Revista Iberoamericana para la Investigación y el Desarrollo Educativo
ISSN 2007 - 7467

https://doi.org/10.23913/ride.v14i28.1850

Scientific articles

Comparativa del aprendizaje basado en proyectos para el desarrollo de software en grupo

Comparison of project-based learning for group software development

Comparação de aprendizagem baseada em projetos para desenvolvimento de software em grupo

José Mendivil-Torres
Universidad Autónoma de Sinaloa, Facultad de Ingeniería Mochis, México
jose.mendivil@uas.edu.mx
https://orcid.org/0000-0002-7621-8722

Alan Ramírez-Noriega
Universidad Autónoma de Sinaloa, Facultad de Ingeniería Mochis, México
alandramireznoriega@uas.edu.mx
https://orcid.org/0000-0002-8634-9988

Carolina Tripp-Barba
Universidad Autónoma de Sinaloa, Facultad de Informática Mazatlán, México
ctripp@uas.edu.mx
https://orcid.org/0000-0002-4811-0247

José Alfonso Aguilar-Calderón
Universidad Autónoma de Sinaloa, Facultad de Informática Mazatlán, México
ja.aguilar@uas.edu.mx
https://orcid.org/0000-0003-2048-9600

Samantha Jiménez
Universidad Autónoma de Baja California, México
samantha.jimenez@uabc.edu.mx
https://orcid.org/0000-0003-0938-7291

Revista Iberoamericana para la Investigación y el Desarrollo Educativo
ISSN 2007 - 7467

Resumen

El aprendizaje basado en proyectos es una estrategia metodológica que implementa tareas sustentadas en la resolución de problemas a través de un proceso de investigación o creación por parte de los estudiantes, quienes trabajan con relativa autonomía, alto nivel de implicación y cooperación para procurar conseguir un producto. Por ende, esta investigación tiene como objetivo determinar el impacto del aprendizaje basado en proyectos en una carrera de ingeniería de *software*. En concreto, se realizó un experimento donde se aplicaron tres proyectos de desarrollo de *software* a tres grupos diferentes, para lo cual se monitoreó el desempeño de ellos y se analizó el conocimiento técnico, metodológico y organizacional de cada uno para medir el impacto en su desempeño académico. Los resultados muestran diferentes niveles de organización y capacidades entre los grupos para cumplir con el proyecto en los tiempos establecidos.

Palabras claves: aprendizaje basado en proyectos, ingeniería de *software*, desarrollo de *software*, trabajo en equipo, proyecto grupal.

Abstract

Project-based learning is a methodological strategy that implements tasks based on problem solving, through a process of research or creation by students who work relatively autonomously and with a high level of involvement and cooperation that culminates in a product. This research aims to determine the impact of project-based learning on a software engineering career. However, the research has the characteristic that the whole group focuses on developing the same project. An experiment was carried out where three software development projects were applied to three different groups, for which the performance of the groups was monitored. In the end, a survey was applied to determine the impact of the development of projects in a group. In the results, the groups showed different levels of organization and capacities to fulfill the project in the established times. The groups' technical, methodological, and organizational knowledge was analyzed to measure the impact on their academic performance. This research is helpful since it allows measuring the impact of project-based learning in groups and defines some guidelines to carry out said pedagogical strategy in software development.

Keywords: Project-based learning, software engineering, software development, teamwork, group project.

Revista Iberoamericana para la Investigación y el Desarrollo Educativo
ISSN 2007 - 7467

Resumo

A aprendizagem baseada em projetos é uma estratégia metodológica que implementa tarefas baseadas na resolução de problemas através de um processo de pesquisa ou criação por parte dos alunos, que trabalham com relativa autonomia, elevado nível de envolvimento e cooperação para tentar alcançar um produto. Portanto, esta pesquisa tem como objetivo determinar o impacto da aprendizagem baseada em projetos na carreira de engenharia de software. Especificamente, foi realizada uma experiência onde foram aplicados três projetos de desenvolvimento de software a três grupos diferentes, para os quais foi monitorado o seu desempenho e analisado o conhecimento técnico, metodológico e organizacional de cada um para medir o impacto no seu desempenho acadêmico. Os resultados mostram diferentes níveis de organização e capacidades entre os grupos para concluir o projeto dentro dos prazos estabelecidos.

Palavras-chave: aprendizagem baseada em projetos, engenharia de software, desenvolvimento de software, trabalho em equipe, projeto em grupo.

Reception date: August 2023 **Acceptance Date:** March 2024

Introduction

The most important skill in the digital age that students must acquire is learning to learn, hence learning has gone from being an individual construction of knowledge to a social process. However, even for the most experienced teachers, keeping students engaged and motivated constitutes a great challenge (Zambrano Briones *et al*., 2022), although there are various teaching strategies that can be applied taking into account the characteristics of the students. Some of them, according to Chimbo Jumbo and Larreal Bracho (2023), may be the following: 1) didactic approach for individualization, such as programmed teaching, self-directed learning and academic tutoring; 2) didactic socialization approach, with models such as the case method, seminar and peer tutoring, and 3) globalized approach, such as problem solving and project-based learning (PBL), which is the object of analysis in the present investigation.

PBL is a methodology in which students have an active role in order to promote academic motivation. In this model, the main activity for the acquisition of training objectives is based on the development of a project that tries to respond to a real need, which, normally, involves the creation of a final product. To achieve this, students must acquire or practice the skills that the teacher has proposed for instruction. Usually, this technique is linked to a group

work method, so that the project is developed by a team of students who organize themselves to achieve the final objectives (Cyrulies and Schamne, 2021; Marnewick, 2023).

This approach can be approached in an intradisciplinary and interdisciplinary way, which allows promoting collaborative work between teachers and with the educational center (Al Mulhim and Eldokhny , 2020; Sotomayor *et al* ., 2021). Its main characteristic, therefore, is to provide students with a real context and involve them directly in the teaching-learning process. In other words, the student is responsible for making a series of decisions to solve a task of a certain level of complexity (Abella García *et al* ., 2020; Morais *et al* ., 2021; Pérez and Rubio, 2020).

In this sense, it has been shown that students who have participated in PBL show a better ability to solve assignments, in addition to being more self-sufficient and committed to their learning, in a way that contributes to promoting autonomy, self-confidence and increase motivation. This means abandoning mechanical teaching in favor of a methodology where tasks are posed as challenges in a common dynamic, instead of a decontextualized assignment of unrelated work (Abella García *et al* ., 2020).

Although PBL is useful for many areas, it is especially relevant in professions where production or service is based on the generation of projects. For this reason, and although there are many careers where it can be applied, this research focuses on Software Engineering (IS), which is based on the application of a systematic, disciplined and quantifiable approach to the development of computer programs. This includes the creation of tools, methods and techniques to support *software* production, as well as all aspects related to project management, involving people, processes and technological tools.

This career combines technical aspects of computer science with soft skills such as communication, negotiation, collaboration, and teamwork, among others. In this way, the *software engineer* can develop IT solutions that satisfy the needs of stakeholders and increase the efficiency of the affected business processes (Gómez Álvarez *et al* ., 2015).

The importance of soft skills in the professional practice of the *software engineer* has driven the development of strategies aimed at their integration in the teaching-learning process, which seek to create collaborative environments where students can cultivate creativity and fundamental social skills. for your engineering practice, such as effective communication, leadership, negotiation skills and teamwork. These experiences also ensure that students get involved in the reality of organizations to reduce the gap between the university and the company, specifically with regard to the development of the skills required by the industry and those that are developed during the professional training of the student

(Gómez Álvarez *et al* ., 2015; Nurbekova *et al* ., 2020). The integration of these soft skills into the *software* engineering teaching process has been carried out through 1) case studies to simulate real *software development environments* and 2) the execution of university-company *software projects.*

In the educational field, the general objective of the Software Engineering subject is for students to acquire the skills to develop software projects *from* their conception to their delivery to the client. Therefore, students must acquire the knowledge and skills necessary for the complete development of a *software project* (Sánchez and Blanco, 2012). Software Engineering students, when faced with PBL, must 1) face incomplete and inaccurate information, 2) self-regulate and commit to the work, so that they can commit to the learning process by defining their own objectives within the established limits, 3) cooperate and work in groups to divide the workload between them and integrate the different parts developed by each one, and (4) work with multidisciplinary topics, practical skills that are important requirements in the business and industrial field. .

