Enhancing the student perception on Software Project Management in Computer Science

Antonia Mas, Antoni-Lluís Mesquida, and Ricardo Colomo-Palacios, *Member, IEEE*

***Abstract*—*Contribution*: This paper presents a research for enhancing the computer science students' perception on the subject Software Project Management. The main objective of this research is to encourage students to learn the Project Management competences and skills and increase their academic performance. The design of the new Software Project Management course presented in this paper has been created based on the improvement proposals from the students.**

***Background*: It is very important that students are motivated with both the technical and other transversal competences provided by the different subjects in the Computer Science curricula. It is not easy to teach a management subject because students do not clearly find the relationship with their studies. However, it is widely accepted that computer science students need to be trained in Project Management to be able to work at full capacity in the industry. There is not much literature on how to increase the motivation of students in management subjects.**

***Research Question*: Which are the causes and consequences of a low level of interest for Software Project Management in Computer Science studies? The final goal is to enhance the student perception on Software Project Management in the Computer Science curricula.**

***Methodology*: Two different experiments were carried out. The first one consisted of determining, during two academic years, the main reasons why computer science students do not get to comprehend the need for such this subject in the curricula. From the Grounded Theory results, a set of innovative actions were introduced in order to re-design the subject. In a second experiment, during the following two academic years, the satisfaction of the students regarding the changes introduced in the course was analyzed.**

***Findings*: The new course design contributes to student motivation and engagement. They can better understand project management techniques and their application to real projects. Students are aware of the importance of these skills and competencies for their background and professional profile.**

***Index Terms*—Computer science students, software project management, student engagement, student's learning perception, student performance**

I. Introduction

SOFTWARE project management has found its place in the Computer Science curricula [1] in many universities across the globe. Although the SWEBOK Guide [2] contains some knowledge areas related to software project management, recent studies [3], [4] have shown that the PMBOK Guide [5] and the PMBOK Software Extension [6] can complement SWEBOK to create a robust software project management course to train current students and future project managers. Effective software project managers are not born but made through education [4].

Some authors consider that software project management is hard to teach and thus, is often taught on an abstract level, delivering fundamental knowledge about planning activities, estimation and controlling [7]. Therefore, it is convenient that educators design software project management courses including project driven learning [8]–[10] and group dynamics in order to prepare students for "the real world". Different studies proposing exemplar software project management courses and describing the experience after years of application have been found [7], [10]–[12]. Some of them emphasize not only the importance of adapting the contents to the student training needs, but also point out that some pedagogical approaches are inappropriate for teaching the subject. As a result, many students seem unwilling or unable to take the course seriously, that is, attending all classes and completing all assigned readings, activities and projects [11]. Moreover, other authors highlight that Industry engagement in the teaching of software engineering is essential. Guest contributors are often much better placed to teach practical software engineering skills than pure academics [13].

According to our perception, after more than fifteen years of teaching a software project management course, software engineering students tend to be more motivated by other technical subjects than by project management. They seem to prefer subjects specifically related to the design of new hardware elements or to the development of new software modules. We have observed that the student's motivation for learning project management is not usually very high. Before having the first contact with the course, a high percentage of students do not know the role of project manager in the software industry. In addition, they consider very far in time the moment of applying the skills and competencies acquired in this subject. Therefore, teaching software project management is a challenge [14] that requires a strategy to motivate students and to actively involve them for successfully taking the subject, for obtaining better learning outcomes and professional development.

With the main goal of improving student performance in our Software Project Management course, in 2015 we initiated a research project in order to get to know the real student perception on this subject. In our context, software project management is a mandatory third-year first-semester six ECTS (European Credit Transfer System) subject in the B.Sc. in

Computer Science. This degree has been awarded with the EURO-INF seal, the international engineering accreditation granted by the EQANIE. Six ECTS are equivalent to 60 face-hours plus 90 personal-study hours. 30 out of the 60 face-hours are oriented to the introduction of the project management best practices. There is one two-hour theoretical session per week over the 15 weeks of the semester. The remaining 30 face-hours are for 15 practical sessions, one practical two-hour session per week. In these practical sessions, students are divided in groups of approximately 20 students per class.

During two academic courses (2015-16 and 2016-17) we undertook a Grounded Theory study with our students to answer the following research question (RQ): *Which are the causes and consequences of a low level of interest for Software Project Management in Computer Science studies?*

From the results of this study, we were able to re-define our course by including some new motivating elements. The new version of the course has been validated during the last two academic years (2017-18 and 2018-19). During these two years, the student perception on software project management has been measured by analyzing four different aspects: student's learning perception, attitude towards software project management, complexity perception, and student performance.

In the literature, many initiatives related to improving student's learning by re-designing courses have been found. Some approaches are related to technical subjects [15]–[17]. Other studies show strategies regarding the development of transversal capabilities, such as leadership, teamwork, decision-making, negotiation, and self-reflection [18], [19]. In [20], models to improve both technical and non-technical skills can be found. The seven dimensions of socio-cultural challenges faced in Global Software Engineering Education [21] can be taken into account when designing a new course. Some studies are related to the use of techniques to improve student learning, effectiveness and motivation [22], [23], while others discuss the benefits of including serious games when teaching a course to undergraduate students [24]. Studies regarding theoretical foundations and empirical investigations about software project management education for software engineers can be also considered [25]–[30]. Moreover, the assessment of management competencies for software engineers [31] and other skills in management education can be found in literature [32], [33].

This paper presents the results we have obtained during the last four years and is structured as follows. Section II describes the research based on Grounded Theory carried out to detect the reasons that influence students to consider software project management as unimportant. Section III presents the results obtained in this research and indicates the actions proposed to re-define the course. Section IV details how the new course design was evaluated during two academics years and Section V shows the results obtained. Section VI opens discussion about the experience and outcomes achieved. Section VII concludes the paper.

II. Research Method

The empirical study about our computer science students' perception on software project management was based on the classic Grounded Theory (GT) proposed by Glaser and Strauss [34]. GT is a research method where theory is generated from the obtained data. This method produces a set of integrated conceptual hypotheses systematically generated to inductively develop a theory about a substantive area [35]. In GT, the research question is formulated during the research process, and not as a result of an extensive literature review done in advance, prior to initiation of the research [36].

A. Data Collection

Data collection is the first phase in the GT method. It is repeated all the time through the research process up to the point of data saturation.

In this research, we used semi-structured interviews to gather primary data (topics to talk about during the interview can be found in Table 4). Two researchers were involved in the data collection process, while the third one was involved in data analysis. All the data collected were reviewed separately by two researchers, and differences in findings were discussed among all three of them. Having three researchers in this study was important for objective data collection, conceptualization and analysis.

