider using a PBL approach helpful to their learning process.

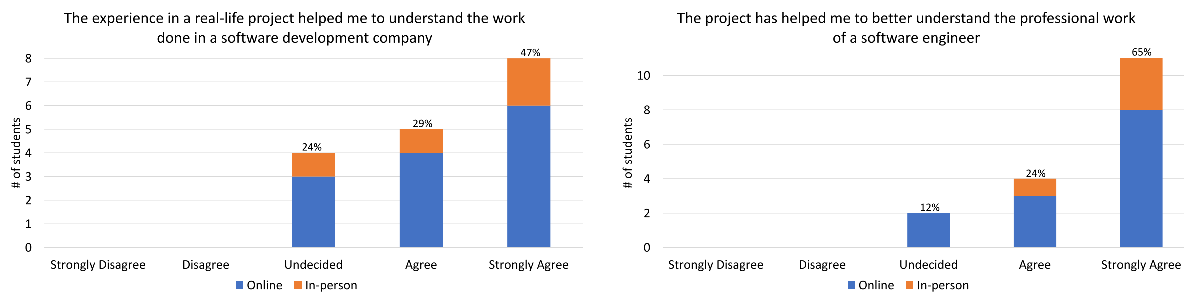
Similarly, regarding the students' understanding of the concepts presented in a SE course (Fig. 8(b)) and specifically, learning about the different stages of the SDLC (Fig. 9(a)), 95% and 94% of the students, respectively, felt that working on a real-life problem as part of our PBL approach helped them to learn. Fig. 9(b) shows that 100% of the students feel that using practical assignments is essential to developing new skills and learning about Software Engineering.

Finally, we wanted to assess how our PBL approach contributed to learning the major topics of a SE course. These topics, aligned with the SDLC, are Software Requirements, Software Design and Analysis,



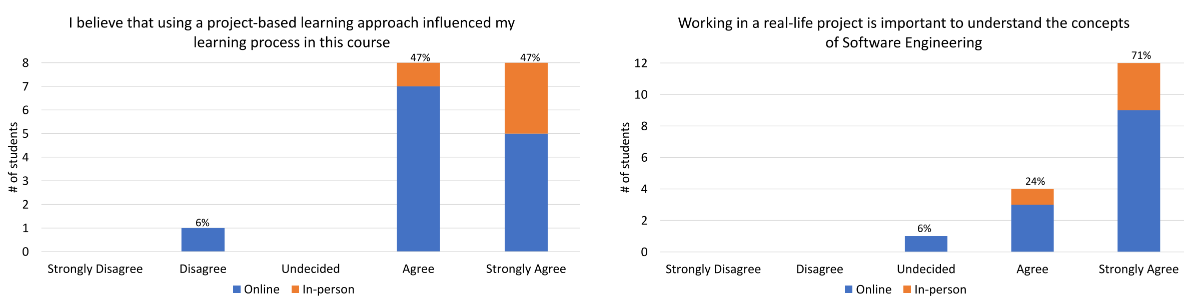
(a) Confidence level *before* the project (b) Confidence level *after* the project

Fig. 6. Confidence level of the students to develop a software project before and after the project. Results for questions #19 and #20, respectively.



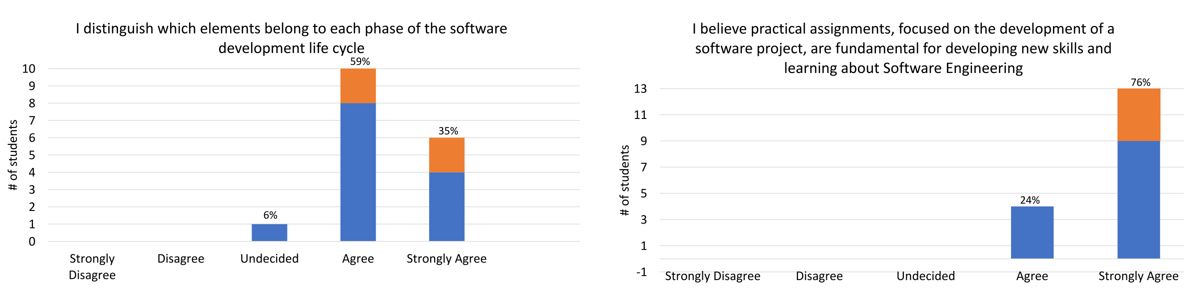
(a) Learning about the work done in a Software development company (b) Learning about work done by a Software Engineer

Fig. 7. Learning about real-life work. Results for questions #11 and #15, respectively.