PBL is relevant to *software development* as it is a systematic teaching method in which students engage in the application of essential knowledge and life skills through an extensive and structured student-influenced inquiry process around of complex authentic questions in order to create high-quality products (Villalobos-Abarca *et al* ., 2018). In other words, by using PBL, students become more responsible for developing their knowledge.

Additionally, it is believed to encourage deeper learning compared to traditional lectures, where students retain less of what they are taught. In fact, it is estimated that evaluation in PBL is also influenced by proximity to real-world situations. Since the artifacts constructed during the project are themselves representations of learning, it is important to provide authentic and constructive feedback for the objectives of the problem (Da Cunha, 2005).

PBL has been widely adopted in engineering education, and is defined as a constructivist instructional teaching method that uses real problems as a motivating element for learning. Several studies in computer education have shown that PBL facilitates the commitment and learning capacity of students, since it contributes to the development of skills such as teamwork, holistic vision, critical thinking and problem solving. In fact, some research has shown that PBL is especially suitable for teaching Software Engineering, as it allows students to stimulate deep learning and apply it from the beginning of the course. In addition to this, the fact of seeing a *software solution materialized* that does not deviate from

industrial practice and that uses cutting-edge technologies represents an important additional stimulus (Adorjan and Solari, 2021).

Considering these aspects, this research applied a PBL case study in a Software Engineering degree to determine the impact of the methodology considering complete work groups, that is, the entire group on the same project. For this, three groups were formed with different *software* development projects , starting from scratch. The groups had to generate a *software* application as a deliverable, as well as documentation about said process. A special feature was that the projects were organized by groups, that is, the entire group was focused on developing the same project, so the software development roles *were* distributed between teams.

The research question posed was the following: what are the main competencies that can be highlighted in the practice of project-based group learning for software *development?*

The structure of the article is as follows: section 2 offers the state of the art regarding PBL. The third section highlights the research method used, while section 4 details the experiment carried out and the results obtained. The results and limitations of the study are discussed in section 5, and conclusions and ideas for future work are presented in section 7.

Related work

The learning process requires changes in pedagogical methodologies. In this sense, one of the most popular in recent years is project-based learning (PBL), which represents one of the most successful student-centered pedagogies and one of the most widely used in software development *courses* (Macías, 2012). The main characteristic of this method is that students get involved in a real scenario where they directly participate in the teaching-learning process to solve a high-level task (Abella García *et al* ., 2020). This approach seeks to promote motivation, collaboration with team members, proactivity and soft skills, key tools for success in projects and in teaching-learning environments (Sánchez and Blanco, 2012).

In a study conducted by Sánchez and Blanco (2012), the authors compared the difference between students receiving lectures and those enrolled in practical, immersive projects in real settings. According to the results found, students who learned using the practical approach improved their final grades by approximately 35 %, which suggests that learning can be better stimulated through this method.

Other researchers, such as Macías (2012), focused on evaluating student motivation and perception in this type of learning environment. To do this, they conducted a survey in order to evaluate student satisfaction and used the student evaluations of educational quality (SEEQ, Student Evaluations of Educational Quality). This is a questionnaire widely used in academic evaluation to measure valuable psychometric characteristics such as reliability, validity, internal consistency, etc. Their findings suggest that better student grades have complemented the increase in student satisfaction; In addition, there was an increase of 40 % compared to previous years in the number of students who passed the degree, and an increase of 30 % in the rate of students who obtain better grades (Macías, 2012). The instructors involved in the experiment met at the end and discussed progress in the *software engineering labs*. Regarding this, they reported a clear improvement in homogeneity thanks to the process, although, as a negative point, all instructors noted a greater workload with the new system, particularly when completing rubrics and writing detailed comments to students (Macías, 2012). This shows that one of the main problems of PBL is evaluation, since the process must consider the contributions of each team member to the project.

Even so, Abella García *et al* . (2020) suggest the use of rubrics, grading components, and a percentage assigned to each (Cyrulies and Schamne , 2021), so that all aspects can be considered to improve grading criteria and reduce the number of injustices. However, Da Cunha (2005) lists some negative experiences in implementing this approach. In their experiment, for example, 88 students participated and approximately 95 % passed the subject. Although the experience was generally positive, students faced some problems, such as working in groups and conflicts due to divided opinions. Additionally, they made poor decisions about some tools and it was necessary to change applications a couple of times, as well as migrate the project to a new application.

Even so, it is worth noting that the implementation of PBL in Software Engineering careers has reduced the gap between the industry and the theoretical approach of all universities or educational institutions (Villalobos-Abarca et *al* ., 2018). In fact, although in some cases its use has caused some difficulties for students, experience has also taught that sooner or later students will face these types of problems in real scenarios.

Materials and methods

This research was based on a mixed approach, since both qualitative and quantitative methods were combined. That is, on the one hand, constant evaluations of numerical and statistical data are carried out, and, on the other, an immersion was carried out in the groups to define how the projects evolve. This mixed approach makes it possible to take advantage of the strengths of both methods and compensate for the weaknesses of each separately, thereby achieving an objective and interpretive analysis according to the experience of immersion in the groups.

The research, on the other hand, had a field work component, which means being immersed in the project development process to guide and advise the groups, as well as explain the technical aspects of the process. This takes shape in two scenarios:

1. The study begins in the classroom of the Mochis Faculty of Engineering, where project advice is provided and the technical aspects of the subject are covered.
2. Students work in their homes, the space where the greatest progress of the project takes place because there are fewer distractions and, therefore, concentration levels can be increased.

Kind of investigation

A quasi-experiment was carried out, since the groups were pre-assigned according to the academic organization of the Mochis Faculty of Engineering (Autonomous University of Sinaloa). These groups were formed at the beginning of the degree and may experience changes due to poor academic performance or personal reasons, but their base remains constant from the beginning.

Likewise, the research adopted an exploratory approach with the objective of identifying relationships between variables that can explain the behavior of the groups. Then, a descriptive analysis was carried out to identify important aspects based on tables and graphs and, subsequently, an analysis was carried out based on hypothesis testing, which allowed variables to be contrasted to identify those that affect the performance of the groups.

The study population was made up of students of the Software Engineering degree at the Mochis Faculty of Engineering of the Autonomous University of Sinaloa, specifically those who are in the sixth semester. These students are mostly men between 20 and 22 years old with a technological profile that includes knowledge of using computers, programming languages, databases and some with experience in web development. The experiment

included three groups per semester: two evening groups and one night group, that is, all students currently taking said subject were considered.

Procedure

The research project collected information for analysis as follows:

- Students periodically submit progress reports on the development of the project, which are presented to the group. The teacher then records this progress and keeps track of the status of the project.
- Data is collected through a general survey applied to all members of the group. This evaluation was carried out using online forms.
- was carried out during classes in the classroom with the groups, where concepts were explained and advice was provided to resolve doubts about the projects.

The procedure followed in the experiment was the following:

1. *software* development and their list of requirements were generated. The names of the projects were 1) development of a point of sale, 2) automation of *software estimation* with *planning poker*, and 3) construction of collaborative *software* for Mexican sign language. Each project is different from each other, but they all require a web application and generating *software* engineering documentation.
2. Projects were randomly assigned using a free web-based system that generates random numbers. Group 1 was assigned the project development of a point of sale, group 2 was assigned the project automation of *software estimation* with Planning. Poker , and group 3 the project to build collaborative *software* for Mexican sign language.
3. *software* projects were explained so that the students knew them and chose two: a main one and a secondary one. This was done to distribute the roles among the students and prevent them from only selecting one. Roles included analyst, designer, programmer, *tester*, documenter, and project leader.
4. The subject where the entire project was developed was Web Application Development, which served as a basis to explain topics such as HTML, CSS, JavaScript and the PHP server-side language. All projects had to apply these technologies, although it was not mandatory; If any group wanted to use other technologies, they were free to do so.

5. The professor assigned to the subject served as the client of the project, that is, the person who requested the project. At the beginning, each group had to set dates for interviews with the client and start generating the *software requirements*. In the end, the requirements generated by the group were compared with the original ones established by the teacher, which served as a point of comparison to measure the analysts' understanding of the problem. Subsequently, the original requirements of the project were provided to avoid confusion.
6. The working groups had to develop various Software Engineering artifacts, such as requirements generation documentation, use case diagrams, use cases, class diagrams, entity-relationship diagrams, interface design, among others.
7. As the classes developed, progress on the project was requested, and each group had to explain their progress according to the progress of their roles. The project leaders served as group coordinators, and communication and demands towards the teams were carried out through them.
8. At the end of the semester, each group had to present their finished project, at least up to the advanced level.