It was agreed to perform initial data collection activities during the course 2015-16, from September 2015 to February 2016. 55 of the 63 students enrolled in the course (representing the 87.3% of the sample) were interviewed. The time schedule of the interviews was created in a manner to have enough time to analyze and conceptualize the obtained data before continuing with other interviews. Interviews lasted for fifteen minutes on average. No more than five interviews were conducted in a single day. The planned pace was well estimated having in mind the need for constant comparison, memoing and conceptualization as a prerequisite of the GT method.

Before the start of each interview, the interviewees gave their consent to record the whole conversation and to allow the creation of transcripts and conclusions for scientific purposes. During the interviews, students were guided to discover their opinion about the software project management course. The participants had the opportunity to share their concerns, describe changes, improvements and drawbacks, and also to offer possible strategies for the definition of a new version of the software project management course. The transcribed data were constantly analyzed, compared and reviewed in researcher pairs as proposed by the GT method.

After the 55 interviews, a preliminary core category was identified. The adequacy of this core category was contrasted with 53 of the 65 students (representing the 81.5%) in the next academic course 2016-17. In this second data collection phase, from September 2016 to February 2017, some items were added (1 and 15) to the questionnaire (Table 4) while others were removed. As an additional change in line with the validation of the core category, the researchers made sure to spend more time with items 2, 3 and 4 during this second phase of data gathering activities (5 minutes as a minimum).

In total, during the two academic courses, the researchers participated in interviews with 108 students summing up to 24

hours and 18 minutes.

B. Data Analysis

GT's data analysis, also called data coding, starts right after first data is collected and it is carried up to the point of emergent theory creation [34]. A code represents the essential relationship between data and theory. Coding breaks the data, and conceptually regroups the data into codes becoming theory, thus explaining the relations among the data. In our research, data analysis was divided into three phases as follows:

1) Open coding, constant comparison and theoretical memoing

Open coding began with line-by-line coding of the transcribed data. Identifying and focusing on key points, instead of words, was found suitable in this study. Line-by-line coding led to the first open codes. Through open coding, key points were unveiled and codes were assigned to the interview transcripts. Some examples of open codes could be "complexity", "lack of experience" or "knowledge difficult to apply".

Concept is the naming of an emergent social pattern grounded in research data [36]. New concepts were developed through repetitive reading of the transcripts, and each key point was compared to the previous key points. Constant comparison process is a fundamental concept of GT [34]. During the open coding process different concepts were identified: "student dedication", "complexity perception", "attitude towards software project management", etc. Through the constant comparison method open codes were grouped in the emerged concepts, and after a few iterations of reorganization and concept fine tuning, categories were identified.

After conducting each interview, the researchers created memos focusing on the identified concepts. Memos are theoretical notes about data and relations between categories such as ideas, feelings, thoughts and concerns related to the concept [35]. Theoretical memoing was done throughout the whole data analysis process.

At the final step of this first phase, categories emerge from a group of concepts. Each category represents a group of concepts that are associated with the same phenomena. From the different categories identified, the core category was selected. The core category should have the most explanatory power, clearly present the problem that is the focus of the study [35] and it should encapsulate and explain the GT as a whole.

Three potential core categories emerged from the open coding process: "Students lack management competencies and experience", "Students do not consider software project management an engineering subject" and "Students do not give enough importance to software project management". The core category identified, "Students do not give enough importance to Software Project Management", had a well-developed range of properties and dimensions, and it was the best candidate to integrate the other identified categories.

2) Selective coding

Selective coding was performed after open coding. In this phase, researchers continued with the data collection process, but with a focus on collecting data on categories for the generation of properties and hypotheses. Selective coding was done in the same manner as open coding, but now, it was more "selective" as the focus was to obtain and analyze the data related to the categories identified in the open coding phase.

During this iterative process, the researchers went back to the data from the interviews in order to reorganize and saturate the categories, concepts and relations among them.

Selective coding is terminated when theoretical saturation is identified. Categories are saturated when the process of gathering new data returns codes that only fit in the existing categories and these categories are sufficiently clarified in terms of their dimensions and properties [37], no longer sparks new theoretical insights, nor reveals new properties of the core theoretical categories. After the second round of interviews during the course 2016-17, the authors identified theoretical saturation and therefore, selective coding was terminated.

3) Sorting, integration and theory write-up

Once theoretical saturation is achieved, researchers should continue with review, sorting and integration of previously created memos associated with the core variable, its properties and related categories [35].

According to [38], four types of generalizing and generalizability may be found: generalizing from data to description, generalizing from description to theory, generalizing from theory to description and generalizing from concepts to theory. In this research, the resulting theory was generalized from concepts.

At this data analysis phase, the obtained data are sorted, merged and a theory as a result of the GT process is written up. The main question in this phase was how to relate and enrich the data as a cohesive whole. During this phase, all the memos were printed out and reviewed by each researcher separately with a focus on fitting them in the theory obtained. After the researchers finished reviewing the separate findings, they jointly integrated and enriched the theory after reaching consensus on all the potential improvements identified.

Glaser offers 18 families of theoretical codes or paradigms helping the researcher to think analytically and set relations among concepts [39]. Coding paradigms are conceptual templates for organizing categories and presenting how they relate to one another. In order to organize and demonstrate the discovered categories and relations with the core category, Glaser 6C's model [34] was found appropriate.

III. Grounded Theory Results

This section describes the gathered results of the GT study conducted. The 6C's coding paradigm was used for presenting the results (Figure 1). It consists of the following instances: Context, Conditions, Causes, Consequences, Contingencies and Covariance.

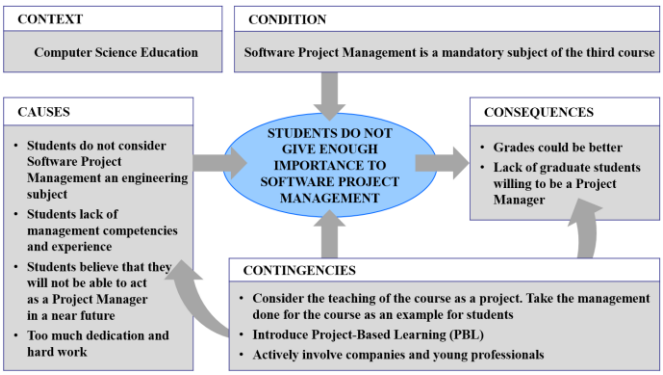

Fig. 1. Theory of Students do not give enough importance to Software Project Management.

Context describes the setting when the core category occurs. In this research, Context is Computer Science education. Conditions define the requisites for the core category to emerge. The condition is that Software Project Management is a mandatory subject of the third course. In the following sections Causes, Consequences and Contingencies are presented. Since the research was conducted in one university, Covariance is not presented in the 6C's model.

A. *Causes*

Causes define the reasons for the core category to occur. Four causes influencing students to do not give enough importance to software project management were identified:

1) Students do not consider Software Project Management an engineering subject. Students do not get to understand why such a management course is included in the syllabus. Due to a lack of knowledge regarding project management in general and software project management in particular, misconceptions exist [4]. They feel closer and more comfortable in technical subjects in which they are supposed to develop and obtain tangible products, either a functional software or a working hardware. It seems that software project management is not perceived as a real engineering process.

