computation

MDPI

Article

Aligning Software Engineering Teaching Strategies and Practices with Industrial Needs

José Metrôlho [1,2,*], Fernando Ribeiro [1,2], Paula Graça [3,4,5], Ana Mourato [2], David Figueiredo [2] and Hugo Vilarinho [5]

[1] R&D Unit in Digital Services, Applications and Content, Polytechnic Institute of Castelo Branco, 6000-767 Castelo Branco, Portugal; fribeiro@ipcb.pt
[2] School of Technology, Polytechnic Institute of Castelo Branco, 6000-767 Castelo Branco, Portugal; amourato@ipcbcampus.pt (A.M.); david.figueiredo@ipcbcampus.pt (D.F.)
[3] Instituto Superior de Engenharia de Lisboa, 1959-007 Lisbon, Portugal; paula.graca@isel.pt
[4] Instituto Politécnico de Lisboa, 1549-020 Lisbon, Portugal
[5] Do iT Lean, 2415-371 Leiria, Portugal; hugo.vilarinho@doitlean.com
* Correspondence: metrolho@ipcb.pt; Tel.: +351-273339300

Abstract: Several approaches have been proposed to reduce the gap between software engineering education and the needs and practices of the software industry. Many of them aim to promote a more active learning attitude in students and provide them with more realistic experiences, thus recreating industry software development environments and collaborative development and, in some cases, with the involvement of companies mainly acting as potential customers. Since many degree courses typically offer separate subjects to teach requirements engineering, analysis and design, coding, or validation, the integration of all these phases normally necessitates experience in a project context and is usually carried out in a final year project. The approach described in this article benefits from the close involvement of a software house company which goes beyond the common involvement of a potential customer. Students are integrated into distributed teams comprising students, teachers and IT professionals. Teams follow the agile Scrum methodology and use the OutSystems low-code development platform providing students with the experience of an almost real scenario. The results show that this approach complements the knowledge and practice acquired in course subjects, develops the students' technical and non-technical skills, such as commitment, teamwork, and communication, and initiates them in the methodologies and development strategies used in these companies. The feedback from the teachers involved, software companies and students was very positive.

Keywords: agile software development; computer science education; industrial needs; product development; software engineering; software development management

Citation: Metrôlho, J.; Ribeiro, F.; Graça, P.; Mourato, A.; Figueiredo, D.; Vilarinho, H. Aligning Software Engineering Teaching Strategies and Practices with Industrial Needs. *Computation* **2022**, *10*, 129. https://doi.org/10.3390/computation10080129

Academic Editor: Demos T. Tsahalis

Received: 20 June 2022
Accepted: 22 July 2022
Published: 27 July 2022

1. Introduction

The industry–academia relationship has always been something that both sides have regarded well. This is a relationship for which several advantages have been identified over time [1]. To exploit these advantages, several strategies and models have been followed in academia involving companies. As the latter is the leading destination of students graduating from higher education institutions (HEIs), it is expected that it is at this level of education that the win-win advantages of such a partnership are most explored (e.g., [2]).

Preparing students to be part of high-performance teams in the labor market is a demanding task for HEI. In software engineering (SE) teaching, HEIs should strive to improve students' skills to develop projects with higher standards and actively contribute to meeting the expected cost and time of industry projects. Behavioral, technical, and organizational skills are essential in achieving excellence in high-performance software development teams [3]. Training students to develop these skills to work proactively in

high-performance teams is a crucial aspect of making them competitive in the labor market. Active learning strategies are one of the present trends to foster this demanding preparation. These aim to promote high levels of interaction and stimulate students to perform not only basic cognitive tasks, such as code reading and writing but also foster and develop other relevant skills, such as efficient communication, coordination, teamwork, team diversity, leadership, and team cohesion and motivation.

Traditional teaching, where the student simply listens to what the teacher presents in class and memorizes concepts and/or methods or solves isolated exercises, has long been insufficient. In the case of engineering, and especially in the case of computer science, where the subjects are so diverse (many tools, paradigms, techniques, etc.) and evolve very quickly, these traditional methods do not address the demands of the companies or the curiosity of the students, who are increasingly thirsty to learn new technologies and be up to date. Furthermore, in engineering and computing, it is increasingly desirable to have a practical and holistic learning approach, beyond theoretical concepts, where the student learns to develop the knowledge and skills of not only how to implement a specific component of a system but also to develop complete systems. Another increasingly important aspect is that students will quickly gain experience in methods of planning, teamwork, self-organization, documentation, and testing of the artefacts they produce. For these aspects, the industry–academia relationship and appropriate methods can also improve the students' learning and motivation.

In computer science courses, this holistic approach can be effectively learned in SE courses [4] or the final year project (project/internship). Competence in planning, requirements analysis, design, development, and testing of software projects highlights several issues that cannot be addressed within a single subject.

This paper aims to describe and share an approach that resulted in gains for both the academy and the company, but above all for the students and even for third parties (a social sector entity). Before describing the approach, it will be contextualized by referring to some important teaching aspects and other approaches that align to strengthen this industry–academia relationship. The approach described in this article has been implemented in final course subjects throughout two academic semesters. Moreover, in this case, the best practices of SE were implemented within a practical case to reinforce the students' acquired knowledge and experience in this field to complement their previously acquired knowledge in their course subjects. For this improvement, the implemented approach and broad cooperation are fundamental, providing an experience with an almost real scenario.

To summarize, the main contributions of this research are as follows:

- Proposal of a project-based learning (PBL) framework to reduce the gap between SE education and practice in the software industries.
- Sharing the experience of implementing this approach where students were integrated into mixed (students, teachers and IT professionals from a software house) distributed teams using an agile methodology and a low-code development platform (LCDP).
- Discussion of the perceived outcomes and benefits of this approach regarding the company's position and the perceived benefits of this approach on students' technical and non-technical skills.

The remainder of this article is organized as follows. Section 2 provides a background on teaching and learning strategies in SE and concludes with problem identification. Section 3 presents the main guidelines of the proposed approach and identifies the research questions. Section 4 presents the industry perspective related to university–industry collaboration. Section 5 presents a case study to illustrate the application of the proposed approach and the results achieved. Section 6 presents the results obtained and describes the details of the survey and the results for each research question. Finally, Section 7 presents the conclusions and directions for future work.