These aspects were the same for all students, although the projects were different to avoid copies of code, documentation or ideas between groups; Furthermore, each group had to develop their project independently.

Projects

The projects assigned to the groups are described as follows:

1. Point of Sale Development: Develop an online point of sale system with quote support. This must allow you to register products and manage sales *stock.* Additionally, it must work with a barcode capture gun.
2. *Software* Estimation with Planning Poker: This involves implementing a user story estimation system for Scrum using the Planning Poker method. The system must allow estimating the complexity of user stories through virtual Planning Poker cards. Additionally, it should include project management features, user stories, members, and an estimate history.
3. Construction of collaborative *software* for Mexican Sign Language: The project involves developing a repository of short videos for Mexican Sign Language. This should allow you to upload videos and perform searches, as well as include the functionality of evaluating videos for possible deletion if users so decide.

Instrument

At the end of the project delivery dates, a survey was applied to collect data and perform an analysis that would identify possible relationships between variables. The various blocks that made up the instrument used in said survey are detailed below.

General data

Basic data was collected such as name, surname, age, sex, average to date, group to which they belong, role played in the project and whether they are a regular student or not.

Questions related to the group project

The questions posed in this section were structured in a 5-point Likert format: totally disagree (1), disagree (2), neutral (3), agree (4) and totally agree (5). Below are the questions or statements raised:

1. Have I participated in another similar group project this semester or previously (yes, no)
2. I found the group project very useful for collaboration between group members (5-point Likert).
3. I found the group project very useful to perceive the function of each role in a *software project* (5-point Likert).
4. I would like to develop another similar project in other subjects (5-point Likert).
5. The theme of the project seemed very complex to me (Likert 5 points).
6. The project was very complex for our technical knowledge (use of programming languages, database managers, code and document repositories, etc.) (Likert 5 points).
7. The project was very complex for our methodological knowledge (software development methodologies, role activities, documentation development, etc.) (5-point Likert).
8. The project was very complex for the organizational capacity of the group (organization of work activities, communication between groups, attendance at meetings) (5-point Likert).
9. You consider that the start and end time of the project was (short time, adequate time, long time).
10. Do you consider that another role was needed for the project to develop in a better way (yes [which], no).

11. How would you rate your performance on the group project (very bad, bad, neutral, good, very good).
12. There is some general aspect (that you would like to comment on) that would have caused the group project to develop in a better way.
13. If the project were repeated, would you choose the same role?
14. You chose to change roles, which role would you choose?
15. Why would you change roles?

Transversal or generic skills

These competencies are essential for individuals to be productive upon entering the world of work, and are not specifically related to a particular professional area. The answers to these questions were based on the Likert format, which evaluates the person's ability to fulfill them, as follows: very unable (1), unable (2), neutral (3), capable (4), and very capable (5). The competencies are detailed below along with their definition:

1. Oral and written communication: Transmit knowledge, express ideas and arguments in a clear, rigorous and convincing manner, both orally and in writing, using the necessary graphic resources and media appropriately to adapt to the characteristics of the situation and the audience.
2. Analysis and synthesis of information: Recognize and describe the constituent elements of a reality, and proceed to organize significant information according to pre-established criteria appropriate to a purpose.
3. Problem formulation and resolution: Analyze the constituent elements of a problem to devise strategies that allow for obtaining, in a reasoned manner, a proven and appropriate solution for certain pre-established criteria.
4. Solution Modeling: Analyze the foundations and properties of existing models, and translate and interpret model elements in real-world terms.
5. Autonomous learning: Learning through initiative and self-interest throughout life.
6. Teamwork: Participate effectively in diverse teams and actively collaborate to achieve common objectives.
7. Decision making: Identify patterns that anticipate possible explanations and/or solutions to industrial, technological and operational problems for adequate decision making.
8. Effective use of ICT tools: Ability to update regarding the use of technology in the area that impacts its continuous improvement, including new technologies.

9. Responsibility in action: Understanding of the professional, ethical, legal, security and social aspects, as well as the responsibility inherent in each of them.
10. Vision on the impact of solutions: The ability to analyze the local and global impact of IT solutions on people, organizations and society in general.

Specific competencies

Questions 11 to 22 correspond to the specific competencies of the *software engineer profile* defined by the National Association of Information Technology Education Institutions (ANIEI) (2022) and the National Accreditation Council in Informatics and Computing (CONAIC) (2023), which must be acquired while studying the degree in Software Engineering. The Likert-type responses were 5 points: very incapable (1), capable (2), neutral (3), capable (4) and very capable (5):

1. *software* requirements engineering: Recognize the context, needs and stakeholders in a system using techniques to identify, obtain, analyze, prioritize, document, verify and validate requirements in the context of the life cycles and processes of software *development* .
2. Design *software* : Design the behavior, architecture and interface of *software solutions* based on requirements and using strategies, methods, techniques and modeling languages typical of *software design* .
3. Build *software* : Develop *software for different types of applications, using programming methodologies and paradigms in the context of software* development life cycles and processes , with the required quality attributes.
4. Perform *software testing* : Plan, assign and execute types, techniques, processes and controls within test scenarios in accordance with the required quality attributes.
5. *software* maintenance : Apply maintenance types, processes and techniques in accordance with the required quality attributes.
6. *software* projects : Use methods, strategies, processes, tools and techniques to manage *software projects* .
7. *software* project parameters : Apply metrics for software estimation (size, cost, effort, personnel, time, productivity, quality and documentation) according to systems life cycle models.
8. *software* quality : Use techniques, tools and strategies to plan, ensure and control the quality of a *software product* .

9. Establish security mechanisms: Create or propose methods and strategies to evaluate security and select criteria that avoid vulnerabilities in software *security*
10. Use life cycles: Use the elements and criteria in the use of life cycle models in accordance with the context of software development *processes* .
11. Verify quality of *software solutions* : Employ various testing models to ensure the quality of the *software product* .
12. Use tools for *software creation* : Use industrial methods and CASE tools for the different phases in the *software process* .

For data treatment and analysis, the SPSS statistical tool was used. Likewise, a within-subjects analysis was carried out, which means that the same experiment was applied to all students and internal aspects of the sample were examined to explore possible relationships between variables. To identify these relationships, the data analysis included various statistical tests, such as the Chi-square test, Fisher's exact test, Student 's t test, and the Kruskal-Wallis test (Flores-Ruiz *et al* ., 2017; Hernández -Sampieri and Mendoza, 2019).

Results

The experiment focused on developing *software projects* in three groups of the Software Engineering degree in the sixth semester: two groups from the afternoon shift and one from the night shift. The characteristics of the groups are detailed in Table 1. The *% symbol* represents the percentage of the number on the left with respect to the total number of students. The word *SD* in parentheses represents the standard deviation.

Table 1. General data of the groups participating in the experiment.

Groups	Students (%)	Men (%)	Women (%)	Ratings (SD)	Age (SD)	Irregular (%)
1	23 (46.9)	19 (82.6)	4 (17.4)	8.8 (0.75)	20.91 (0.95)	9 (39.1)
2	19 (38.8)	14 (73.7)	5 (26.3)	8.2 (0.78)	21.21 (1.36)	9 (47.4)
3	7 (14.3)	7 (100)	0 (0)	8.1 (0.92)	25.43 (5.91)	0 (0)
Totals	49 (100)	40	9	8.4	22.51	18

Source: self made

As can be seen in Table 1, 49 students participated, mainly men. The grade point average mentioned in the table corresponds to the level of academic performance reported by students at that time in their degree, and may vary slightly. The ages of the students are

around 22.5 years, with a notable increase in the ages of the night shift students, who are usually workers and some have family responsibilities.