2) Students lack of management competencies and experience. Students lack of real experience identifying tasks and establishing dependencies among them, defining milestones, making budgets, defining roles, etc. They have not worked these skills in any other previous course. The results from [8] show that students struggled in their people handling skills, negotiation skills and organizational skills.

3) Students believe that they will not be able to act as a Project Manager in a near future. Students consider that the competencies acquired in the course will not be applicable in the short term. They believe that they will not participate in any aspect of project planning until they occupy positions of responsibility, something they consider distant future. They can imagine themselves in programmer or analyst positions, but not yet in Project Manager positions.

4) Too much dedication and hard work. Students consider that the number and variety of concepts is too high to be assimilated and applied in such a short period (one semester). Moreover, in this course, students have to imagine and define a project to be planned and monitored, but without executing the tasks neither obtaining the planned deliverables. It is difficult for students to make this simulation effort. In addition, they appoint that other cross-curricular competencies which are also assessed in this course, such as teamwork, communication and presentation of results and use of project management software, imply a huge extra effort.

B. *Consequences*

Consequences define the effects of the occurrence of the core category. We identified two consequences:

1) Grades could be better. Students do not make the appropriate effort. They do not work as hard as in other courses in which they have to implement a software module or to develop a hardware element. The grades obtained could be higher. According to [11], the percentage of students who do not take their education seriously greatly exceeds the failure rate most institutions will accept.

2) Lack of graduate students willing to be a Project Manager. Few students state they would like to play the role of a Project Manager in their career. The vast majority consider that if they need to put into practice management skills in their professional future, they could be trained later, when it is necessary.

C. *Contingencies*

The main goal of the enlisted contingencies is to moderate the causes influencing the core category and the consequences arising from the core category. Three different contingencies were identified:

1) Consider the teaching of the course as a project. Take the management done for the course as an example for students. First contingency was focused on considering the teaching of the course as a project. In higher education, a project may consist of organizing and teaching a course for an academic year. Then, the idea was to apply project management best practices to the whole teaching process. In this way, students could understand better what does software project management mean and, moreover, perceive the benefits of a course taught in a managed manner. Teachers decided to introduce practical examples of how different project management concepts had been applied in order to manage the course. Some examples could be the schedule of the sessions showing the contents of each class, a work breakdown structure containing all the lessons during the semester, or the distribution of teachers assigned to each class.

Managing a course involves activities that go beyond the preparation of lectures, supervising projects or preparing and correcting exercises and exams. These activities are part of what we called "Course life cycle", which starts when the course is proposed to be part of a syllabus. At this time, the purpose, scope and content are outlined, and the project is ready to start. However, once the course is assigned to a teaching team, is when the project really begins. Now, it is time to define and detail the scope, to generate the schedule according to the academic calendar, to distribute the contents of the sessions among the professors, to determine all the infrastructure and resources that will be necessary, or to identify the risks that may threaten the proper delivery of the course. All these activities

are included in the phase we called "Organizing and Preparing", which is aligned to the planning stage in a project management context.

Once the academic year begins, it is when the activities planned in the previous phase should be carried out. This set of tasks belongs to the phase we called "Teaching", which coincides with the execution and control stages in a project management context.

Once all the assessment activities have been done, the course is closed. It is time for the lessons learnt to be collected and suggestions for improvement to be taken into account. We are therefore in the phase we called "Ending", which is aligned to the closing stage in a project management context.

For each phase of the course life cycle, different management supportive assets were created. The assets in Table 1 are provided to students during the lessons for them to have examples of application of the project management techniques to one real project: the course they are enrolled in. For the design of these assets, the authors took the guidelines of the PMBOK Guide [5] as a basis. Table 1 relates each exemplar asset with the specific project management process and knowledge area.

TABLE 1
EXEMPLAR PROJECT MANAGEMENT ASSETS

Course life cycle phase (PM stage)	Asset	Related PMBOK®	
		Process/es	Knowledge area
Organizing and Preparing phase (Planning stage)	Course teaching guide	4.2 Develop Project Management Plan	4. Project Integration Management
	Course document of requirements	5.2 Collect Requirements	5. Project Scope Management
	Course WBS (Work Breakdown Structure)	5.3 Define Scope 5.4 Create WBS	
	Course schedule	6.2 Define Activities 6.3 Sequence Activities 6.4 Estimate Activity Durations 6.5 Develop Schedule	6. Project Schedule Management
	Course quality plan	8.1 Plan Quality Management	8. Project Quality Management
	Course resource needs	9.2 Estimate Activity Resources	9. Project Resource Management
	Distance-learning platform	10.1 Plan Communications Management	10. Project Communications Management
	Risk register	11.2 Identify Risks 11.3 Perform Qualitative Risk Analysis 11.4 Perform Quantitative Risk Analysis 11.5 Plan Risk Responses	11. Project Risk Management
	Material and infrastructure needs	12.1 Plan Procurement Management	12. Project Procurement Management
	Course stakeholders record	13.1 Identify Stakeholders	13. Project Stakeholder Management
	Student management plan	13.2 Plan Stakeholder Engagement	
Teaching (Execution and monitoring stages)	Course knowledge register	4.4 Manage Project Knowledge	4. Project Integration Management
	Change register	4.6 Perform Integrated Change Control	
	Assessment survey of the teaching task	8.2 Manage Quality 8.3 Control Quality	8. Project Quality Management
	Student engagement matrix	13.3 Manage Stakeholder Engagement	13. Project Stakeholder Management
Ending (Closing stage)	Lessons learnt register	4.7 Close Project or Phase	4. Project Integration Management

2) Introduce Project-Based Learning (PBL). To prepare the students for the 'real world' educators frequently use PBL approaches in their curricula [8]. Students should learn by doing and, wherever possible, assessed in the context of practical work [13]. The second contingency was the introduction of PBL to support students to apply all the knowledge, techniques and tools introduced during the theoretical sessions to their projects. PBL is a cooperative learning methodology based on students learning together while helping other colleagues. Many studies have shown that using PBL, students can simulate situations closer to a professional environment and are better prepared to carry out their future work in a company [16], [17]. Moreover, PBL facilitates the acquisition of cross-curricular competences and skills such as group cooperation, teamwork, leadership and presentation skills. For this contingency, we also considered Reflexive Weekly Monitoring (RWM) [18] together with PBL, as RWM positively influences coordination among team members, increases student's sense of belonging to a team and increases team's productivity.

Our PBL approach was applied to the 15 practical sessions of 2 hours, with approximately 20 students. The project workload per student is estimated at 90 hours, including these 30 hours for the practical sessions and 60 additional hours for out-of-class work: 30 hours of individual work and 30 hours of teamwork.