(a) Influence of PBL in the learning process of SE (b) Real-life project and SE

Fig. 8. Influence of PBL in the learning of SE. Results for questions #10 and #12, respectively.



(a) Influence of PBL in learning the SDLC (b) Learning with practical assignments

Fig. 9. Influence of PBL in the developing of new skills for SE. Results for questions #14 and #18, respectively.

Software Implementation and Construction, Software Testing, and Software Quality. The students were asked to rank how much the project contributed to their learning of the mentioned topics. They had the option to use a ranking from 1 to 5, with 1 being the lowest and 5 the highest. Fig. 10 compares the results for the online and in-person students. From the Figure, we can observe that both groups ranked with a high value (i.e., 4 and above) the project's contribution to their learning of these topics. For the online students, we found that

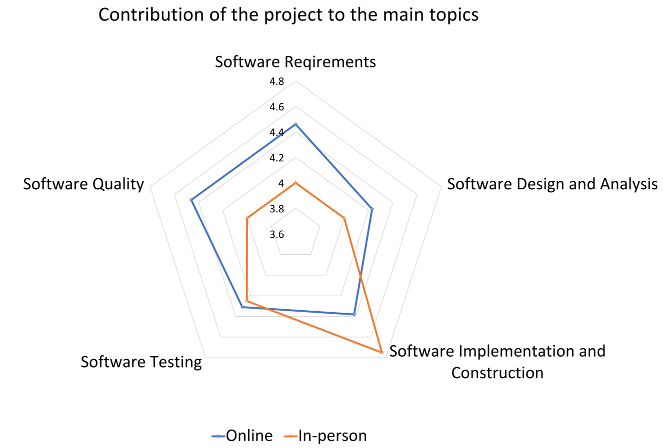


Fig. 10. Contribution of the Project to the learning the topics: "Software requirements", "Software design and analysis", "Software implementation and construction", "Software design", and "Software Quality". Results for question #21.

Software Requirements and *Software Quality* were the topics having the highest average rank, with 4.46. For the in-person students, the topic of *Software Implementation and Construction* had the highest average rank with, 4.75.

These results indicate that our PBL approach helped the students understand the main topics in a SE course. Therefore, we can conclude that the learning experience was positive for the students as they worked on a real-life problem that allowed them to apply the different concepts covered in class.

5.4. Other results from the survey

By using a PBL approach, students enhanced their ability for autonomy, self-confidence, teamwork, and learning to learn, as indicated by other research [53]. In our assessment, we asked questions to evaluate students' opinions toward the real-life project and their overall experience.

One such question is shown in Fig. 11(a). Here, we can see that the overall satisfaction with the final product has a variety of opinions. 70% of the students felt good about their final solution. In the same content, in any software development team, there are members with different sets of skills and different experience levels.

Therefore, the difference in skills inside the team could contribute to dissatisfaction with the final product as those students could feel that they did not contribute enough. Fig. 11(b) shows that 77% of the students felt that the project topic (see Section 4.2) was interesting. Fig. 11(c) shows that 88% of the students think that the feedback received helped them during the course.

5.5. Performance of teams and students

We wanted to see if the performance of the different teams in the project is affected by the type of students forming the team. Fig. 12(a) shows the average grades for each team. This average was computed using the grades from all the stages in the project.

From this Figure, we can see that overall the project was successful for most of the teams. Teams 1, 3, and 7 were formed by all online students (see Fig. 2). Of these teams, only Team 7 obtained a low score. This team had difficulties during the semester, given the availability and time restrictions of one of the team members that withdrew from the course at the end. Overall, the performance of the hybrid teams was good.