Table 2. Distribution of students by role and group at the beginning (I) and end (F) of the semester

	Cluster						Total	
	1		2		3			
Role	I	F	I	F	I	F	I	F
Analyst	5	5	3	3	3	2	11	10
Designer	6	6	3	3	2	0	11	9
Programmer	5	4	7	6	4	0	16	10
Tester	4	4	3	2	2	2	9	8
Documenter	3	2	4	3	3	1	10	6
Project leader	2	2	2	2	2	2	6	6
Total	25	23	22	19	16	7	63	49

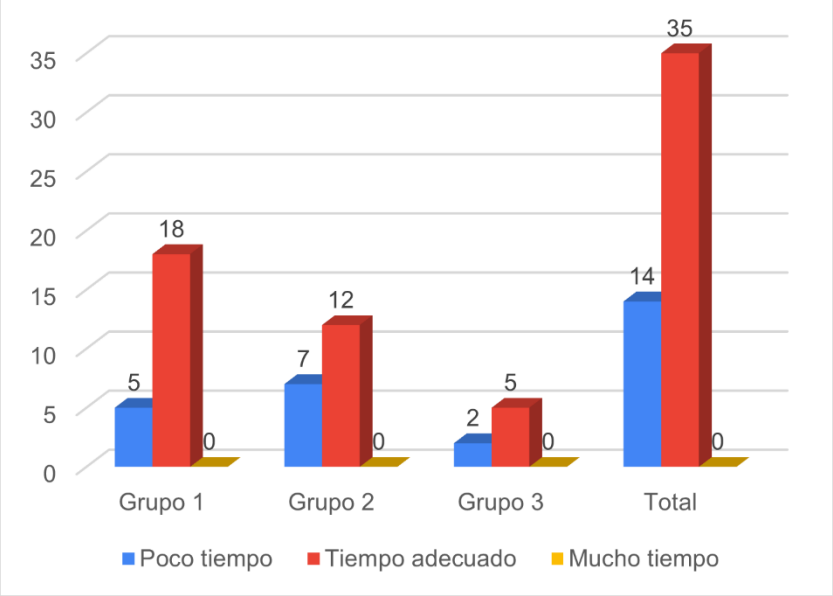
Source: self made

Table 2 shows the roles that students selected at the beginning of the semester. Although a total of 63 students initially chose a role, at the end of the semester only 49 students completed the survey, suggesting a high dropout rate in the subject, especially in group 3. According to table 3, group 3 They recorded desertions in the roles of designer and programmer, which meant that the other members of the group had to assume these additional responsibilities.

Likewise, a survey was administered to the students about their experience in the development of the projects. When asked about working in groups, 94 % responded that they had no previous experience. Regarding the time allocated for the development of the projects (six months, from January to June), the majority considered that it was adequate, although a minority expressed that it was insufficient. Figure 1 summarizes these responses by work groups.

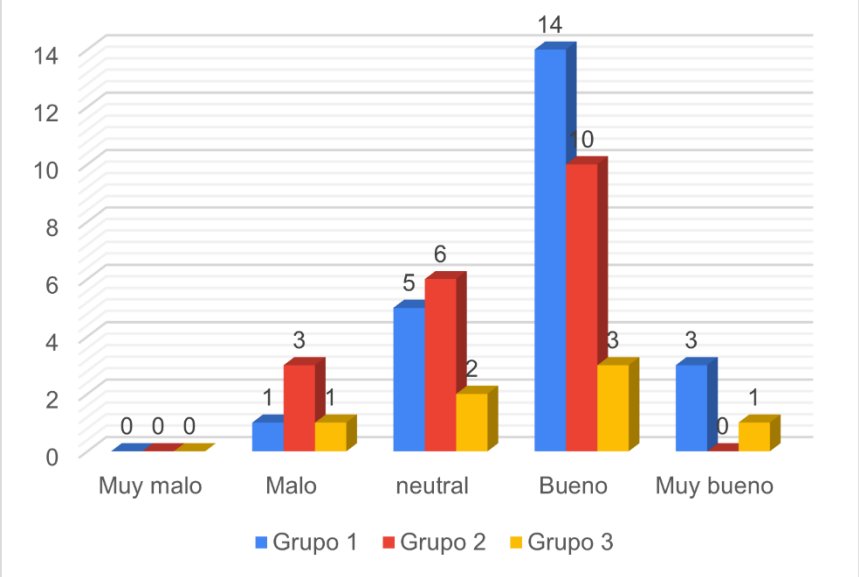
The next question focused on the perception of the students' performance in the development of the project. Only a small number indicated negative performance (10.2 %). On the contrary, the majority perceived their performance as good or very good (63.3 %) (figure 2). Finally, they were asked about their agreement with the role they played in the project. The majority stated that they had chosen the role with which they felt most comfortable (82 %). However, on the 18 % indicated that they would change their role if they had the opportunity to participate in another similar project.

Figure 1. Time dedicated to project development



Source: self made

Figure 2. Performance of participation in project development



Source: self made

Table 3 shows the deliverables of the groups at the end of the semester, where the items that were evaluated on a measure from 0 to 100 are considered. %. It is notable that group 1 was the one with the highest percentage of delivery.

Table 3. Final status of the projects by group

	Activities	Group 1	Group 2	Group 3
	Project	Point of sale	*Planning poker*	Video repository
Analysis	Client interview	100 %	100 %	100 %
	Preparation of list of requirements	100 %	60 %	80 %
Design	Development of use cases	100 %	100 %	100 %
	Prototype design	Figma	-	Adobe XD
	Document repository	Google Drive	Git Hub	One Drive
	Entity-relationship model design	100 %	0 %	100 %
	Interface design	0 %	40 %	100 %
	Aesthetics design	0 %	40 %	100 %
	Content design	0 %	40 %	100 %
	Architectural design	100 %	80 %	100 %
Implementation	Use case programming	90 %	*50* %	70 %
Evidence	System tests	100 %	*20* %	0 %
	Entity-Relationship Diagram Review	100 %	0 %	0 %
	Database Design Evaluation	100 %	0 %	0 %
	Review of use cases	100 %	0 %	0 %

Source: self made

According to table 4, the 51% of students consider the group project useful to collaborate with team members. On the other hand, 30.6 % did not express a clear opinion for or against the idea. Finally, the remaining percentage did not consider the group activity useful to establish collaboration.

On the other hand, as seen in table 5, the 75.5 % of the students who participated in the research consider this type of methodologies applied to the development of projects useful to perceive the function of each role. In contrast, the 6.1 % do not consider it useful, while the rest maintained a neutral position.

In general, there was a lot of division regarding the desire to develop another similar project. The 46.8 % of students agree with this idea; However, the 38.7 % would not be interested in repeating the way of working on the project. The rest of the students have a neutral opinion on the matter (table 6).

Table 4. I found the group project very useful for collaboration between group members.

	Cluster			Total
	1	2	3	
Totally disagree	0 (0.0 %)	3 (15.8 %)	0 (0.0 %)	3(6.1 %)
Disagree	1 (4.3 %)	4 (21.1 %)	1 (14.3 %)	6 (12.2 %)
Neutral	5 (21.7 %)	8 (42.1 %)	2 (28.6 %)	15 (30.6 %)
Agree	7 (30.4 %)	3 (15.8 %)	2 (28.6 %)	12 (24.5 %)
Totally agree	10 (43.5 %)	1 (5.3 %)	2 (28.6 %)	13 (26.5 %)
Total	23 (100.0 %)	19 (100.0 %)	7 (100.0 %)	49 (100.0 %)

Source: self made

Table 5. I found the group project very useful to perceive the function of each role in a *software project*

	Cluster			Total
	1	2	3	
Totally disagree	0 (0.0 %)	1 (5.3 %)	1 (14.3 %)	2 (4.1 %)
Disagree	0 (0.0 %)	1 (5.3 %)	0 (0.0 %)	1 (2.0 %)
Neutral	3 (13.0 %)	5 (26.3 %)	1 (14.3 %)	9 (18.4 %)
Agree	5 (21.7 %)	8 (42.1 %)	2 (28.6 %)	15 (30.6 %)
Totally agree	15 (65.2 %)	4 (21.1 %)	3 (42.9 %)	22 (44.9 %)
Total	23 (100.0 %)	19 (100.0 %)	7 (100.0 %)	49 (100.0 %)

Source: self made

Table 6. I would like to develop another similar project in other subjects

	Cluster			Total
	1	2	3	
Totally disagree	2 (8.7 %)	8 (42.1 %)	3 (42.9 %)	13 (26.5 %)
Disagree	2 (8.7 %)	3 (15.8 %)	1 (14.3 %)	6 (12.2 %)
Neutral	4 (17.4 %)	4 (21.1 %)	2 (28.6 %)	10 (20.4 %)
Agree	4 (17.4 %)	3 (15.8 %)	0 (0.0 %)	7 (14.3 %)
Totally agree	11 (47.8 %)	1 (5.3 %)	1 (14.3 %)	13 (26.5 %)
Total	23 (100.0 %)	19 (100.0 %)	7 (100.0 %)	49 (100.0 %)

Source: self made

Although the majority of students (42.9 %) consider that the assigned project is complex, group 2 is the one that provides the highest percentage of students who consider their project complex in terms of the topic (table 7). In addition, the students were questioned about the complexity of the project considering the technical and methodological knowledge and organizational capacity of the group.