PBL was designed to be carried out in teams of four students. According to [40], they are free to form their own teams, as teachers prefer not to participate in this decision. If there are some students without a team, teachers encourage them to form a new team or to complete other existing ones. Each team should be headed by a member with the role of Project Manager. This student should be efficient in dividing the tasks among all team members in order to complete the work to deliver and meet the deadlines.

Our proposed PBL approach considers the previous experiences by [41] and [4]. It consists of the following tasks (TX) and milestones (MLX):

- *T1*: Students have to propose a software project. They are not supposed to develop the product or result, but they are supposed to accurately plan the project and simulate the monitoring and control of the project execution.
- *ML1*: Project approved. The project has to be presented and agreed with teachers in order to validate the project scope.
- *T2*: Students have to weekly apply to their own project the concepts, techniques and tools introduced during the theoretical sessions. They have to deliver on time though the Moodle platform, all the required planning

assets, such as a Work Breakdown Structure, a schedule, a budget or a communication plan, among others.

- *T3*: Students have to present the results obtained during the week. Each team can decide which member will present the results. The only condition is that all students must have participated equally at the end of the semester. Each team receives questions or improvement suggestions from both the teachers and other teams. Students can iteratively improve their results by observing how other colleagues have solved the same activities.
- *T4*: Students have to improve the results (if it is required) and deliver again the results.
- *ML2*: Project finished and delivered.
- *T5*: At the end of the semester, students have to defend their project in a one-hour assessment session. Each team has to prepare a 15-20 minutes project closing presentation in order to introduce how the project ended according to all the project management areas. Teachers decide which student has to perform the presentation. Afterwards, teachers ask individual questions to all team members (10 minutes per student). T5 activity is very useful to know if the learning objectives have been reached, and to assess both the technical competences and other transversal skills.
- *ML3*: Project passed. If ML3 is not reached, students can deliver the project in the next call, only after having addressed all the weaknesses detected during *T5*.

3) Actively involve companies and young professionals. Industry engagement in the teaching of software engineering is essential [13]. The third contingency consists of scheduling different sessions involving software development companies and professionals in this sector. Three different sessions with external invited speakers were planned. The first invited session was scheduled to be delivered by a young former student who acts as a project manager in a software company. This talk should focus on two different goals: 1) to describe the motivation that made her to choose the role of Project Manager and 2) to introduce the tasks that performs in her day to day. This way, students can realize that a project manager applies many of the knowledge and abilities learnt during this course, such as defining tasks, estimating the effort for carrying them out, defining the competencies of required human resources, among many others.

The second invited session was planned to show to students how a Project Management Office (PMO) works. This session was scheduled to be delivered by a company in the tourism sector with a computer department of more than 100 professionals and 3 full-time PMO employees. The goal of this talk is that students can realize that the contents of the course are fully aligned with the tasks that are carried out by a PMO.

Last session of the semester was scheduled to be delivered by the staff of the company *SM2 Software & Services Management*, which has developed TALAIA OpenPPM, an open source solution for Project Portfolio Management. In this session, students can learn the functionality of this software tool and how all the management areas are addressed by each of its modules.

D. Contingencies

The moderation of each contingency on the appropriate cause or consequence is represented in Table 2.

TABLE 2
CONTINGENCIES AND THEIR RELATION TO THE CAUSES AND CONSEQUENCES

Contingency	Cause	Consequence
Consider the teaching of the course as a project. Take the management done for the course as an example for students	Students do not consider Software Project Management an engineering subject Students lack of management competencies and experience	Grades could be better
Introduce Project-Based Learning (PBL)	Too much dedication and hard work Students lack of management competencies and experience	Grades could be better
Actively involve companies and young professionals	Students believe that they will not be able to act as a Project Manager in a near future	Lack of graduate students willing to be a Project Manager

IV. EVALUATION OF THE NEW COURSE RE-DESIGN

This section describes how the identified contingencies were considered in a re-designed software project management course, and how this new course was delivered during the academic years 2017-18 and 2018-19.

A. Participants

118 students participated in the new course evaluation (61 students in 2017-18 - 56 male and 5 female students; 57 students in 2018-19 - 54 male and 3 female students). A 93% were male students. A 76% of the teams were composed only by male students whereas a 24% of teams were mixed. Only seven teams had one female student and one team had two. All participants were of similar age, ranging between 20 and 24 years old. All students belonged to the same bachelor program, and thus they had comparable technical skills at the beginning of the course. Some students already worked as interns and, therefore, they may have had some interaction with project management. The teachers were the same for the whole study period (the two academic years).

B. Taking the management done for the course as an example for students

The project management supportive assets in Table 1 were provided to students for them to have real examples of the application of the PMBOK Guide project management techniques. Students provided feedback on them. The assets with the highest value for them were:

- *Course teaching guide*, with a complete description of

the course scope and the teaching methodology. According to the new course design introducing PBL, each team had to define its own project.

- *Course schedule*, composed by an activity list, deliverables and effort estimates for each activity. The schedule for the academic year 2018-19 is shown in Table 3. The estimated effort, both at team and individual levels, is informed to students in columns TEE (Team Estimated Effort) and IEE (Individual Estimated Effort). Students real dedication can be reported in columns TRD (Team Real Dedication) and IRD (Individual Real Dedication).

TABLE 3
COURSE SCHEDULE FOR THE ACADEMIC YEAR 2018-19

Week	Task/Deliverable	TEE	IEE	TRD	IRD
1	Project and team presentation *(Milestone ML1)*	3	2	3.8	2.1
2	Environmental factors and organizational assets	1	1	1.4	1.3
3	Project stakeholders	2	1	2.2	1.1
4	Project charter	1	1	1.2	1.2
5	Project scope	2	2	1.8	1.9
6	Project schedule	3	3	2.9	2.3
7	Project resources	2	2	2.1	1.9
8	Project communications	2	1	1.9	1.1
9	Project procurements	2	2	1.9	1.9
10	Project budget	2	2	2.3	1.9
11	Project risks	2	2	1.9	2.2
12	Project quality	2	2	2.1	2.1
13	Project monitoring and control	2	2	1.8	1.8
14	Project closing	2	2	2.4	2.2
15	Closing presentation (*Milestone ML2*)	2	5	2.1	5.9
Total		30	30	31,8	29

- *Course quality plan*, containing the assessment rubrics to be applied during each project supervision session (each week). A rubric to assess the whole project (used during the PBL task T5) is also provided to students.

C. Introducing Project-Based Learning (PBL)

The defined PBL methodology was applied. Students were grouped in teams of four people. Teams had to work continuously throughout the semester according to the schedule (Table 3). They had to weekly apply new project management techniques from each knowledge management area. They had to deliver on time (before each practical session) the weekly results. Randomly, some teams were required to present the obtained products during the practical sessions. If they had not fulfilled the expected results, they were able to deliver again the work, in a continuous improvement cycle.