Similarly, Fig. 12(b) compares the average grades of online and in-person students for each team. For the average, we consider the project and all the individual assignments and quizzes during the semester. The results show that, the average grade for online students was higher (91.7) than the average grades for in-person students (85.4).

There are several reasons why online college students may perform better than in-person students in the same class [54]. For instance, online students may be more driven to succeed because they opted to continue their education online while balancing other commitments like work and family. Online courses also give online students greater flexibility regarding when and where they do their assignments, which is advantageous for those with busy schedules or those who like to work at their own pace. Online classes could let students set their own learning goals and progress at their own pace, which is advantageous for those who learn best on their own. Similarly, online students must be self-driven and exhibit strong time management abilities to keep up with their studies and fulfill deadlines. These reasons can result in improved grades.

Overall, our experience applying the PBL meets the objectives indicated for this methodology [55]: learning is student-centered, authentic problems are focused on learning, new information is acquired through self-directed learning, learning occurs in a small group, and teachers act as facilitators.

6. Conclusions

This paper presented an application of a PBL approach to a Software Engineering course with online and in-person students in a hybrid class environment. The following conclusions can be drawn from the results obtained after surveying the students from the course.

Our proposed *relaxed plan-based* model, combines the essential features from SE base models. This flexible model helps SE students learn the main characteristics of a software development project and its different stages.

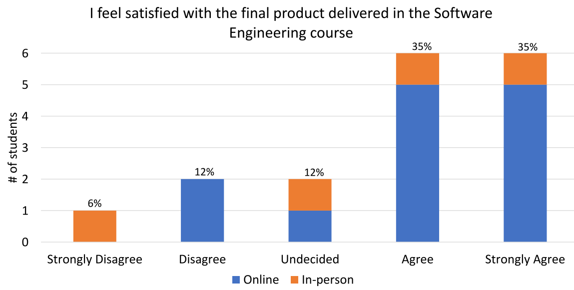
PBL's capacity to bring creativity and energy to teaching Software Engineering (SE) was demonstrated, including a real-life project in the classroom. Unlike classes that solely present information in an expository manner, PBL encourages students to take an active and responsible part in their knowledge acquisition.

PBL applied to our SE course prepared students with technical skills related to software development and, at the same time, helped them to develop soft skills such as communication skills. Applying PBL allowed the students to acquire these abilities and skills while integrating theoretical and practical knowledge related to the different concepts covered in a SE course. The project has contributed to working on skills such as autonomy, group work, trust in themselves, and motivation.

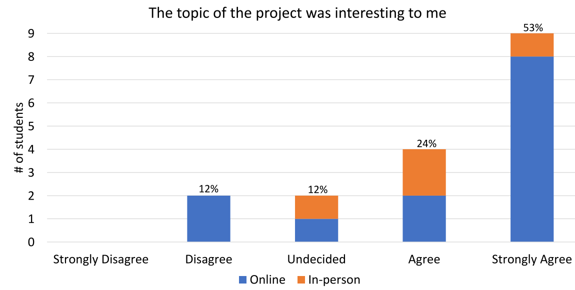
We found some differences in the experience of online and in-person students. Overall, although, on average, their grades were lower than *online* students, *in-person* students seem more involved and satisfied with the experience of using PBL in the course. This difference could be because of the more significant interaction with the instructor in the classroom. Another aspect that could influence the students' experience and satisfaction is the difference in hard (i.e., technical) skills among the team members. The members of a professional software development team have different skills. However, the technical knowledge gap between them is not as ample and evident as among the students. For some students, recognizing that they have technical deficiencies compared to their peers can motivate them to improve and learn more. However, for other students, this situation can be demotivating.