In general, they consider that, in terms of their technical knowledge, the project is not complex. However, a high percentage of group 2 believes that technical complexity does exist (table 8). That is, they do not have the necessary knowledge about the use of programming languages, database managers, code repositories and documentation generation, among others.

Table 7. The theme of the project seemed very complex to me

	Cluster			Total
	1	2	3	
Totally disagree	1 (4.3 %)	0 (0.0 %)	0 (0.0 %)	1 (2.0 %)
Disagree	9 (39.1 %)	0 (0.0 %)	2 (28.6 %)	11 (22.4 %)
Neutral	9 (39.1 %)	5 (26.3 %)	2 (28.6 %)	16 (32.7 %)
Agree	4 (17.4 %)	8 (42.1 %)	0 (0.0 %)	12 (24.5 %)
Totally agree	0 (0.0 %)	6 (31.6 %)	3 (42.9 %)	9 (18.4 %)
Total	23 (100.0 %)	19 (100.0 %)	7 (100.0 %)	49 (100.0 %)

Source: self made

Table 8. The project was too complex for our technical knowledge

	Cluster			Total
	1	2	3	
Totally disagree	2 (8.7 %)	1 (5.3 %)	1 (14.3 %)	4 (8.2 %)
Disagree	8 (34.8 %)	6 (31.6 %)	0 (0.0 %)	14 (28.6 %)
Neutral	12 (52.2 %)	5 (26.3 %)	3 (42.9 %)	20 (40.8 %)
Agree	1 (4.3 %)	4 (21.1 %)	0 (0.0 %)	5 (10.2 %)
Totally agree	0 (0.0 %)	3 (15.8 %)	3 (42.9 %)	6 (12.2 %)
Total	23 (100.0 %)	19 (100.0 %)	7 (100.0 %)	49 (100.0 %)

Source: self made

Regarding methodological knowledge, groups 2 and 3 indicate that they mostly lack knowledge about development methodologies, role activities and documentation development (table 9). Likewise, the three groups consider that the project was complex in terms of the organizational capacity they had, and a high percentage considers that there were many problems in organizing on issues such as the division of work activities, communication between groups and attendance at meetings. (table 10).

The average grade assigned to each group is related to the students' performance on the project. Group 1 obtained the highest average with 8.9 on a scale of 0 to 10. Group 3 is the second in average (with 6.0) and, finally, group 2 obtained the lowest average with 6.2.

Table 9. The project was very complex for our methodological knowledge

	Cluster			Total
	1	2	3	
Totally disagree	1 (4.3 %)	1 (5.3 %)	1 (14.3 %)	3 (6.1 %)
Disagree	8 (34.8 %)	3 (15.8 %)	0 (0.0 %)	11 (22.4 %)
Neutral	12 (52.2 %)	3 (15.8 %)	2 (28.6 %)	17 (34.7 %)
Agree	1 (4.3 %)	6 (31.6 %)	1 (14.3 %)	8 (16.3 %)
Totally agree	1 (4.3 %)	6 (31.6 %)	3 (42.9 %)	10 (20.4 %)
Total	23 (100.0 %)	19 (100.0 %)	7 (100.0 %)	49 (100.0 %)

Source: self made

Table 10. The project was very complex for the organizational capacity of the group

	Cluster			Total
	1	2	3	
Totally disagree	2 (8.7 %)	0 (0.0 %)	0 (0.0 %)	2 (4.1 %)
Disagree	6 (26.1 %)	1 (5.3 %)	1 (14.3 %)	8 (16.3 %)
Neutral	6 (26.1 %)	6 (31.6 %)	2 (28.6 %)	14 (28.6 %)
Agree	8 (34.8 %)	1 (5.3 %)	0 (0.0 %)	9 (18.4 %)
Totally agree	1 (4.3 %)	11 (57.9 %)	4 (57.1 %)	16 (32.7 %)
Total	23 (100.0 %)	19 (100.0 %)	7 (100.0 %)	49 (100.0 %)

Source: self made

The following list shows the competencies established by ANIEI (2022) for a *software engineer* . The numbering of these competencies including the average obtained by each group is represented in table 11:

1. Oral and written communication.
2. Analysis and synthesis of information.
3. Problem formulation and resolution.
4. Solution modeling.
5. Autonomous Learning.
6. Teamwork.
7. Decision making.
8. Effective use of ICT tools.
9. Responsibility in action.
10. Vision on the impact of the solutions.

software engineer are listed below. Table 12 also includes the average obtained by each group.

1. *software* requirements engineering .
2. Design *software* .
3. Build *software* .
4. *software* testing .
5. *software* maintenance .
6. Manage *software projects* .
7. Estimate parameters of the *software project* .
8. Ensures the quality of the *software* .

9. Establish security mechanisms.
10. Use life cycles.
11. Verify quality of *software solutions* .
12. *software* creation .

Table 11. Average transversal or generic skills by group

	Competencies										
Cluster	1	2	3	4	5	6	7	8	9	10	Avg .
1	3.83	3.83	3.96	3.74	3.52	3.96	3.39	3.78	4.04	3.78	3.78
2	3.89	3.37	3.58	3.68	3.63	3.53	3.89	4.05	3.74	3.58	3.69
3	4.00	4.00	4.14	3.86	4.43	4.43	3.86	4.14	4.57	4.29	4.17

Source: self made

Table 12. Average of specific Software Engineering competencies by group

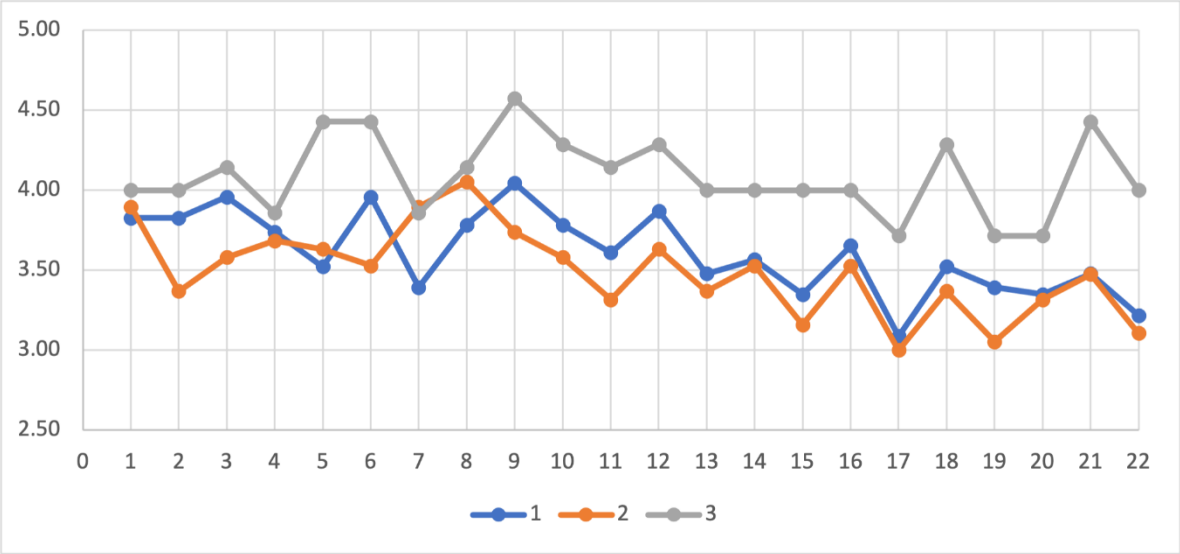
		Competencies										
Cluster	11	12	13	14	15	16	17	18	19	20	21	22	Avg .
1	3.61	3.87	3.48	3.57	3.35	3.65	3.09	3.52	3.39	3.35	3.48	3.22	3.47
2	3.32	3.63	3.37	3.53	3.16	3.53	3.00	3.37	3.05	3.32	3.47	3.11	3.32
3	4.14	4.29	4.00	4.00	4.00	4.00	3.71	4.29	3.71	3.71	4.43	4.00	4.02

Source: self made

Table 11 shows the transversal or generic competencies of the groups that participated in this research. The data in the table reflects that group 2 is mostly the one with the lowest averages, while group 3 has the highest.

On the other hand, table 12 shows the average per group of the specific competencies for a *software engineer* . Again, group 2 has the lowest averages. In fact, in all the competencies individually the lowest values of the three groups are found. It is possible to see how generic skills have higher averages than specific skills. Furthermore, in general, lower averages are seen for group 2.

Figure 3. Skill averages



Source: self made

Statistical analysis

The analysis of the results led to the identification of relationships between variables, which allowed the following hypotheses to be formulated.