2. Background and Problem Identification

Over the years, different teaching methods have been implemented, particularly in the teaching of engineering and technology subjects. Teaching in HEI courses has also started to include, as a priority, the preparation of their students for employability and, besides the technical component, soft skills (problem-solving, communication, teamwork, etc.) have also started to be emphasized. On the other hand, in Europe, engineering degree courses are typically held in six semesters, which highlights the need to work on some aspects to compensate for their natural immaturity and the short duration of the courses. In addition, many degree courses typically offer separate courses to teach requirements engineering, analysis and design, coding, or validation. The integration of all these phases normally requires experience in a project context. In this context, many approaches have been proposed to teach and learn engineering and technology subjects and, more specifically, SE subjects. Usually, the proposed approaches differ from each other and sometimes have specific goals, but there is a general goal that has been common to many approaches over time: to improve the training of students by allowing them to develop their skills so that they can face the real needs of companies in this area.

2.1. Teaching and Learning Software Engineering

This section presents a review of strategies and approaches used in SE teaching. This review aims to obtain an overview of the most used strategies and the research interest that this area has aroused.

An analysis of the results obtained from a search in the Scopus database about works that address strategies for teaching and learning SE allows us to obtain an overview of the interest that this area has aroused, as well as the characteristics of the different approaches.

To perform this analysis, a set of search terms related to SE teaching and learning were identified. After some initial experiments and considering an initial analysis of some literature studies, three groups of terms were identified. First, terms related to SE should be included since they represent the subject considered in the study. Second, terms related to teaching. Thirdly, terms related to strategies or approaches. The complete string for the search was as follows:

("software engineering") AND (teach*) AND (strateg* OR approach*)

The search was performed on the field's article title and keywords and only considered computer science and engineering subjects. Only works published until 2021 were considered. The search was performed in November 2021 and resulted in 370 studies. The 370 studies included in the analysis were published between 1988 and 2021 (see Figure 1). The graph shows the growing interest that has taken place over the years, mainly from 2008 onwards. In the analysis, it is necessary to consider that a significant part of the 2021 publications may not yet be available in the database.

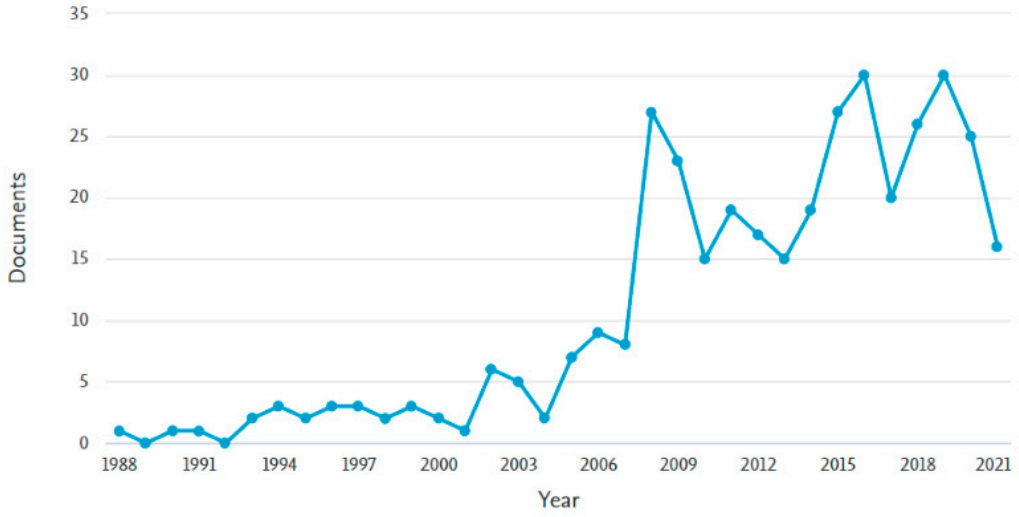

Figure 1. Documents by year of publication.

Despite the specificities of each country and the different approaches they may have concerning SE teaching, research in this area is common in many countries. The results grouped by country/territory to which the authors belong can be seen in Figure 2.

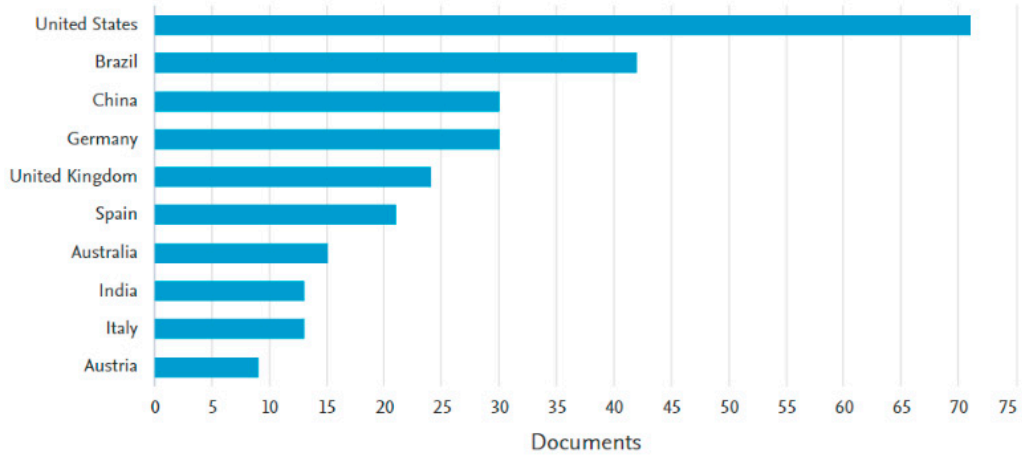

Figure 2. Documents by country or territory (10 most representative).

An analysis of the titles and abstracts of the selected articles makes it possible to identify some of the most discussed terms. After excluding the keywords used in the search, the keywords of the works were analyzed to obtain a characterization of their frequency. Figure 3 shows the tag cloud that was obtained based on these results.