Although our proposed approach still has room for improvement, the implemented PBL strategy has met the standards for teaching Software Engineering. Based on our experience with PBL, we present the following guidelines:

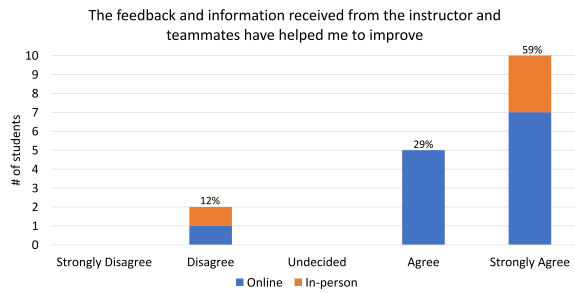
- *The Team Leader must have regular meetings with the customer.* The students need to constantly contact their "client" to validate the requirements and determine the priority of the activities.



(a) Final product satisfaction

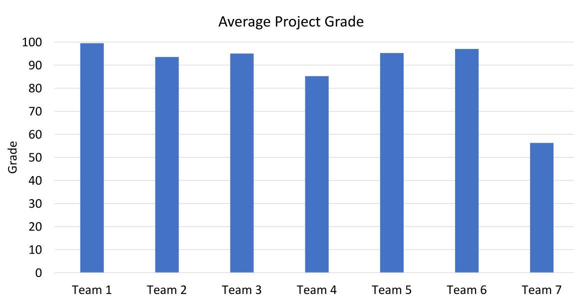


(b) Interest of the project

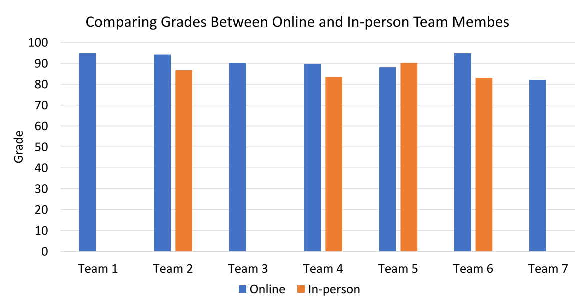


(c) Feedback received

Fig. 11. Other opinions. Results for questions #13, #16, and #17, respectively.



(a) Average grade for the project for each team.



(b) Average grade for type of student

Fig. 12. Average grades (Teams 1, 3 and 7 were formed by online students.).

- *Encourage mixed teams.* Our results indicate that in-person students tend to interact more with the instructor. Therefore, having at least one member from the in-person section could benefit the team because their online teammates can use the extra knowledge they might obtain in the classroom.
- *Use results of the meetings to make adjustments.* Based on the meetings with the Team Leaders, the instructor can determine whether anything about the process needs to change.
- *Allow students to choose the tools and techniques they prefer to solve the problem.* The instructor needs to provide students with an overview of available methods, tools, and techniques. However, the students must be allowed to choose those they prefer to solve the problem. This autonomy will allow them to feel more ownership over their learning and more motivated to go above and beyond their expectations.
- *Implement a peer grading scheme.* Working in a team can be difficult for some students that have never done it. To prevent students from taking advantage of the work of others, implement strategies to allow students to grade their teammates' commitment to the project. The students should understand that there is no such thing as a "free ride" in the project. All members need to collaborate.
- *Invite external stakeholders.* The instructor might be biased based on the constant meetings with all teams. Therefore, additional feedback from outside "stakeholders" (i.e., instructors, clients) is essential for the overall experience. Moreover, students might feel motivated to present their final solution to another person besides their instructor.

We anticipate that our strategy will bring students closer to the real-world software development setting, resulting in students having a greater understanding of the course's topics (see Section 5.3) and, more importantly, feeling more inspired to study SE. Therefore, our study confirms that PBL is a good strategy for teaching SE. Moreover, our proposed PBL approach has demonstrated a positive contribution in supporting teaching SE in a hybrid class environment with online and in-person students. From this experience, we can say that students were generally excited about the PBL method and the value of tackling real-world challenges.

CRediT authorship contribution statement

Edgar Ceh-Varela: Conceptualization, Methodology, Validation, Software, Writing – original draft. **Carlos Canto-Bonilla:** Writing – review & editing, Validation. **Dhimitraq Duni:** Writing – review & editing, Validation.