Hypothesis 1: There is a significant difference in the performance obtained by the groups during the semester.

The *performance variable* is based on the students' final grades at the end of the semester. In this sense, the groups were previously defined as 1, 2 and 3. To evaluate the disparities in the means between the groups, the Kruskal-Wallis test was used because the data did not present a normal distribution. According to the results of this statistical test, the alternative hypothesis is accepted by obtaining a significance value of 0.000. With a significance level of 0.000 < 0.05, the existence of a significant difference in the grades obtained by groups 1, 2 and 3 during the semester is confirmed.

Table 13 provides a representation of the results generated by the SPSS statistical *software* . In this table, it is observed that group 1 exhibits a significant difference compared to groups 2 and 3, but not between the latter two (table 13).

Table 13. Comparison between groups (difference in grades)

	Qualification obtained in the project
Kruskal-Wallis H	25.670
Df	2
Next.	0.000

Source: self made

To evaluate the effect size, the *eta squared test* , recommended by Tomczak and Tomczak (2014), was applied. The results obtained show an *eta coefficient* of 0.725 and *squared* of 0.527.

Hypothesis 2: The complexity of the project topic impacts student performance.

The analyzed variables were transformed to reduce the categories and avoid an extensive contingency table. The *complexity of the topic variable* is based on the statement "I found the topic of the project very complex," while the *performance variable* refers to the student's final grade on the project at the end of the semester.

Initially, the *topic complexity variable* was on a 5-point Likert scale, but it was transformed to a 3-point Likert scale. Three categories were created: disagreement (including "strongly disagree" and "disagree"), neutral, and agreement (including "strongly agree" and "agree"). The *performance variable* was also reduced to three categories: undesirable (ratings of 5 and 6; category 1), fair (ratings of 7 and 8; category 2), and desirable (ratings of 9 and 10; category 3).

The results of the Fisher exact test in Table 14 indicate a bilateral significance of 0.004, which leads to the acceptance of the hypothesis (0.004 < 0.05) that the complexity of the project topic affects student performance.

Table 14. Complexity of the topic vs. school performance

		Performance			Total
		Undesirable	Regular	Desirable	
Complexity of the theme	Disagree	1	1	10	12
	Neutral	5	2	9	16
	Agree	13	4	4	21
Total		19	7	23	49

Source: self made

Hypothesis 3: The level of technical knowledge to develop the project affects student performance.

The *technical knowledge variable* is obtained from the Likert response to the statement "The project was too complex for our technical knowledge." This variable evaluates whether the student has the technical knowledge necessary to carry out the project. Like the previous variables, it was divided into 3 categories: disagree, neutral and agree.

According to the results in Table 15, Fisher's exact test shows a two-sided significance of 0.001. Since this value meets the condition 0.001 < 0.05, it is concluded that there is a significant relationship between the technical knowledge necessary to develop the project and the student's performance. Therefore, the hypothesis that states that "the technical knowledge to develop the project influences the student's performance" is accepted.

Table 15. Technical knowledge vs. school performance

		Performance			Total
		Undesirable	Regular	Desirable	
Technical knowledge	Disagree	7	1	10	18
	Neutral	4	3	13	20
	Agree	8	3	0	11
Total		19	7	23	49

Source: self made

Hypothesis 4: The level of methodological knowledge to develop the project affects the student's performance.

The *methodological knowledge variable* is based on the Likert response to the statement "The project was too complex for our methodological knowledge." This variable evaluates whether the student has the methodological knowledge necessary to carry out the project. As in the previous hypotheses, it was divided into 3 categories: disagree, neutral and agree.

According to the results in Table 16, Fisher's exact test showed a two-sided significance of 0.001. Since this value meets the condition 0.001 < 0.05, it is concluded that there is a significant relationship between the methodological knowledge necessary to develop the project and the student's performance. Therefore, the hypothesis that states that "the methodological knowledge to develop the project influences the student's performance" is accepted.

Table 16. Methodological knowledge versus school performance

		Performance			Total
		Undesirable	Regular	Desirable	
Methodological knowledge	Disagree	5	1	8	14
	Neutral	3	1	13	17
	Agree	11	5	2	18
Total		19	7	23	49

Source: self made

Hypothesis 5: The organizational capacity of the group to develop projects impacts the student's academic performance.

The *organizational capacity variable* is defined from the response on the Likert scale to the statement "The project turned out to be very complex in terms of organization for the group." This variable was also categorized into 3 levels, as shown in table 17.

However, the results of Fisher's exact test reveal a two-sided significance of 0.153. Given that this value does not meet the established significance criterion (0.153 > 0.05), we cannot accept the hypothesis that suggests that "the organizational capacity of the group to develop projects influences the student's academic performance."

Table 17. Organizational capabilities versus school performance

		Performance			Total
		Undesirable	Regular	Desirable	
Ability organizational	Disagree	1	3	6	10
	Neutral	5	2	7	14
	Agree	13	2	10	25
Total		19	7	23	49

Source: self made

Discussion

The students' experience in group projects was innovative, since it introduced different forms of organization and work, which usually motivates students and, therefore, translates into improvements in their academic performance. Even so, it should also be noted that the work strategy can influence the students in various ways, since - as observed in the final state of the projects (table 3) - group 1 had the best performance, while group 2 presented the worst performance. This last group did not deliver many of the activities and others were completed incompletely, which resulted in very low grades. Furthermore, the activities listed

in Table 3 have different levels of complexity, so calculating an average of the scores would not be fair. One of the main and most significant activities was coding (implementation), where the progress of the project is reflected more clearly, allowing us to appreciate the extent to which each group managed to achieve its objectives.

Analysis of the groups

Group 1 demonstrated the best performance compared to the others. The majority of its members considered that the times assigned to the project were adequate, which was reflected in their performance very consistent with the final grades. In fact, the 73.9 % of students evaluated their performance between good and very good. Although six students expressed their willingness to change roles if necessary, which represents a low percentage (26.1 %), this suggests that overall the group performed well, even though some students were not completely satisfied with their assigned roles.

Likewise, the group showed positive aspects in terms of their knowledge, since they did not consider the project complex in technical and methodological terms, nor did they find the topic complicated. Although they evaluated their organizational skills as average, they demonstrated good attitudes in general during the development of the project.

Considering the average of their grades (8.9), the students in group 1 expressed a positive disposition towards the idea of developing another project in the same way, since they did not consider this way of working as complex. Although the group had the best performance, their skills were not particularly highlighted; In fact, in four of the generic competencies they obtained the lowest values of the three groups. Regarding specific skills, they did not achieve the lowest values, but neither did they achieve the highest.

On the other hand, group 2 recorded the lowest performance in the projects. The 63.2 % indicated that the time allocated was adequate, which, although not an overwhelming majority, suggests a trend toward compliance with established deadlines. Only three cases (15.8 %) expressed poor performance, while 52.5 % considered that their performance on the project was good. Furthermore, only three students (15.8 %) expressed their willingness to change roles if they carried out another project, which implies that the majority were satisfied with the role they played.

The group acknowledged not having sufficient knowledge to tackle the project. Although there was a balanced distribution between negative and positive opinions about the necessary technical knowledge, there was no majority who considered they had the necessary

knowledge. Furthermore, a high percentage indicated they lacked technical knowledge. The organizational capacity of the group for the project was evaluated as very low for the most part, which was reflected in the perception that the project's subject matter was complex. These aspects negatively impacted their performance, since the low ability to develop use cases affected their final grade in the subject, with an average performance of 6.2 among students.

This experience was not satisfactory for them, as they expressed their displeasure and indicated that they would not like to carry out another project of this type in the future. Regarding skills, group 2 showed a lower level than the others. In the generic competencies, they obtained the lowest average in three of the ten competencies, and in the specific competencies, they had the lowest averages in each of the eleven competencies. Although students tend to evaluate themselves more generously, on average they were evaluated with lower values than the rest.

Finally, group 3 demonstrated average or poor performance on the project. Although the group considered that the time allocated for the development of the project was adequate and that their performance on the project was good, the results obtained do not support this perception. Furthermore, all students confirmed that the role they chose was the best fit for them. In the group's opinion, the project allowed collaboration between team members. However, they considered that they lacked the necessary technical, methodological knowledge and organizational capacity, which led them to be dissatisfied with the idea of repeating a similar project. Although they achieved regular performance in the implementation of the use cases with the 70 % were unable to take tests, which determined the group's average score to be 6.9.