Students were required to weekly report team and individual dedication by filling the columns TRD and IRD. In Table 3, the TRD column shows the average team dedication for each task reported by the students of the 2018-19 academic year. The IRD column shows the average individual real dedication. The last row of Table 3 shows the average real dedication to complete the project (60.8 hours per student of out-of-class work: 31.8 hours of teamwork and 29 hours of individual work), which is very similar to the estimated effort (60 hours per student: 30 hours of teamwork and 30 hours of individual work).

D. Involving companies and young professionals

The first invited session was delivered, during week 3, both years 2017-18 and 2018-19, by a 31 years-old ex-student that works as a Project Manager in a technology company for the tourism sector.

The second industry session was conducted during week 12, both years 2017-18 and 2018-19, by the PMO Responsible of a software development company of more than 200 employees mainly working in projects for the tourism sector.

Finally, the third industry session (at week 15) was delivered both years 2017-18 and 2018-19 by the head of unit of *SM2 Software & Services Management*, owner of TALAIA OpenPPM software tool.

V. DATA COLLECTION AND RESULTS

This section describes the data collection procedure and presents the data collected during the four academic years 2015-16 to 2018-19. Data were collected through student's responses to two anonymous questionnaires administered at the end of the semester. The first questionnaire (Table 4) was created to collect from the academic year 2015-16 onwards the general opinion of students regarding the course.

TABLE 4
OPINION ON THE COURSE

	2015 -16	*2016 -17*	**2017 -18**	**2018 -19**
1. The existence of this course within the syllabus is appropriate.	*1.9*	*2.2*	**2.7**	**2.9**
2. The course design has allowed me to acquire the basic knowledge about software project management.	*2.4*	*2.3*	**3.1**	**3.1**
3. The difficulty level of the concepts is appropriate.	*2.4*	*2.3*	**2.4**	**2.4**
4. The workload of the course is appropriate.	*3.3*	*3.5*	**2.7**	**2.6**
5. The knowledge acquired will help me to develop my professional career.	*2.4*	*2.5*	**3.0**	**3.1**
6. The exam is an effective tool to ensure that a student has learnt the concepts.	*2.8*	*2.6*	**2.9**	**2.8**
7. The project is an effective tool to apply the learnt concepts.	*2.6*	*2.4*	**3.1**	**3.0**
8. The way to assess the project is appropriate.	*2.6*	*2.5*	**2.7**	**2.6**
9. I will be able to apply the knowledge acquired in this course in my career.	*2.3*	*2.6*	**3.2**	**3.1**
10. I would like to play the role of Project Manager in the near future.	*2.7*	*2.9*	**3.6**	**3.6**
11. I would like that my Final Degree Project was related to software project management.	*2.4*	*2.3*	**2.7**	**2.7**
12. The complexity of this course is higher than most of the courses in the syllabus.	*3.6*	*3.5*	**3.7**	**3.5**
13. The dedication to this course is higher than most of the courses in the syllabus.	*3.8*	*3.6*	**3.5**	**3.3**
14. After completing this course, my interest in software project management has increased.	*2.4*	*2.5*	**2.7**	**2.8**
15. In case this was an optional course, I would recommend it to my colleagues.	*1.8*	*1.9*	**2.3**	**2.6**

The second questionnaire (Table 5) was created to collect the opinion of the students regarding the three contingencies

applied from the academic year 2017-18 onwards.

TABLE 5
OPINION ON THE CONTINGENCIES APPLIED

	2017-18	2018-19
Taking the management done for the course as an example for students		
1. Considering the course as a project, and using it as an example during the entire course, has helped me to better understand the concepts.	3.2	3.3
2. The provided assets are intuitive and have helped me in applying the concepts.	3.6	3.5
Introducing Project-Based Learning (PBL)		
3. PBL has helped me to understand better the concepts.	3.5	3.6
4. PBL has given a practical orientation to the course.	3.4	3.3
5. PBL has helped me to apply the concepts in a project.	3.7	3.6
6. I have learnt to work in a professional way, closer to the market.	3.3	3.4
7. PBL has helped me to keep the course up to date.	3.5	3.5
8. The equitable distribution of the work throughout the semester has been a positive fact.	3.3	3.4
9. PBL has helped me to work in a team in an effective way.	3.3	3.3
10. PBL has helped me to become aware of the importance of meeting the milestones in the established deadlines.	3.6	3.7
11. The weekly presentations have made me improve my communication skills.	3.3	3.4
12. Being able to see and discuss the results of my colleagues has enriched my own results.	3.7	3.6
Involving companies and young professionals		
13. The session with the young project manager has allowed me to know the existence of such this professional role.	3.5	3.4
14. The session with the young project manager has helped me to understand the importance of the course.	2.9	3.2
15. To know how a PMO works has been useful.	2.8	2.7
16. Observing the application of a project management software tool in a real organization has helped me to better understand the concepts worked in the course.	2.6	3.1
17. Being able to get in touch with business professionals is very enriching for my career.	3.4	3.6
18. The participation of companies is a good complement to the theoretical and practical sessions.	3.6	3.8
19. The participation of companies has helped to increase my interest in the course.	3.1	3.1

In both questionnaires, student's responses were measured based on a 4-point Likert scale. They had to indicate their level of agreement with each statement with one of the following options: (1) Not agree: 0%-15% agree; (2) Partially agree: 16%-50% agree; (3) Largely agree: 51%-85% agree and (4) Totally agree: 86%-100%.

To reduce a wrong interpretation of the statements, the used terminology was as near as possible to the student's language. The validity of the questionnaires was assessed with a sample of the study population. During the course 2017-18, the authors met with five students to collaborate in improving the statements of the first questionnaire and drafting the statements of the second questionnaire. Then, both questionnaires were given to other five students who were asked about their understandability. Some small changes were necessary until the final versions of the questionnaires were agreed.

VI. DISCUSSION

Based on other previous studies on the student attitude change [42], [11], [12] and [43], this section analyses the results gathered over the last four academic years. It discusses the influence of the applied contingencies on four different areas: student's learning perception, attitude towards software project management, complexity perception and student performance.

A. Student's learning perception

The student's learning perception was measured according the responses to items 2, 5, 7 and 9 from Table 4. In the courses 2015-16 and 2016-17, the values of these four items were below 3. However, in 2017-18 and 2018-19 the obtained values in all these items were equal or higher than 3.

Looking at the results of Table 5, we can deduce that the use of the PBL methodology has had a positive influence on the student's learning perception. Items 3 to 12, related to the use of PBL, had very good ratings. Since students can choose the projects they wish to manage, they are more involved in the course. This is because they propose projects that they really like, usually related to areas that attract their attention and in which they would like to work. Since they can choose their own project, students make a bigger effort in describing the scope and the functionality of the resulting product. Students consider PBL very valuable for developing skills such as teamwork, leadership, communication or coordination, which they consider very useful for their integration into the market.