Figure 3. Tag cloud of the most representative keywords.

Many of the keywords are generic. However, several seem to be related to approaches taken in teaching SE. For example, project, project-based, gamification or collaborative.

A more detailed analysis of the search results allowed us to characterize them according to some of the strategies often associated with teaching SE. Searching within the 370 records, using terms generally associated with some strategies used in teaching SE, the following results were obtained (see Figure 4):

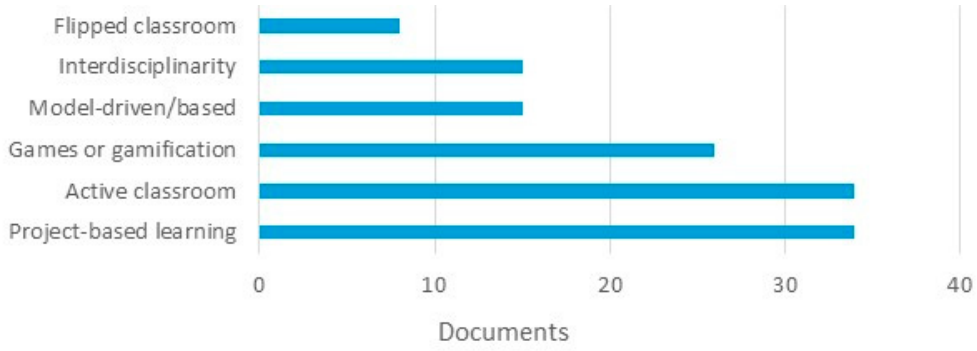

Figure 4. Some strategies for SE education.

The search performed was not exhaustive but is intended to illustrate the interest of some strategies that have been used. Furthermore, the strategies can be used independently or combined. For example, gamification strategies can be used when using PBL.

2.2. Teaching Strategies

Teaching SE-related courses to undergraduate students is a challenging task. It is a very dynamic knowledge area, and changes in programming paradigms and software process development methodologies are frequent. As a result, it is difficult to predict what knowledge or methodologies will be of use to students after joining their professional activity. This has led to the emergence of new strategies to foster the assimilation of knowledge, approximation with what is done in software companies and student motivation [4]. It has also highlighted the need to use strategies that reduce the gap between how teachers transfer knowledge to students and how students acquire it, leading to more student-centered approaches and active learning strategies [5].

PBL flipped classroom and gamification are strategies that are frequently used to teach SE.

A PBL strategy would enable students to contextualize new knowledge, learn to communicate in context, solve problems in context and practice a design process in context,

leading to professionally prepared students [6]. It engages students in a collaborative process focused on a specific project. Many of the proposed PBL approaches have involved companies, usually acting as potential clients, in the teaching process to foster approximation with what is done in software companies. This is the case of the study presented in [7] that describes a teaching approach for a machine learning course offered to various computer science Master's programs. They describe a PBL with differentiated projects provided by industrial partners that address diverse study programs.

The adoption of games or gamification strategies in SE education has been increasing. They are used to improve the commitment and performance of students since they allow them to perform their tasks following the game mechanics, thus making the task more attractive. A study in [8] investigates the use of games and game elements in SE education. They found that games are primarily used to cover 'Software Process' and 'Project Management', and the most used game elements are points, quizzes, and challenges. In [9], an approach that couples PBL and game-based learning approaches in teaching SE courses is described. This study found that students' interest and engagement were improved with gamification strategies.

Model-driven approaches are also used for SE teaching. The main idea behind model-driven approaches for SE is that software can be built starting with an abstract representation, which is successively refined until an implementation is reached. These approaches are model-centric (unlike other code-centric approaches) and emphasize the use of models at all levels of the software development lifecycle. A model-driven approach to SE education and its impacts on teaching SE is discussed in [10].

Flipped classroom approaches are also used to teach SE. An experience teaching SE using a flipped classroom approach is described in [11]. The goal was to increase student learning by providing more time for active learning in class. They concluded that students' academic success can be improved by introducing a flipped classroom.

Other approaches, such as the one described in [12], have been implemented to foster a better preparation of students by assigning roles to students so that they gain experience with the importance of those roles for the development process of a software project. In these approaches, teachers play the client role and provide support as external consultants, applying theoretical concepts in practical activities to prepare them for the roles they will have in the IT industry. In the approach described in this article, the IT industry collaborators participate directly and actively in teaching both technical and soft skills. This participation of IT industry collaborators allows the students to get an overview of the different roles in an almost-real context and gain experience with the work methodologies and soft skills considered necessary, by companies, for professional activity. This active participation of the IT industry collaborators allows direct transmission of knowledge and good practices, followed by professionals on the industrial side, during the students' academic preparation. It allows students to acquire know-how from all the team members involved (academic and IT industry) with whom they work during the project timeline. There are also other studies that approach the teaching of agile methodologies in SE or present successful cases of its application. In [13], the authors report a PBL implementation for the initial education on the Scrum framework. In this case, the study addresses an academic scenario without the involvement of the industry, and they do not follow many Scrum recommendations (e.g., sprints last 1 day, daily meetings are held every hour). A multiple case study on agile practices in distributed Scrum projects was described in [14]. In this study, they interviewed 19 project team members and described how Scrum practices, such as daily scrums, backlogs, and sprints, were adopted for distributed development. The case studies presented allowed them to identify challenges and benefits as well as lessons learned from applying Scrum in distributed settings. Although they describe methodologies and good practices used in an industrial environment, these case studies do not involve academic–industry collaboration or student training.

In fact, during the last decades, SE education has been continually adapting to keep students updated with the software technologies, processes, and practices that industries

require. The different approaches proposed and studied over the years confirm this. However, despite the diversity of strategies, there seem to be some trends. These include strategies that are more focused on students, demanding a more active attitude from them, and with the involvement of companies, mainly acting as clients. A study [15] that investigated from the literature the extent to which SE education addressed major SE trends in the academic setting revealed that agile software development is the major trend. In this study, the authors also pointed out some guidelines to face the difficulties in the adoption of SE trends, namely (i) the active involvement of industry stakeholders (mentors, clients, and customers), to recreate agile industry practices and preparing students with real challenges, and (ii) partnerships with enterprises from the software industry to help in the coordination and technical set up of the course.