In short, the results of the competitions are contradictory, since they contradict the statements about the theoretical, methodological and organizational knowledge of the group. Despite this, the averages indicate that group 3 obtained the highest scores in the generic and specific competencies compared to the other two groups. However, these results do not agree with the results of the project or with the knowledge they claim to possess.

Competencies

The generic competencies with the lowest values for the group with the worst performance are analysis and synthesis of information, and teamwork. These results coincide with the group's performance in obtaining project requirements, which were incomplete and, in some cases, inconsistent. Additionally, the group showed a notable lack of coordination, communication, and cooperation among members, demonstrating a deficiency in teamwork. These findings agree with what was mentioned by Don and Raman (2019), who highlight the importance of teamwork to achieve goals and increase productivity, in addition to fostering camaraderie. However, these data contrast with those mentioned by Souza *et al* . (2019), who applied PBL in a *software* engineering course and found that this strategy benefited the learning of *software* requirements .

Regarding the specific competencies of the *software engineer* , the group with the lowest performance showed a low score in estimating *software project parameters* and establishing security mechanisms for the *software* . The first competence focuses on estimates such as costs, time, personnel and documentation, among others, aspects that were rarely used in the project, given that the deadlines were set and it was not necessary to calculate a cost for the project; Furthermore, the documentation was already defined. Even so, the group indicated that it needs to improve in this competition. These results are similar to those found by Souza *et al* . (2019), who determined that the *software configuration* was the aspect that generated lower student achievement when applying PBL.

Regarding the second competence — *software security* , which refers to the development of aspects that avoid *software vulnerabilities*— , although it was not specifically requested in the project, it was required to comply with certain minimum security standards in user access to the system. Despite this, group 2 indicated having a low level of competencies for more advanced security aspects.

Hypothesis

The scores for groups 1, 2, and 3 were 8.9, 6.2, and 6.9, respectively. According to the Kruskal-Wallis hypothesis test (see *Statistical Analyzes section*), there is a significant difference in grade averages between the groups. According to the interpretation ranges proposed by Cohen (1988), the effect size is large (*eta* = 0.725, *eta squared* = 0.527), indicating that the difference in means is notable at least in some groups. Group 1 obtained

the best score and proved to be more united, with similar levels of knowledge, which is why it stands out from the other two groups.

Groups 2 and 3 have similar characteristics, but group 3 shows better performance, probably due to its smaller number of students, which allowed them to better organize and confront the problems that arose during the semester, thus outperforming the group. 2. Although the projects were different, the groups exhibit different characteristics depending on the skills they indicate they possess.

According to Fisher's exact test, the complexity of the project topic influences performance. According to the students' opinion, group 2 considered the project complex, while group 3 considered it to a lesser extent. However, due to the greater number of students in group 2, the hypothesis test was positive, in addition to the contrast with group 1, which did not consider the project complex. Therefore, it is concluded that the complexity of the project impacts the student's grade.

This is consistent with the research of Wang *et al* . (2022) and Luo *et al* . (2017), who indicate that the complexity of projects has a negative impact on their development, so a mechanism is required to reduce risk. Therefore, if at the beginning the project is considered complex by the students, it will be necessary to modularize the project and start development in iterations or use other mechanisms to reduce complexity.

software project influences the student's academic performance. This makes sense, since technical knowledge represents one of the most important aspects to carry out the project, since it involves the use of basic tools such as the programming language and database managers; Without these, it is impossible for the project to finish properly. The strange thing is that a high percentage of students indicate that they do not have the technical knowledge to face this project, when for the semester in question they should already cover these needs. This causes the statistical test to be positive.

Despite the importance of technical knowledge, Sedelmaier and Landes (2014) maintain that not only this aspect is important, but it must also be combined with soft skills to have comprehensive knowledge. The results of this research coincide with those of Ceh - Varela *et al* . (2023), who claim that PBL allows them to increase their technical skills, necessary in real-life jobs related to software *development* . In turn, Adorjan and Solari (2021) explain that, although the fundamental techniques of *software engineering* are stable, technological change continually impacts the development platform, *software execution* , and development tools.

According to the hypothesis test, it is confirmed that methodological knowledge is related to student performance. Methodological knowledge is fundamental to *software* development , as it dictates the organization necessary to complete the task. However, it is considered less significant than the previously mentioned technical knowledge, since the organization can be approached using methodologies other than those specific to the *software* , as long as there is order in the work and in the assignment of tasks. This allows students to cover the needs with prior knowledge they possess, although some students could not find an adequate form of organization, which led them to consider not having the methodological knowledge necessary for the project. This is reflected in the fact that those students who claimed to have methodological knowledge also obtained high grades, while those who claimed not to have it obtained low grades.

software development is not a simple task, so there are numerous development methodologies, some of which have successes and others errors , so sometimes the developer's intuition and enthusiasm for doing their job well are valuable. Still, having guides is quite useful, especially for novice developers (O'Regan , 2022).

According to the results, the hypothesis "The organizational capacity of the group to develop projects influences the student's academic performance" was negative, since there was disagreement among the students as to whether their organizational capacity was sufficient to address the development needs of the student. project. Some people who agreed with the statement got good grades regardless, while others who disagreed also got good grades, indicating the lack of a clear trend for the hypothesis to be positive.

Finally, while certain tasks in *software development* can be executed in parallel, others must begin when the previous task finishes, which demands exceptional organizational ability to produce the best *software* at minimal cost (Alsaqqa *et al* ., 2020). Therefore, Ceh -Varela *et al* . (2023) conclude that communication between the leader and team members is of vital importance to identify application requirements effectively. In this sense, by using a PBL approach, students improved their capacity for autonomy, self-confidence, teamwork and learning skills.

Conclusions

Project-based learning (PBL) is a pedagogical approach in which students acquire knowledge and skills through the completion of concrete and meaningful projects. Instead of teaching concepts in isolation, PBL engages students in solving problems or completing tasks that allow them to apply and consolidate their learning more effectively. The main goal of PBL is to help students develop critical thinking, problem-solving, collaboration and communication skills, while fostering their motivation and enthusiasm for learning.

In this research, PBL was applied in the context of *software engineering* , specifically in a group project where students were organized into teams. In these, they commonly work in teams of three to four members in order to distribute tasks and assume various roles to develop the product. However, in this research it was proposed to organize students into work teams focused on specific *software* engineering roles and then each group was assigned a specific task within the process.

Now, regarding the research question formulated at the beginning of this research (what are the main competencies that can be highlighted in the practice of project-based group learning for software development *?*), the following can be indicated. Firstly, and from a theoretical point of view, it can be stated that *software development projects* using PBL allow students to learn fundamental skills for teamwork, effective communication and problem solving, which are highly valued. in the workplace, since they encourage collaboration to achieve common objectives. Furthermore, by working on concrete projects, students have the opportunity to apply and consolidate their theoretical knowledge in a practical and contextualized way, which significantly improves their understanding and retention of concepts.

Secondly, and from a practical point of view, this strategy allows students to develop practical technical skills, such as programming, problem solving and the use of specific tools and technologies. This experience gives them the opportunity to acquire practical skills directly related to the *software industry* , thereby preparing them to face real challenges in their professional future. Additionally, this approach makes learning more dynamic and engaging, so it can be more interesting and positively impact your academic performance.

On the other hand, and more specifically, the aforementioned question can be answered from several angles. In this sense, it can be noted that, according to the analysis of general competencies, 1) this type of project should not be assigned to work groups with a low level of analysis and synthesis of information, and 2) it is recommended that the group

can work as a team effectively, given that this aspect is vital for the coordination and organization of the project.

software requirements engineering could be identified through observation and reviews . Furthermore, this competence is related to the general competence of analysis and synthesis of information.

Finally, regarding the analysis of the results of the hypotheses, the following can be concluded:

1. Differences in knowledge levels were observed between the study groups, as group 1 reached higher levels, while groups 2 and 3 showed a balance towards lower levels.
2. The complexity of the subject was identified as a determining factor for obtaining good results, which suggests that the topics to be developed must be easy to understand for students.
3. It was evident that this type of project works better when students have adequate technical knowledge, which highlights the importance of evaluating this item before applying PBL in a group.
4. It is crucial to have methodological knowledge of *software development* to properly carry out the project.

Future work

This research has identified several important aspects of the topic of study, but has also generated areas of future research. Therefore, it is proposed, first, the application of a second test to contrast the results obtained in this study and reach more solid conclusions, which would allow us to avoid possible biases that could arise in the first investigation.

Secondly, it is suggested to replicate the same project with another group of students to compare their behavior and the results of the project, which would help confirm the skills that are necessary for the groups to effectively use project-based learning.