Declaration of competing interest

One or more of the authors of this paper have disclosed potential or pertinent conflicts of interest, which may include receipt of payment, either direct or indirect, institutional support, or association with an entity in the biomedical field which may be perceived to have potential conflict of interest with this work. For full disclosure statements refer to https://doi.org/10.1016/j.infsof.2023.107189.

Data availability

The data that has been used is confidential.

References

[1] V. Leite, Innovative learning in engineering education: Experimenting with short-term project-oriented research and project-based learning, in: 2017 IEEE 26th International Symposium on Industrial Electronics, ISIE, IEEE, 2017, pp. 1555–1560.

[2] P. Guo, N. Saab, L.S. Post, W. Admiraal, A review of project-based learning in higher education: Student outcomes and measures, Int. J. Educ. Res. 102 (2020) 101586.

[3] H. Chen, J. Yang, Application of IT-integrated project-based learning in the teaching reform of undergraduate education, Int. J. Emerg. Technol. Learn. (IJET) 16 (5) (2021) 248–260.

[4] N. Hosseinzadeh, M.R. Hesamzadeh, Application of project-based learning (PBL) to the teaching of electrical power systems engineering, IEEE Trans. Educ. 55 (4) (2012) 495–501.

[5] Y. Zhou, Y. Hu, L. Dong, Y. Zhao, Q. Hao, et al., Let's do it OR deal with it: Teamwork in project-based learning, in: 2013 IEEE Frontiers in Education Conference, FIE, IEEE, 2013, pp. 1510–1512.

[6] L. Faba, E. Díaz, Combining the project-based learning methodology and computer simulation to enhance the engagement in the context of environmental engineering courses, Comput. Appl. Eng. Educ. 28 (5) (2020) 1311–1326.

[7] J. Des Marchais, M. Bureau, B. Dumais, G. Pigeon, From traditional to problem-based learning: A case report of complete curriculum reform, Med. Educ. 26 (3) (1992) 190–199.

[8] D. Bédard, C. Lison, D. Dalle, D. Côté, N. Boutin, Problem-based and project-based learning in engineering and medicine: Determinants of students' engagement and persistance, Interdiscipl. J. Problem-Based Learn. 6 (2) (2012) 7–30.

[9] N. Najid, R. Ridzwan, T. Tee, A review of problem-based learning in electronic engineering course, Int. J. Acad. Res. Bus. Soc. Sci. 7 (12) (2018) 39–47.

[10] B. Elahi, Implementation of a project-based learning approach, case study of ''measurement and evaluation techniques in industrial engineering'' course, in: 2019 ASEE Annual Conference & Exposition, 2019.

[11] J. Montero, R. San-Segundo, J. Macías-Guarasa, R. de Córdoba, J. Ferreiros, A project-based learning approach to a curricular design on electronics, IEEE Trans. Educ. 49 (3) (2006).

[12] J. Chen, A. Kolmos, X. Du, Forms of implementation and challenges of PBL in engineering education: A review of literature, Eur. J. Eng. Educ. 46 (1) (2021) 90–115.

[13] D.L. Edy, et al., Revisiting the impact of project-based learning on online learning in vocational education: Analysis of learning in pandemic Covid-19, in: 2020 4th International Conference on Vocational Education and Training, ICOVET, IEEE, 2020, pp. 378–381.

[14] R.C. Panicker, D. John, Fully remote project-based learning of hardware/software codesign, in: 2021 IEEE Frontiers in Education Conference, FIE, IEEE, 2021, pp. 1–5.

[15] A. Zlotnik, J.M. Montero, DRAC (distributed remote acess system) an on-line open source project-based learning tool, in: Conference ICL2007, September 26-28, 2007, Kassel University Press, 2007, p. 6.

[16] V. Hayashi, R. Arakaki, F. De Almeida, Project based learning during the covid-19 pandemic: Experiment reports of initiatives in computer engineering, in: 13th International Symposium on Project Approaches in Engineering Education and 18th Active Learning in Engineering Education Workshop, PAEE/ALE 2021, 2021, pp. 247–255.