2.3. Problem Identification

The existence of a gap between SE education and the needs and practice in the software industry is well known and has been referred to in several studies (e.g., [4,15,16]). To reduce this gap, it is necessary to provide students with real challenges, more realistic experiences, recreating industry software development environments and collaborative development. As mentioned before, several approaches have been followed to reduce this gap. Many of them follow PBL approaches or gamification strategies. The involvement of companies has also been increasingly frequent, mainly acting as potential customers for the product to be developed.

There is also another great challenge that arises. There has been a growing interest and need for scenarios involving distributed teams. Many software companies operate in a distributed environment to counter the concerns related to project development costs, acquire high-skilled resources, and increase production. With the COVID-19 pandemic, this paradigm has become a common practice and recently, we have seen a growing interest amongst workers, mainly in the technological areas, to move from large cities to low-density populated areas (e.g., rural areas, small–medium-sized cities) looking for places with a low cost of living and healthier lifestyles.

Thus, despite the positive results obtained by several of the previously identified strategies, teaching and learning strategies can still benefit from a significant and closer involvement of companies, especially software houses, in the development process. This will allow students to integrate into the distributed teams of companies, develop their non-technical skills and immerse themselves in the methodologies and development strategies used in these companies. However, this approach requires a great deal of involvement and availability from companies.

3. The Approach and the Research Questions

The courses of higher education, namely licentiate degrees, take place in polytechnic institutes for three years (six semesters) and, in the last year, there are internships or project subjects. During this period, it is intended that the students consolidate the knowledge acquired in the other subjects of the respective course and acquire new knowledge (technical and behavioral), in addition to that learned so far. Given the youth of these undergraduate students, this antechamber of preparation for the labor market can be especially useful in providing them with valuable skills for the transition from academia to industry. Some approaches in higher education schools follow an internal scope (without involving companies) with a major focus on research and improving the students' self-learning skills. On the other hand, other approaches, such as the approach presented here, argue that for these students, it is particularly important that this period is invested in improving the students' skills (technical, behavioral, planning, team integration, interaction with others, global vision of a project, etc.) for what companies require at the moment. For this, it is crucial to involve companies not only as recipients of products/prototypes they request (quite common) but as they are an active and present part of the students' teaching process. Besides resulting in a high-quality product for the company, this participation should also

allow the student to acquire experience with development processes used in a company context and an accurate notion of what should be the best practices and the artefacts and tools to use to build a professional software project from scratch. To achieve these goals, it is vital that there is good availability and continuous collaboration between the staff of the company, the staff (teachers) of the teaching institution, and the students (throughout the project, in this case, the fifth and sixth semesters of the course).

The final year project is often carried out simultaneously with other subjects, which the student must attend, which makes it difficult for students to carry out the internship or project work in person at a software company.

With the proposed approach it is intended to reconcile these aspects while, at the same time, providing students with experience in:

- Distributed teams: students are integrated into teams made up of elements that work in different geographical locations.
- Multidisciplinary teams: teams are made up of elements with different characteristics and functions. The teams include students, teachers and professionals from the software house involved.
- The real environment in a software house: the environment in which the students are integrated, including tools, technologies, and work methodologies, is very close to the real environment of the software house involved.
- Methodologies and technologies used in these companies: students are integrated into teams and will use the same tools and methodologies used in the real activity of the company involved.

Following this approach and to assess its impact on some skills of the students, it is intended to answer the following research questions:

- RQ 1—Does the development of software applications proposed by external entities (and not just an academic project) contribute to greater student motivation and commitment?
- RQ 2—Does the involvement of software houses during the software development process promote a greater student commitment?
- RQ 3—What are the main benefits identified by students from their integration into distributed teams?
- RQ 4—Are Scrum and OutSystems LCDP fit for distributed teams?
- RQ 5—To what extent did the integration of students into distributed teams contribute to their communication practices, team collaboration, task allocation and distribution, and usage of collaboration tools?

4. University–Industry Collaboration—Industry Point of View

University–industry collaborations have received increased attention in management and research practices. The need for innovation in today's business environment intensifies this trend [17]. The transfer of knowledge and technology between academia and industry stimulates innovation by combining heterogeneous partners and mainly heterogeneous knowledge [18]. In particular, companies can benefit from qualified local human resources, researchers, and students to obtain technological and business benefits derived from distinctive innovation [19]. Moreover, a firm's decision to collaborate with universities for innovation is influenced by the geographical proximity and quality [20], as well as the willingness of institutions to collaborate. In short, cooperation between universities–industry is seen as a source of growth because it encourages knowledge transfer, boosting innovation and the performance of companies [21].

Regarding the company's point of view, they are concerned with the outcomes and benefits. Based on the literature review [22], university–industry collaboration allows the selection of tangible and intangible goals, which allow us to assess the collaboration. The intangible goals described in Table 1 focus on creativity and innovation anchored in state of the art and the alignment of the students' training with the company's project development methodology and best practices. Table 2 describes the tangible goals that can be further con-

verted into key performance indicators (KPIs) to objectively assess the collaboration benefits between the company and the polytechnics/universities. KPIs may be expressed in terms of the number of recruited and hired students, trainers (company employees) involved, contribution to the knowledge base and technical knowledge dissemination sessions.

Table 1. Intangible goals for the collaboration between industry–polytechnics/universities.

Goal	Description
Fostering creativity and innovation	Encourage employees to obtain ideas to propose projects in the polytechnics/universities with creativity and innovation.
Alignment with the state of the art	Identify and include in students' projects current polytechnic/university research that may be useful for designing and developing innovative projects that can result in new features or potential products for customers.
Boosting employees' technical skills	Upgrade the expertise of employees as trainers in the graduation projects.
Effective training for future hiring	Train the students in accordance with the company's project development methodology and best practices.

Table 2. Tangible goals for the industry-polytechnic/university collaboration.