Finally, as a third line of research, the possibility of developing a series of guidelines for the implementation of project-based learning in different groups is proposed, and then testing the effectiveness of these guidelines in another practical case.

References

Abella García, V., Ausín Villaverde, V., Delgado Benito, V. y Casado Muñoz, R. (2020). Aprendizaje basado en proyectos y estrategias de evaluación formativas: percepción de los estudiantes universitarios. *Revista Iberoamericana de Evaluación Educativa*, *13*(1), 93. https://doi.org/10.15366/riee2020.13.1.004

Adorjan, A. and Solari, M. (2021). Software Engineering Project-Based Learning in an Up-To-Date Technological Context. *2021 IEEE URUCON, URUCON 2021*, 486–491. https://doi.org/10.1109/URUCON53396.2021.9647348

Al Mulhim, E. N. and Eldokhny, A. A. (2020). The Impact of Collaborative Group Size on Students' Achievement and Product Quality in Project-Based Learning Environments. *International Journal of Emerging Technologies in Learning (iJET)*, *15*(10), 157–174. https://doi.org/10.3991/ijet.v15i10.12913

Alsaqqa, S., Sawalha, S. and Abdel-Nabi, H. (2020). Agile Software Development: Methodologies and Trends. *International Journal of Interactive Mobile Technologies (iJIM)*, *14*(11), 246–270. https://doi.org/10.3991/ijim.v14i11.13269

Asociación Nacional de Instituciones de Educación en Tecnologías de Información (ANIEI) (2022). *Asociación Nacional de Instituciones de Educación en Tecnologías de Información*. http://www.aniei.org.mx/ANIEI/

Ceh-Varela, E., Canto-Bonilla, C. and Duni, D. (2023). Application of Project-Based Learning to a Software Engineering course in a hybrid class environment. *Information and Software Technology*, *158*. https://doi.org/10.1016/j.infsof.2023.107189

Chimbo Jumbo, J. J. y Larreal Bracho, A.J. (2023). Metodologías educativas para el desarrollo de competencias científicas. *Ciencia Latina Revista Científica Multidisciplinar*, *7*(1), 7021-7048. https://doi.org/10.37811/cl_rcm.v7i1.4942

Cohen, J. (1988). *Statistical Power Analysis for the Behavioral Sciences* (2nd ed.). Lawrebce ErlBaum Associates.

Consejo Nacional de Acreditación en Informática y Computación (CONAIC) (6 de febrero de 2023). *Consejo Nacional de Acreditación en Informática y Computación*. https://www.conaic.net/

Cyrulies, E. y Shamne, M. (2021). El aprendizaje basado en proyectos: una capacitación docente vinculante. *Páginas de Educación*, *14*(1), 1-25. https://doi.org/10.22235/pe.v14i1.2293

Da Cunha, P. R. (2005). Teaching Software Engineering using Project-Based Learning. *Exploring Innovation in Education and Research*, 1–4. https://www.researchgate.net/publication/215575463

Flores-Ruiz, E., Miranda-Novales, M. G. y Villasís-Keever, M. Á. (2017). El protocolo de investigación VI: cómo elegir la prueba estadística adecuada. Estadística inferencial. *Revista Alergia México, 64*(3), 364. https://doi.org/10.29262/ram.v64i3.304

Gómez Álvarez, M. C., Manrique-Losada, B. y Gasca-Hurtado, G. P. (2015). Propuesta de evaluación de habilidades blandas en ingeniería de software por medio de proyectos universidad-empresa. *Revista Educación en Ingeniería*, *10*(19), 131–140.

Hernández-Sampieri, R. y Mendoza, C (2018). *Metodología de la investigación. Las rutas cuantitativa, cualitativa y mixta*. Editorial Mc Graw Hill Education

Luo, L., He, Q., Jaselskis, E. J. and Xie, J. (2017). Construction Project Complexity: Research Trends and Implications. *Journal of Construction Engineering and Management*, *143*(7), 4017019. https://doi.org/10.1061/(ASCE)CO.1943-7862.0001306

Macías, J. A. (2012). Enhancing project-based learning in software engineering lab teaching through an e-portfolio approach. *IEEE Transactions on Education*, *55*(4), 502–507. https://doi.org/10.1109/TE.2012.2191787

Marnewick, C. (2023). Student experiences of project-based learning in agile project management education. *Project Leadership and Society, 4*, 1-10. https://doi.org/10.1016/j.plas.2023.100096

Morais, P., Ferreira, M. y Veloso, B. (2021). Improving Student Engagement With Project-Based Learning: A Case Study in Software Engineering. *IEEE Revista Iberoamericana de Tecnologías del Aprendizaje, 16*(1), 21-28. https://doi.org/10.1109/RITA.2021.3052677

Nurbekova, Z., Grinshkun, V., Aimicheva, G., Nurbekov, B. and Tuenbaeva, K. (2020). Project-Based Learning Approach for Teaching Mobile Application Development Using Visualization Technology. *International Journal of Emerging Technologies in Learning (iJET), 15*(8), 130-143. https://www.learntechlib.org/p/217072/.

O'Regan, G. (2022). *Concise Guide to Software Engineering. From Fundamentals to Application Methods* (2nd ed.) Springer.

Pérez, B. and Rubio, a. L. (2020). *A Project-Based Learning Approach for Enhancing Learning Skills and Motivation in Software Engineering*. 51st ACM Technical

Symposium on Computer Science Education (SIGCSE '20), 309–315. https://doi.org/10.1145/3328778.3366891

Sánchez, P. y Blanco, C. (2012). *Implantación de una metodología de aprendizaje basada en proyectos para una asignatura de Ingeniería del Software.* XVIII Jornadas de Enseñanza Universitaria de la Informática, 41–48. http://www.comunio.es/

Sedelmaier, Y. and Landes, D. (2014). *Software engineering body of skills (SWEBOS).* 2014 IEEE Global Engineering Education Conference (EDUCON), 395–401.

Sotomayor, C., Vaccaro, C. y Téllez, A. (2021). *Aprendizaje basado en proyectos: un enfoque pedagógico para potenciar los procesos de aprendizaje hoy*. Fundación Chile. Centro de Innovación del Ministerio de Educación y el financiamiento de la Embajada de Estados Unidos en Chile.

Souza, M., Moreira, R. and Figueiredo, E. (2019). *Students perception on the use of project-based learning in software engineering education.* SBES '19: Proceedings of the XXXIII Brazilian Symposium on Software Engineering, 537–546. https://doi.org/10.1145/3350768.3352457

Wang, T., Chan, A., He, Q. and Xu, J. (2022). Identifying the gaps in construction megaproject management research: a bibliographic analysis. *International Journal of Construction Management*, *22*(9), 1585–1596. https://doi.org/10.1080/15623599.2020.1735610

Tomczak, M. and Tomczak, E. (2014). The need to report effect size estimates revisited. An overview of some recommended measures of effect size. *TRENDS in Sport Sciences*, *1*(21), 19–25.

Villalobos-Abarca, M. A., Herrera-Acuña, R. A., Ramírez, I. G. and Cruz, X. C. (2018). Real project-based learning applied to Software Engineers' education. *Formación Universitaria*, *11*(3), 97–112. https://doi.org/10.4067/S0718-50062018000300097

Zambrano Briones, M. A., Hernández Díaz, A. y Mendoza Bravo, K. L. (2022). El aprendizaje basado en proyectos como estrategia didáctica. *Conrado, 18*(84), 172-182.

Contribution Role	Author(s)
Conceptualization	Alan Ramirez-Noriega
Methodology	José Mendivil-Torres, Samantha Jiménez (same)
Software	Does not apply
Validation	Carolina Tripp-Barba, José Alfonso Aguilar-Calderón (same)
Formal Analysis	Carolina Tripp-Barba, José Alfonso Aguilar-Calderón (same)
Investigation	José Alfonso Aguilar-Calderón
Resources	José Mendivil-Torres
Data curation	José Mendivil-Torres
Writing - Preparation of the original draft	Alan Ramírez-Noriega, Carolina Tripp-Barba (same)
Writing - Review and editing	Samantha Jiménez, Carolina Tripp-Barba (same)
Display	Alan Ramírez-Noriega, José Mendivil-Torres (same)
Supervision	Alan Ramirez-Noriega
Project management	Alan Ramirez-Noriega
Fund acquisition	José Mendivil -Torres, Alan Ramírez-Noriega, Carolina Tripp-Barba, José Alfonso Aguilar-Calderón, Samantha Jiménez (same)