The values of items 7 and 8 from Table 5 indicate that the use of RWM has had a positive impact for the students. Using this system of deliveries and continuous improvement, students know the complete schedule for the whole project, the workload and how much time is estimated for each delivery.

The values of item 13 from Table 4, related to the dedication to the course, reflect that students think they have to devote more time to software project management than to other courses with the same ECTS credits. During the semester, students are required to report the time used to perform the tasks, both individually and in teams. Time reported by the teams is never higher than 60 hours, which is the expected dedication to the project. Therefore, authors believe students have this perception because teachers in the rest of courses do not monitor student dedication. The fact of making students to deliver a dedication form makes them aware that they do not spend more hours than those planned in the teaching guide. Therefore, the workload is adequate and in accordance with the established ECTS credits.

Another positive fact related to the use of PBL and the weekly presentation of results obtained by each team is that the students can observe how the colleagues from other teams have given different solutions to the same problems. They like to participate in constructive criticism. Moreover, they can get the good ideas from colleagues for improving their own project and include them in the final delivery. Values of item 12 from Table 5 indicate that this fact has given great value to them and then, we could deduce that it has influenced their learning.

Another issue to consider is that a small number of students felt uncomfortable with oral presentations because they have fear of public speaking. Item 8 from Table 4, related to the way to assess the project, indicate that the final project defense

session generates stress. We believe that the traditional passive learning has still a lot of influence on the student's behavior.

B. Attitude towards software project management

Authors are aware that it can be difficult to state that the attitude of the students towards the course has improved, since the questionnaires have not been applied to the same students during the different academic years. However, attitude towards software project management can be perceived from the values of items 10 and 11 in Table 4. The increase obtained in 2017-18 and in 2018-19 in the number of students interested in completing a Final Degree Project related to software project management indicates a bigger interest of the students towards the course. An increase on the interest to take on the role of Project Manager can be observed as well.

The importance that students give to this course is related to items 1 and 15 in Table 4. We can see that during the course 2017-18, both values experienced an increase of more than a 20% with respect to previous years, when contingencies had not yet been applied. During the course 2018-19 the increase was even greater.

Authors also consider very positive the increased value of item 14 in Table 4 during the courses 2017-18 and 2018-19, since it allows to observe that after completing the course, the students' interest is greater than before.

Looking now at Table 5, we strongly believe that attitude towards software project management has been significantly improved. The obtained values for items related to the involvement of companies and young professionals (items 13 to 19), show that students highly appreciated the external participation, since speakers managed to capture they interest. With the participation of the young project manager, students could realize that there exist the professional role of project manager, of which they were unaware. With the PMO session, students had the opportunity to see how a real PMO in a software development company is structured and how it works. When they observed the business map and organization chart, they realized that professionals with their same studies occupy these roles. Students are very motivated when they see a real application of what they learn at the University. Finally, the session on the project management software tool motivates students doubly. On the one hand, because they can see implemented in a real tool the different project management areas they have worked with and, on the other hand, because they can deal with the team that has developed this software. Perhaps, the aspects related to the tool development attract them even more than seeing its functionality.

In summary, the integration of academia and industry, benefits undoubtedly the interests of all parties. In this case, has helped to improve the students' attitude towards software project management. Providing students with relevant skills and expertise to succeed and to bring students closer to the real professional environment it is a mission of the University. We truly believe that our experience has contributed to this goal.

C. Complexity perception

The perception on the complexity of the course has been collected using the items 3, 4 and 12 from Table 4. In general, it seems that students keep on considering this course difficult, since the value of item 12, comparing the complexity of this course with other courses in the syllabus, has not decreased. Although it may seem that the contents of this course are not complex, this is not completely true for the profile of a third-year computer science student, since:

- They are students of a technical degree and, in general, have little interest in management topics.
- They do not usually have work experience in real projects. Therefore, they need a non-short period for understanding the management areas and activities that are carried out in any project.
- Management tasks, in which they have participated at a particular level, have been performed unconsciously.

According to item 3 in Table 4, the implementation of the new course re-design has not led to a reduction in the perceived difficulty level of the concepts. However, and according to item 4, the workload perception seems to be more adequate. Therefore, from the values of the items related to the complexity of the course in Table 4, we cannot conclude it has been significantly reduced with the application of the three contingencies.

However, some values from Table 5 could be interpreted by the authors as a sample of complexity reduction. The values of items 1 and 2 seem to indicate that the exemplar assets really help students to understand the project management concepts and how they should be applied to their projects. Moreover, values of items related to the use of PBL (items 3 to 12) also have very good ratings, from 3.3 to 3.8. It seems that students understand better the project management concepts when they apply them to a project. To sum up, both contingencies have influenced to decrease student perception about the complexity of the course.

D. Student performance

Students have to pass two different parts in order to complete the course: the project and an exam. Traditionally, this exam has been composed of 30 sentences. Students must decide if each sentence is "true" or "false" and, in this last case, they have to correct it to make it true. Task T5, described in section three, corresponds to the project assessment activity. The final grade of the course is obtained from the average of the exam mark and the project mark, only if the student has passed both parts.

Table 6 summarizes the evolution of the grades over the last four academic years. The pass rate was over 75% in two first years, that is, 1 out of 4 students failed. The worst results were obtained in 2016-17, with a fail rate of 28%. During the last two years, with the new design of the course, the failure rate was reduced and, in this last course, only 1 out of 5 students failed. Moreover, the percentage of students who have obtained the grade "Good" has also increased, from 35% in 2016-17 to 43% in the last course. A similar increase has been obtained for the grade "Excellent".

TABLE 6
EVOLUTION OF THE COURSES GRADES

	2015-16	2016-17	*2017-18*	*2018-19*
Number of students	*63*	*65*	**57**	**61**
Fail (0-4.9)	*16 (25%)*	*18 (28%)*	**12 (21%)**	**12 (20%)**
Pass (5-6.9)	*26 (41%)*	*22 (34%)*	**20 (35%)**	**19 (31%)**
Good (7-8.9)	*18 (29%)*	*23 (35%)*	**22 (39%)**	**26 (43%)**
Excellent (9-10)	*3 (5%)*	*2 (3%)*	**3 (5%)**	**4 (7%)**

While it is true that the increase is discrete and has not led to a drastic change in the course success rate, teachers are satisfied and think that the application of the proposed contingencies has influenced this improvement. Moreover, since students are now more engaged with their projects, the project mark has improved and, consequently, the course global grades have been better.

Table 7 shows the comparison of the grades over the last four years, obtained from the average of the exam mark and the project mark. The number of students that pass the course is near the same number of students that passed the exam, as most (almost all) of them pass the project. Data in Table 7 demonstrates that it is easier for students to pass the project than to pass the exam. It seems that because students perform the project in teams, some members contribute more than others do. This fact can have some influence on the individual exam mark. However, it is not a goal of this research to analyze this aspect.