Goal	Description
Recruitment	Attract qualified students to join the company through final graduation projects or internships.
Employee engagement	Engage employees to apply to become trainers by motivating them with professional benefits and work satisfaction.
Company disclosure	Submission of proposals to the polytechnics/universities for final graduation projects and internships.
OutSystems disclosure	Train the students using OutSystems guided paths.
Company reputation	Co-authorship of scientific publications in refereed journals or conferences. Companies use joint publications as a public relations tool to add to their prestige.
Company knowledge	Build a knowledge base with the results of the graduation or internship projects.
Knowledge dissemination	Realization of internal technical sessions to disseminate the graduation or internship projects.

Closer involvement of the industry stakeholders in the SE education process can benefit all parties involved. However, they need to dedicate time, effort, and engagement to this collaboration. Some of the tangible and intangible goals identified by the company involved in the process (Tables 1 and 2) are also recognized in other similar works. For instance, the approach described in [23] involves industry in an undergraduate SE project course. In this approach, the industry plays an active role in supervising software development. The benefits identified for the companies are in line with the previously mentioned ones: opportunity to recruit the best graduates; branding of their company and products with undergraduates and other sponsors; relationships with faculty, and networking with students and other corporate sponsors. Another strategy is to develop additional training programs (e.g., [24]) before the hiring process, which also helps companies screen viable candidates. These training programs aim to screen new graduates, introduce them to core skills relevant to the organization and industry, and assess their attitudes toward mastering those skills before the hiring process begins. Another significant, recognized benefit is elevating the company's image among university students.

5. A Case Study

As mentioned before, there are already cases that successfully followed this strategy in implementing end-of-course projects involving final-year students. To exemplify some crucial aspects of implementing this approach, one of the cases will be briefly presented here. In this description are shared aspects about the process, the project and the product that was followed/implemented/achieved over two semesters. The following sections present some aspects of this representative case that are common within the approach that is being followed with other projects that take place within the scope of this collaboration.

5.1. The Academy–Industry Commitment

Long before starting the development of the project, there was an interaction between the teachers of the higher education institution and the company representative to define the project themes and goals. In other words, providing a set of topics and initial requirements that are presented to all students and that, after an assignment process carried out by the coordinator of the respective course, are allocated to specific students. When defining the themes and general requirements, the professors and the company representative (later with the role of product owner) ensure the subsequent availability of development tools, the technical feasibility of the project, technical and pedagogical support conditions, as well as the objectives that must be achieved in each of the two semesters. That is, guaranteeing that the amount of work justifies its inclusion over two semesters. This is an important aspect because to approach the job market, students must be challenged and led to producing good volumes of work in a methodical, systematic, and well-planned way. In other words, following good SE practices, achieving significant levels of performance and productivity and, in the end, surpassing ambitious goals with quality.

Thus, in this specific case, the theme defined was the implementation, from scratch, of a web platform and mobile application to be developed using the OutSystems LCDP (something that students did not learn during their course but that the market values) to support voluntary institutions. This theme arose when the COVID-19 pandemic was a current and universal theme and when this type of institution needed the support of technologies for communication and management among their staff (namely volunteers). It was also established that the web platform must be implemented in the first semester and the mobile application in the second semester.

Another aspect that was immediately guaranteed was that two people from the company's staff would be allocated to support the work with the educational institution doing the same thing. The theme proposed to all students from the outset contemplated a current theme, a new technology (in addition to those taught in the course) and demand in the job market—additionally, a team of four people to guarantee the technical and pedagogical support needed by the students. In addition to these guarantees, the company representative and the teachers also jointly defined other tools necessary to support communication and the project's development. The development process was also defined, in this case, Scrum, as it is the most common one in the company's projects and adapts to the type of product understood (since the general requirements were initially defined, but the concrete requirements needed further analysis of the problem to solve and subsequent definition, with the students involved). Moreover, as it was known that the development platform was new to the students, a learning plan was prepared for them, with the support of the company's staff, so that they could have the necessary expertise at hand for the implementation stage and do so successfully.

5.2. The Team

The two students who chose this theme became part of a team of six members (two professors, two company employees and the students), which is Scrum compatible (for cases of only one student carrying out the project, the team has five members that guarantee the same conditions). In this team, the two students, together with one of the company's employees, played the role of developers. The other company employee played the product-

owner role (involved with the two professors from the beginning in defining the project's objectives). The remaining members, both from the higher education institution and teaching faculty, acted in the role of Scrum Master and student supervisors. In the more technical parts (related to software development in OutSystems), they were followed very closely and frequently (daily scrums) by one of the company's members, while the teachers and product owner monitored components involving documentation, planning, and requirements. Each sprint planning, each sprint review, and each sprint retrospective involved all members. The sprints were defined for periods of three weeks. The daily sprints were sometimes not daily, but they always took place at least twice a week, involving the students and the members of the company in charge of the technical follow-up. This permanent interaction throughout the school year led to a strong team atmosphere between the six members. The open, demanding posture of the students contributed to this, which is considered essential and is one of the objectives of this approach. Furthermore, an important fact is that team members were geographically distributed. In this case, the members of the company were in different places from the members of the academy and students. However, they did not stop working closely with each other, using new communication technologies.

5.3. The Product

As mentioned previously, this project theme (as well as others that have been proposed) should have a volume of work (i.e., complexity), be useful (possible to be tested by third parties) and be interesting to motivate students and have goals and objectives for an academic year (two semesters). In this case, everything was guaranteed from the first moment, as mentioned in the previous sections, because in addition to a web application (Backoffice), the availability of a mobile application was also defined as a requirement. It was defined during the steps previously described in Section 5.1 that in the first semester (Project I), the Backoffice would be implemented and that in the second semester (Project II), the mobile application and the tests of the complete system (Backoffice + App) would be implemented. For students to better understand the activity of these entities, they were led to carry out a state-of-the-art survey of existing applications for this type of activity. This initial task, carried out in the first four weeks, allowed the team to know and clearly define the specific requirements (user stories) for the new application to be developed, which the product owner validated. In parallel with this key step to define the products, the students performed a learning period with the new development tool, the OutSystems LCDP, to develop the Backoffice, which was the goal for the first semester. At the beginning of the second semester, similarly, the students learned to develop a mobile environment for the development of the mobile component. For both learning stages, they had the technical support of a company's member and were guided with adequate content for solid learning, in the same way as is done when new staff are integrated into the company.