[17] K. Claypool, M. Claypool, Teaching software engineering through game design, ACM SIGCSE Bull. 37 (3) (2005) 123–127.

[18] S.C. Dos Santos, M.d.C.M. Batista, A.P.C. Cavalcanti, J.O. Albuquerque, S.R. Meira, Applying PBL in software engineering education, in: 2009 22nd Conference on Software Engineering Education and Training, IEEE, 2009, pp. 182–189.

[19] P. Morais, M.J. Ferreira, B. Veloso, Improving student engagement with project-based learning: A case study in software engineering, IEEE Rev. Iberoamericana de Tecnol. Del Aprendizaje 16 (1) (2021) 21–28.

[20] R.M. Capraro, Project-Based Learning: An Integrated Science, Technology, Engineering, and Mathematics (STEM) Approach, Sense publishers, 2008.

[21] W.C. Liu, C.J. Wang, O.S. Tan, C. Koh, J. Ee, A self-determination approach to understanding students' motivation in project work, Learn. Individ. Differ. 19 (1) (2009) 139–145.

[22] T.C. Reeves, J. Herrington, R. Oliver, Authentic activities and online learning, 2002.

[23] W.S. Humphrey, The software engineering process: Definition and scope, in: Proceedings of the 4th International Software Process Workshop on Representing and Enacting the Software Process, 1988, pp. 82–83.

[24] M.L. Fioravanti, B. Sena, L.N. Paschoal, L.R. Silva, A.P. Allian, E.Y. Nakagawa, S.R. Souza, S. Isotani, E.F. Barbosa, Integrating project based learning and project management for software engineering teaching: An experience report, in: Proceedings of the 49th ACM Technical Symposium on Computer Science Education, 2018, pp. 806–811.

[25] I. Richardson, Y. Delaney, Problem based learning in the software engineering classroom, in: 2009 22nd Conference on Software Engineering Education and Training, IEEE, 2009, pp. 174–181.

[26] I. Richardson, L. Reid, S.B. Seidman, B. Pattinson, Y. Delaney, Educating software engineers of the future: Software quality research through problem-based learning, in: 2011 24th IEEE-CS Conference on Software Engineering Education and Training, CSEE&T, IEEE, 2011, pp. 91–100.

[27] N. Piccinini, G. Scollo, Cooperative project-based learning in a web-based software engineering course, J. Educ. Technol. Soc. 9 (4) (2006) 54–62.

[28] M. Souza, R. Moreira, E. Figueiredo, Students perception on the use of project-based learning in software engineering education, in: Proceedings of the XXXIII Brazilian Symposium on Software Engineering, 2019, pp. 537–546.

[29] J. Faizi, M.S. Umar, A conceptual framework for software engineering education: Project based learning approach integrated with industrial collaboration, Int. J. Educ. Manag. Eng.(IJEME) 11 (5) (2021) 46–53.

[30] C. Almeida, C. Franca, Improving the pbl method with experiential learning theory in software engineering teaching, in: 2022 IEEE/ACM 4th International Workshop on Software Engineering Education for the Next Generation, SEENG, IEEE, 2022, pp. 28–35.

[31] A. Saad, The effectiveness of project based learning with computational thinking techniques in a software engineering project course, J. Contemp. Issues Thought 12 (1) (2022) 65–79.

[32] M. Souza, R. Moreira, E. Figueiredo, Playing the project: Incorporating gamification into project-based approaches for software engineering education, in: Anais Do XXVII Workshop Sobre Educação Em Computação, SBC, 2019, pp. 71–80.

[33] M. Ende, R. Lämmermann, P. Brockmann, G. Ayurzana, A virtual, global classroom to teach global software engineering: A mongolian-german team-teaching project, in: 2013 Second International Conference on E-Learning and E-Technologies in Education, ICEEE, IEEE, 2013, pp. 229–233.

[34] J. Olivares-Ceja, B.G. Sanchez, P. Brockmann, A. Kress, J. Staufer, Project-based learning in an international virtual classroom to teach global software engineering, in: EDULEARN17 Proceedings, IATED, 2017, pp. 6263–6273.