TABLE 7
EVOLUTION OF THE EXAM AND PROJECT GRADES

	2015-16	2016-17	*2017-18*	*2018-19*
Number of students	*63*	*65*	**57**	**61**
Number and (%) of students who passed	*47 (75%)*	*47 (72%)*	**45 (79%)**	**49 (80%)**
Average individual exam mark	*5.8*	*5.6*	**6.2**	**6.5**
Average team project mark	*7.2*	*7.4*	**8.2**	**8.6**
Number and (%) of students who passed the team project	*55 (87%)*	*57 (88%)*	**57 (100%)**	**60 (98%)**

The average exam mark was below 6 out of 10 in 2015-16 and 2016-17. However, it was greater of 6 in 2017-18 and 2018-19, after the course re-design. According to item 6 of Table 4, it seems that students consider the exam as an appropriate system for assessing their learning.

With the introduction of the PBL methodology, which favors the incremental development through weekly deliveries, students can make a continuous improvement of their project and achieve much better marks than before the application of the contingencies. The average project mark was 7.2 over 10 in in 2015-16. In the last course, the average was 8.6, a 14% higher. Students appreciate the continuous improvement method because it allows them to ensure they obtain a good project grade.

VII. CONCLUSION

This paper presents a wide research to engage computer science students towards software project management. Two empirical studies were conducted. The first one, based on Grounded Theory research, took place during the academic years 2015-16 and 2016-17, and was carried out to detect the reasons that influence students to undervalue the software project management course. The obtained findings allowed the introduction of some improvements in a course re-design. The identified contingencies were applied during the academic years 2017-18 and 2018-2019 and, in a second study, the student's satisfaction level regarding the new course design was analyzed.

From the data collected and the analysis performed, authors believe that the new design contributes to student motivation and engagement towards the course. The introduction of the PBL methodology and the sessions involving companies have greatly influenced the students' attitude towards the course. Moreover, students have observed the importance of cross-functional competencies for their background and professional profile. They have acquired a better understanding of the project management skills and their application to real projects. The course complexity perception has slightly decreased. The number of students who pass the course has increased only a 5%. However, the grades have increased by around a 15%.

The obtained values in items 1 and 15 of Table 4, during the last two academic courses, evidence an increase in the motivation of the students towards the course. For the authors, it is very positive to observe that during the last two years, the value obtained for the item on how appropriate is the existence of this course within the syllabus has increased a 23%. It is also very positive the value for the item about if students would recommend the course to their colleagues, increasing a 21%.

Authors had some limitations in the application of this research. In order to avoid the threats of validity of the Grounded Theory method, such as observational bias, the codification and analysis was carried out by three researchers. The items of the two questionnaires could be formulated by specialists to better capture the students' perception and obtain more precise data. Moreover, some of the suggestions about quality analysis of a teaching survey from [44] could be taken into account. Another limitation is that students do not usually put all the interest we would like when completing the questionnaires, as they do it as a routine. They are not completely aware of the importance that their inputs may have for future students.

Authors, although they continue applying and improving the teaching method detailed in this paper to deliver the practical sessions, are currently focused on defining strategies for teaching the concepts during the theoretical sessions.

ACKNOWLEDGMENT

This work has been supported by the Spanish Ministry of Science and Technology with ERDF funds under grants TIN2016-76956-C3-3-R.

REFERENCES

[1] *Computer Science Curricula 2013. Curriculum Guidelines for Undergraduate Degree Programs in Computer Science*. The Joint Task Force on Computing Curricula Association for Computing Machinery (ACM) IEEE Computer Society, 2013.

[2] IEEE Computer Society, *Guide to the Software Engineering Body of Knowledge SWEBOK. A Project of the IEEE Computer Society*. 2014.

[3] A. M. Moreno, M. I. Sánchez-Segura, F. Medina-Domínguez, L. Peters, and J. Araujo, "Enriching traditional software engineering curricula with software project management knowledge," *Proc. - Int. Conf. Softw. Eng.*, pp. 404–411, 2016.

[4] L. Peters and A. M. Moreno, "Educating Software Engineering Managers - Revisited What Software Project Managers Need to Know Today," *Proc. - Int. Conf. Softw. Eng.*, vol. 2, pp. 353–359, 2015.

[5] Project Management Institute, Ed., *A Guide to the Project Management Body of Knowledge (PMBOK® Guide)*, Sixth Edit. 2017.

[6] Project Management Institute, Ed., *Software Extension to the PMBOK® Guide*, Fifth Edit. 2013.

[7] M. Kuhrmann and J. Münch, "When teams go crazy: An environment to

experience group dynamics in software project management courses," *Proc. - Int. Conf. Softw. Eng.*, pp. 412–421, 2016.

[8] Z. S. H. Abad, M. Bano, and D. Zowghi, "How much authenticity can be achieved in software engineering project based courses?," *Proc. - 2019 IEEE/ACM 41st Int. Conf. Softw. Eng. Softw. Eng. Educ. Training, ICSE-SEET 2019*, pp. 208–219, 2019.

[9] C. Peraire, "Dual-Track Agile in Software Engineering Education," *2019 IEEE/ACM 41st Int. Conf. Softw. Eng. Softw. Eng. Educ. Train.*, pp. 38–49, 2019.

[10] H. Longjun, L. Dai, G. Bin, and L. Gang, "Project -driven teaching model for software project management course," *Proc. - Int. Conf. Comput. Sci. Softw. Eng. CSSE 2008*, vol. 5, pp. 503–506, 2008.

[11] P. Ralph, "Re-imagining a course in software project management," *Proc. - Int. Conf. Softw. Eng.*, pp. 116–125, 2018.

[12] G. Bavota, A. De Lucia, F. Fasano, R. Oliveto, and C. Zottoli, "Teaching software engineering and software project management: An integrated and practical approach," *Proc. - Int. Conf. Softw. Eng.*, pp. 1155–1164, 2012.

[13] R. Chatley and T. Field, "Lean learning - Applying lean techniques to improve software engineering education," *Proc. - 2017 IEEE/ACM 39th Int. Conf. Softw. Eng. Softw. Eng. Educ. Track, ICSE-SEET 2017*, pp. 117–126, 2017.

[14] M. Rambocas and M. K. S. Sastry, "Teaching Business Management to Engineers: The Impact of Interactive Lectures," *IEEE Trans. Educ.*, vol. 60, no. 3, pp. 212–220, 2017.

[15] I. Cabrera, J. Villalon, and J. Chavez, "Blending Communities and Team-Based Learning in a Programming Course," *IEEE Trans. Educ.*, vol. 60, no. 4, pp. 288–295, 2017.