The resulting product consisted of a portal and a mobile application developed in OutSystems, according to a development process described below and resulting in project documentation that not only complied with the standards usually required in the company's projects but also considered the requirements demanded by the academic juries.

Figure 5 presents two of the several screenshots of the final product, one from the Backoffice and the other from the mobile application. The system was built in Portuguese, according to the demands of the product owner.

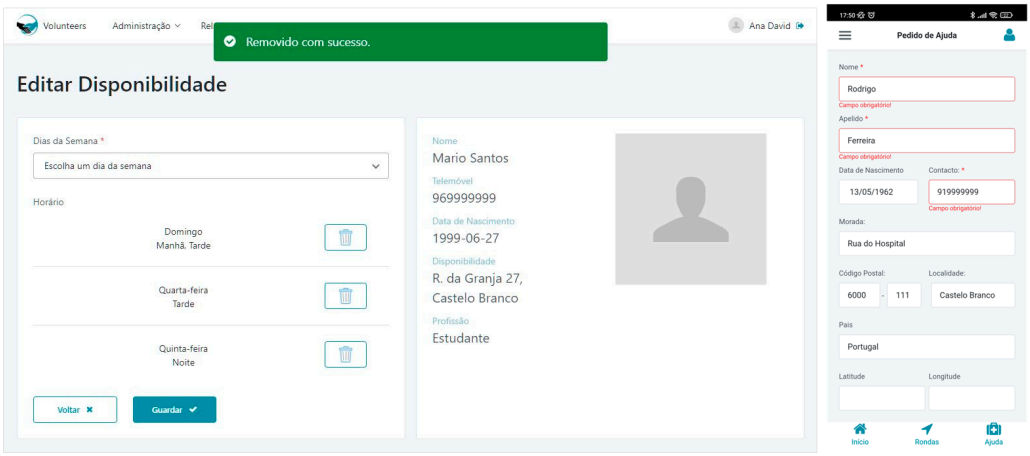

Figure 5. Backoffice (**left**) and App (**right**) interfaces.

5.4. The Process and Project

The development process adopted was very close to the one typically used in the company for business purposes. In this way, students are included in a work environment similar to what happens in a professional environment. For the development process, Scrum was followed, in which the roles of each one were as described above in Section 5.2. In defining each sprint backlog, the volume of work, the user stories assigned to each sprint, and other academic commitments (workload with exams, assignments for other subjects, etc.) of the students were also considered. Both the product backlog for each semester and the corresponding sprint backlogs resulted from a process of great participation from all the team members. However, it is up to the product owner to validate the priorities in terms of user stories for each sprint. Backlogs and user stories were recorded using the Trello tool, which enabled the monitoring and recording of work in progress, with visibility for all team members. Another important aspect of planning is that at the beginning of each semester, a timeline is defined with the dates of all sprints, including the milestones to be reached, so that there is a detailed plan of goals to be achieved. Figure 6 presents the timeline of the first semester of this project.

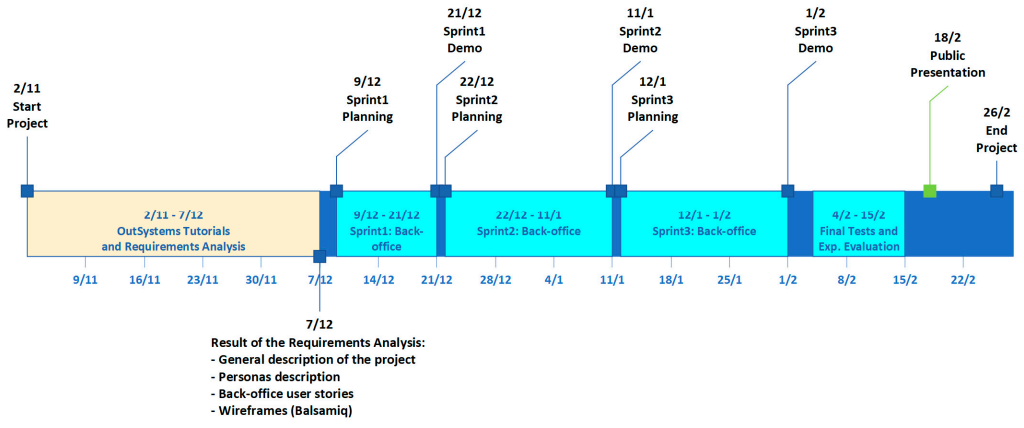

Figure 6. Timeline (first semester).

The daily Scrum meetings lasted 15 min to answer the typical questions suggested by Scrum ("What have you done since the last scrum meeting? Do you have any obstacles? What will you do before the next meeting?"). Since we were also facing a training process (student and not an experienced professional), the daily meeting times sometimes lasted longer because they served to guide the student to unlocking solutions to the identified difficulties. At the end of each sprint, there were demos, and mid-term reviews, where students presented the product improvements they have made to that date. After this

sprint review, the team also made a retrospective sprint to evaluate aspects that could be improved in the following sprints and thus optimize the team's performance and software production speed. At the end of the second semester, the developed system (web application + mobile app) was also evaluated, in the testing phase, by an external entity with voluntary activity (end-user). In other words, the product was also submitted to obtain feedback from potential end-users of the developed system, including this vital stage of the development cycle of a software application, to check that the software product satisfies or fits their intended use. In other words, collaboration and participation in this project were further extended, with the inclusion of a third entity, in this case, a potential end-user of the application/product. The feedback was positive, and some improvement suggestions were still included in the final product before the final evaluation of the work. The participation of this external entity added another player (the point of view of the final recipient of the product) that is important in this strategy of involving students in a real development environment. That is, from the initial definition of the product to be developed to the end, as shown in Figure 7, the number of stakeholders increased, resulting in a complete development process and allowing the exploration of all stages of the development cycle.