[35] D.M. Marutschke, V.V. Kryssanov, P. Brockmann, East meets west: Global software engineering course in Japan and Germany, in: Proceedings of the 13th International Conference on Global Software Engineering, 2018, pp. 30–34.

[36] P. Rai, S. Dhir, Impact of different methodologies in software development process, Int. J. Comput. Sci. Inf. Technol. 5 (2) (2014) 1112–1116.

[37] H.D. Benington, Production of large computer programs, Ann. History Comput. 5 (4) (1983) 350–361.

[38] B.W. Boehm, A spiral model of software development and enhancement, Computer 21 (5) (1988) 61–72.

[39] P. Matković, P. Tumbas, A comparative overview of the evolution of software development models, Int. J. Ind. Eng. Manag. 1 (4) (2010) 163.

[40] R.S. Yadav, Improvement in the V-Model, Int. J. Sci. Eng. Res. 3 (2) (2012) 1–6.

[41] R. Matinnejad, Agile model driven development: An intelligent compromise, in: 2011 Ninth International Conference on Software Engineering Research, Management and Applications, IEEE, 2011, pp. 197–202.

[42] T. Chau, F. Maurer, G. Melnik, Knowledge sharing: Agile methods vs. Tayloristic methods, in: WET ICE 2003. Proceedings. Twelfth IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, 2003, IEEE, 2003, pp. 302–307.

[43] J.J.A. Pimentel, Aprendizaje basado en proyectos utilizando L-systems en un curso de compiladores, Programación Mat. Software (2018).

[44] P. Zainal, D. Razali, Z. Mansor, Team formation for agile software development: A review, Int. J. Adv. Sci. Eng. Inf. Technol. 10 (2) (2020) 555–561.

[45] P. Sánchez, C. Blanco, Implantación de una metodología de aprendizaje basada en proyectos para una asignatura de ingeniería del software, Jornadas de Enseñanza de la Inform. (18es: 2012: Ciudad Real) (2012).

[46] S.R. Porter, M.E. Whitcomb, Non-response in student surveys: The role of demographics, engagement and personality, Res. Higher Educ. 46 (2) (2005) 127–152.

[47] G. Matturro, F. Raschetti, C. Fontán, Soft skills in software development teams: A survey of the points of view of team leaders and team members, in: 2015 IEEE/ACM 8th International Workshop on Cooperative and Human Aspects of Software Engineering, IEEE, 2015, pp. 101–104.

[48] C. Kwiatkowski, Effective team leader and interpersonal communication skills, in: Sustainable Leadership for Entrepreneurs and Academics, Springer, 2019, pp. 121–130.

[49] S.K. Judge, K. Osman, S.F.M. Yassin, Cultivating communication through PBL with ICT, Procedia-Soc. Behav. Sci. 15 (2011) 1546–1550.

[50] R. Jain, U. Suman, A project management framework for global software development, ACM SIGSOFT Software Eng. Not. 43 (1) (2018) 1–10.

[51] A.J. Rohm, M. Stefl, N. Ward, Future proof and real-world ready: The role of live project-based learning in students' skill development, J. Market. Educ. 43 (2) (2021) 204–215.

[52] A. Lanthony, E. Azzouzi, A. François, N. Peyret, Self-assessment in PBL: A tool to develop self-confidence and autonomy of students: The example of self-assessment experiment at ISMEP-supmeca in France, in: 2018 IEEE Global Engineering Education Conference, EDUCON, IEEE, 2018, pp. 263–267.

[53] M. Reynolds, K. Trehan, Assessment: A critical perspective, Stud. Higher Educ. 25 (3) (2000) 267–278.

[54] B. Means, Y. Toyama, R. Murphy, M. Bakia, K. Jones, Evaluation of evidence-based practices in online learning: A meta-analysis and review of online learning studies, 2009.

[55] R. Novalinda, M. Giatman, M. FAJRA, et al., Problem-based learning: 21st century vocational education, Int. J. Multi Sci. 1 (08) (2020) 12–19.