[16] F. Martinez-Rodrigo, L. C. Herrero-De Lucas, S. De Pablo, and A. B. Rey-Boue, "Using PBL to Improve Educational Outcomes and Student Satisfaction in the Teaching of DC/DC and DC/AC Converters," *IEEE Trans. Educ.*, vol. 60, no. 3, pp. 229–237, 2017.

[17] I. Calvo, I. Cabanes, J. Quesada, and O. Barambones, "A Multidisciplinary PBL Approach for Teaching Industrial Informatics and Robotics in Engineering," *IEEE Trans. Educ.*, vol. 61, no. 1, pp. 21–28, 2018.

[18] M. Marques, S. F. Ochoa, M. C. Bastarrica, and F. J. Gutierrez, "Enhancing the Student Learning Experience in Software Engineering Project Courses," *IEEE Trans. Educ.*, vol. 61, no. 1, pp. 63–73, 2018.

[19] C. Y. Chen, Y. C. Hong, and P. C. Chen, "Effects of the meetings-flow approach on quality teamwork in the training of software capstone projects," *IEEE Trans. Educ.*, vol. 57, no. 3, pp. 201–208, 2014.

[20] G. Rodriguez, A. Soria, and M. Campo, "Measuring the Impact of Agile Coaching on Students' Performance," *IEEE Trans. Educ.*, vol. 59, no. 3, pp. 202–209, 2016.

[21] R. Hoda, M. A. Babar, Y. Shastri, and H. Yaqoob, "Socio-Cultural Challenges in Global Software Engineering Education," *IEEE Trans. Educ.*, vol. 60, no. 3, pp. 173–182, 2017.

[22] R. Q. Gonçalves, C. G. Von Wangenheim, J. C. R. Hauck, and G. Petri, "An instructional feedback technique for teaching Project Management tools aligned with PMBOK," *Informatics Educ.*, vol. 16, no. 2, pp. 197–224, 2017.

[23] J. L. Fernández-Alemán, J. M. Carrillo-De-Gea, J. Vidal Meca, J. Nicolás Ros, A. Toval, and A. Idri, "Effects of Using Requirements Catalogs on Effectiveness and Productivity of Requirements Specification in a Software Project Management Course," *IEEE Trans. Educ.*, vol. 59, no. 2, pp. 105–118, 2016.

[24] R. O. Chaves, C. G. Von Wangenheim, J. C. C. Furtado, S. R. B. Oliveira, A. Santos, and E. L. Favero, "Experimental evaluation of a serious game for teaching software process modeling," *IEEE Trans. Educ.*, vol. 58, no. 4, pp. 289–296, 2015.

[25] J. Söderlund, "Theoretical Foundations of Project Management: Suggestions for a Pluralistic Understanding," in *The Oxford Handbook of Project Management*, P. W. G. Morris, J. Pinto, and J. Söderlund, Eds. Oxford University Press, 2011.

[26] H. J. Smyth and P. W. G. Morris, "An epistemological evaluation of research into projects and their management: Methodological issues," *Int. J. Proj. Manag.*, vol. 25, no. 4, pp. 423–436, 2007.

[27] P. L. Lalonde, M. Bourgault, and A. Findeli, "An empirical investigation of the project situation: PM practice as an inquiry process," *Int. J. Proj. Manag.*, vol. 30, no. 4, pp. 418–431, 2012.

[28] M. Hällgren and A. Söderholm, "Projects-as-Practice: New Approach, New Insights," in *The Oxford Handbook of Project Management*, P. W. G. Morris, J. Pinto, and J. Söderlund, Eds. Oxford University Press, 2011.

[29] T. O'Leary and T. Williams, "Managing the social trajectory: A practice perspective on project management," *IEEE Trans. Eng. Manag.*, vol. 60, no. 3, pp. 566–580, 2013.

[30] P. W. G. Morris, J. Pinto, and J. Söderlund, "Introduction: Towards the Third Wave of Project Management," in *The Oxford Handbook of Project Management*, P. W. G. Morris, J. Pinto, and J. Söderlund, Eds. Oxford University Press, 2011.

[31] J. J. Lee, S. Y. Kang, and J. H. Heo, "Business and management competency of engineers: Curriculum and assessment," *2010 IEEE Educ. Eng. Conf. EDUCON 2010*, pp. 295–302, 2010.

[32] A. K. Garg and K. K. Rajah, "Management education for engineers: Research issues in the context of south africa," *IEEE Eng. Manag. Rev.*, vol. 40, no. 2, pp. 3–8, 2012.

[33] S. H. Pulko and S. Parikh, "Teaching 'Soft' Skills to Engineers," *Int. J. Electr. Eng. Educ.*, vol. 40, no. 4, pp. 243–254, Oct. 2003.

[34] B. G. Glaser and A. Strauss, *The Discovery of Grounded Theory Strategies for Qualitative research*. New Jersey, USA: Aldine Transaction, 1967.

[35] B. G. Glaser and J. Holton, "Remodeling Grounded Theory," *Forum Qual. Soc. Res.*, vol. 5, no. 2, 2004.

[36] B. G. Glaser, "Conceptualization: On Theory and Theorizing Using Grounded Theory," *Int. J. Qual. Methods*, vol. 1, no. 2, pp. 23–38, 2002.

[37] M. Birks and J. Mills, *Grounded Theory: a practical guide*. Thousand Oaks, California: SAGE Publications Limited, 2012.

[38] A. S. Lee and R. L. Baskerville, "Generalizing Generalizability in Information Systems Research," *Inf. Syst. Res.*, vol. 14, no. 3, pp. 221–243, Sep. 2003.

[39] B. G. Glaser, *Theoretical sensitivity: advances in the methodology of grounded theory*. Mill Valley, CA: Sociology Press, 1978.

[40] G. Pinto, C. Ferreira, C. Souza, I. Steinmacher, and P. Meirelles, "Training Software Engineers Using Open-Source Software: The Students' Perspective," no. December 2018, pp. 147–157, 2019.

[41] R. L. Quintanilla Portugal, P. Engiel, J. Pivatelli, and J. C. S. Do Prado Leite, "Facing the challenges of teaching requirements engineering," *Proc. - Int. Conf. Softw. Eng.*, pp. 461–470, 2016.

[42] M. Paasivaara, D. Vodă, V. T. Heikkilä, J. Vanhanen, and C. Lassenius, "How does participating in a capstone project with industrial customers affect student attitudes?," *Proc. - Int. Conf. Softw. Eng.*, pp. 49–57, 2018.

[43] V. T. Heikkilä, M. Paasivaara, and C. Lassenius, "Teaching university students Kanban with a collaborative board game," *Proc. - Int. Conf. Softw. Eng.*, pp. 471–480, 2016.

[44] C. Lacave, A. I. Molina, and M. A. Redondo, "A Preliminary Instrument for Measuring Students' Subjective Perceptions of Difficulties in Learning Recursion," *IEEE Trans. Educ.*, vol. 61, no. 2, pp. 119–126, 2018.