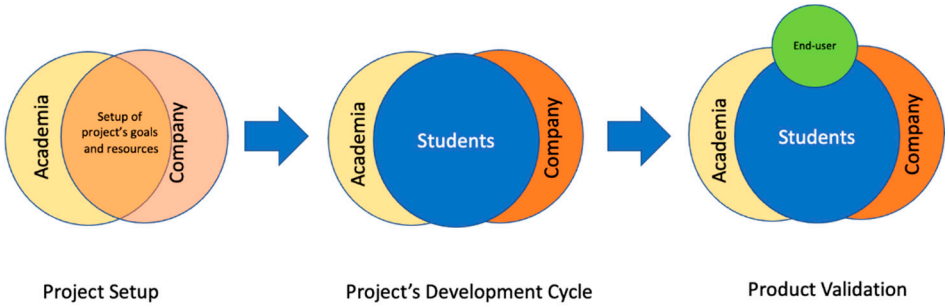

Figure 7. Collaborative evolution.

As for the project documentation, Backoffice and Mobile, the students built their reports, one per semester, not at the end but during the entire development cycle. Thus, from state of the art, the definition of user stories, training periods of new tools, tools used, application design, data models, wireframes, as well as analyses and conclusions were documented to fulfil the requirements of academic reports while emphasizing the type of technical documentation used in company projects. In other words, here, too, approaching and realizing the company's requirements while complying with academic rules (sometimes more demanding in terms of documentation than agile approaches, in practice, need).

5.5. Results and Feedback

In this case, as in others where this approach was applied, the results were very positive. The students obtained approval in the regular evaluation periods, in which they participated, presenting a complete product, also tested by external entities. The company has expressed their total satisfaction with the objectives achieved and the product quality. The feedback on the product developed by the external entity was praiseworthy, and even this interaction resulted in the opportunity to carry out another application for that institution, which is currently under development.

Due to having invested in emerging technology and working according to this methodology, the students both got jobs in a company that also develops in OutSystems, and which also follows Scrum in less than a month. The adaptation period was facilitated according to their experience. In addition, after completing the course and having started work, the students presented their end-of-course work in a lecture to students from a secondary school in the region, serving as ambassadors and motivators to young people in the area of software development.

Currently, the same approach is being applied in other projects with different themes, which are under development, involving similarly structured teams and methodologies. This case was a success story. Nevertheless, to better assess how students accept this approach, it is also essential to know and analyze the feedback of other students who developed their end-of-course projects according to the same approach. For this reason, a survey of 15 students (who followed the same approach, having this company as a common factor) was carried out, the results being presented and analyzed below. These results show that the approach is very well accepted from the students' points of view.

6. Results and Discussion

In this section, the survey results, answered by students involved in this approach, and mapped against the study's research questions, are presented and discussed. The goal was to analyze the students' perspective during the development of a final year computer engineering project following a methodology that integrated them in a mixed (students, teachers, and IT professionals from a software house) distributed team, following the Scrum methodology and using the OutSystems LCDP.

6.1. Survey

The survey included 24 questions organized into five groups, each corresponding to one of the research questions to be studied. All questions were answered using the Likert Scale (1–5). Additionally, in each group of questions, there was an optional question where respondents could include any contributions.

Before accessing and completing the survey and any data processing, the potential participants were informed that no personal data or sensitive data would be collected. They were also informed that the information collected would only be accessed and processed by those responsible for the project and that it would only be collectively disclosed in statistical indicators at scientific events and journals. Each participant was free to accept this and, only if accepted, he/she participated in the study.

The survey was answered by 15 students from 4 higher education institutions who carried out their final computer engineering project following this approach.

RQ 1—Does the development of software applications proposed by external entities (and not just an academic project) contribute to greater student motivation and commitment?

To assess this RQ, respondents were asked to answer five questions to assess whether having an external company propose the project had any influence on their attitude towards the activities to be carried out. Table 3 presents the results obtained.

Table 3. External proposed project vs. students' motivation and commitment.

Question	1	2	3	4	5	Mode	Average	Coefficient of Variation
Influence my motivation	0%	0%	7%	53%	40%	4	4.33	0.14
Influence my commitment to the tasks to be developed	0%	0%	7%	53%	40%	4	4.33	0.14
Influence my commitment to the company that proposed the project	0%	0%	0%	47%	53%	5	4.53	0.11
Improve your work habits	0%	0%	0%	47%	53%	5	4.53	0.11
Increase the time pressure to get results faster	7%	7%	40%	33%	13%	3	3.40	0.30

According to the results, it was recognized that an external company proposing the project had a very significant influence on motivation and commitment to the tasks developed and to the company that proposed the project. It was also recognized that it contributed, in a very positive way, to improving their work habits. However, it did not seem to have influenced the feeling of time pressure to obtain results faster. This

question received a greater diversity of answers as confirmed by the significant coefficient of variation (0.3).

RQ 2—Does the involvement of software houses during the software development process promote a greater student commitment?

On this RQ, students were asked five questions to assess to what extent the involvement of software houses during the software development process influenced their motivation and commitment. All the respondents were involved in distributed teams which included IT professionals from a software house. The results can be seen in Table 4.

The results show that the involvement of a software house during the software development process strongly influenced the students' motivation and commitment, either with the tasks developed or with the company. Its influence on work habits was also very significant and was recognized by students as a significant contributor to obtaining new knowledge in addition to those taught in the course. Thus, the results suggest that the involvement of software houses during the software development process contributes to increasing the students' motivation and commitment, both with the tasks and the company involved. However, it did not seem to have significantly influenced the time pressure to get results faster.

Table 4. Involvement of an external company vs. student motivation and commitment.

Question	1	2	3	4	5	Mode	Average	Coefficient of Variation
Influence my motivation	0%	0%	0%	60%	40%	4	4.40	0.11
Influence my commitment to the tasks to be developed	0%	0%	0%	67%	33%	4	4.33	0.11
Influence my commitment to the company that proposed the project	0%	0%	0%	60%	40%	4	4.40	0.11
Improve my work habits	0%	0%	0%	53%	47%	4	4.47	0.11
Increase the time pressure to get results faster	0%	7%	33%	40%	13%	4	3.47	0.30
Obtain new knowledge in addition to those taught in the course	0%	0%	0%	47%	53%	5	4.53	0.11

RQ 3—What are the main benefits identified by students from their integration into distributed teams?

In this RQ, students were asked to indicate to what extent their integration into distributed teams contributed to improving some of their skills. The results can be seen in Table 5.

Table 5. Students' perceived benefits from their integration into distributed teams.

Question	1	2	3	4	5	Mode	Average	Coefficient of Variation
Teamwork	0%	0%	7%	73%	20%	4	4.13	0.12
Communication	0%	0%	0%	73%	27%	4	4.27	0.10
Cooperation	0%	0%	0%	60%	40%	4	4.40	0.11
Critical thinking	0%	0%	13%	53%	33%	4	4.20	0.16
Responsibility	0%	0%	0%	53%	47%	4	4.47	0.11
Sharing knowledge	0%	0%	7%	40%	53%	5	4.47	0.14

In addition to providing an experience close to the current reality in the software industry, the integration of students into distributed teams seemed to influence several of their soft skills positively. The answers show that students recognized that having worked in distributed teams had a very positive influence on the soft skills evaluated (from 86% of responses at levels 4 or 5 in critical thinking to 100% of responses at levels 4 or 5 in communication, cooperation and responsibility on the 5-level Likert scale (4—positive; 5—very positive)).

RQ 4—Are Scrum and OutSystems' low-code development platforms fit for distributed teams?

To answer this research question, seven questions were asked to understand to what extent the students recognized that the agile Scrum methodology and the use of the OutSystems LCDP were adequate and contributed to the project's development. The results of these questions are presented in Table 6.

Table 6. Scrum and OutSystems in distributed teams.

Question	1	2	3	4	5	Mode	Average	Coefficient of Variation
Do you think the Scrum methodology was adequate and contributed to the better development of the work?	0%	0%	7%	33%	60%	5	4.53	0.14
How do you assess the importance and impact that the daily scrum meetings had on the planning and development of the work?	0%	0%	7%	47%	47%	4, 5	4.40	0.14
How do you assess the importance and impact that the sprint planning meetings had on the planning and development of the work?	0%	0%	7%	40%	53%	5	4.47	0.14
How was your adaptation to the OutSystems development platform?	0%	7%	27%	53%	13%	4	3.73	0.21
To what extent did the OutSystems low-code platform contribute to faster development?	0%	0%	0%	43%	57%	5	4.57	0.11
Did the joint use of the agile development methodology Scrum and the low-code development platform OutSystems contribute to the better productivity of the team?	0%	0%	0%	47%	53%	5	4.53	0.11
To what extent did the methodology used help to meet the deadlines defined in the timeline?	0%	0%	7%	47%	47%	4, 5	4.40	0.14

The results show that the students adapted well to the agile Scrum methodology. They acknowledged that the Scrum methodology was adequate and contributed to better development of the work (93% of responses at levels 4 or 5) and rated the impact of the daily scrum meetings (94% of responses at levels 4 or 5) and the sprint planning meetings (93% of responses at levels 4 or 5) as important/very important to the planning and development of the work.

It is essential to consider that 78.6% (11 in 14—1 did not respond) of respondents had not used the OutSystems LCDP before joining this project. This seems to have caused some difficulties in adapting to the OutSystems LCDP (see Figure 8) and to have been reflected in the less favorable assessment (66% of responses at levels 4 or 5) obtained in the question "How was your adaptation to OutSystems LCDP?". Despite these difficulties, at the end of the project, the students recognized that the use of the OutSystems LCDP had a significant contribution (100% of responses at levels 4 or 5) to faster development. This question even received the highest rating in this group of questions. Students also recognized that the joint use of the agile development methodology Scrum and the OutSystems LCDP

contributed very positively to better team productivity (100% of responses at levels 4 or 5) and meeting the deadlines defined in the timeline (94% of responses at levels 4 or 5).

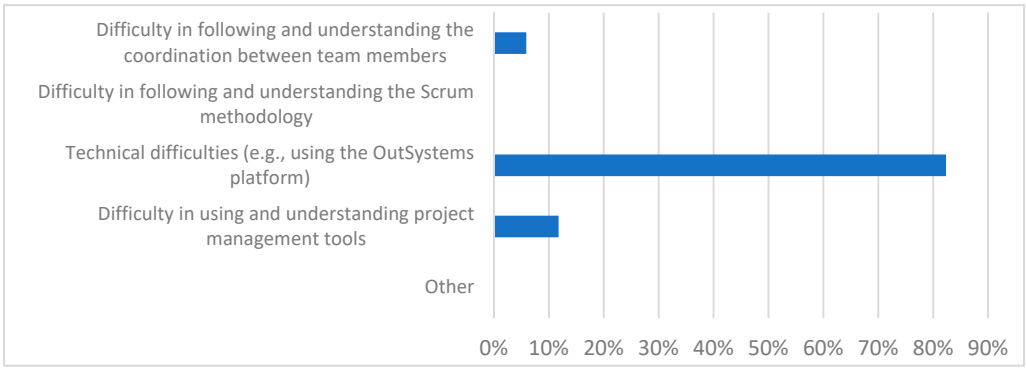

Figure 8. Most important challenges students had to face.

RQ 5—To what extent did the integration of students into distributed teams contribute to their communication practices, team collaboration, task allocation and distribution, and usage of collaboration tools?

Five questions were asked to evaluate this research question. The results are presented in Table 7.

Table 7. Scrum and OutSystems vs. distributed teams.

Question	1	2	3	4	5	Mode	Average	Coefficient of Variation
Were you always aware or were informed of changes in the project design?	0%	0%	7%	7%	87%	5	4.80	0.11
Did your team face any challenges related to communication barriers, scheduling meetings, or using different tools?	40%	27%	20%	7%	7%	1	2.13	0.56
Was the communication tool (Skype) useful and easy to use and understand (Skype/Google Meet/Other)?	0%	0%	7%	33%	60%	5	4.53	0.14
Was the project management tool (e.g., Trello/Jira) useful and easy to use and understand?	0%	13%	7%	33%	47%	5	4.13	0.25
Despite the team being geographically distributed, how do you rate the support you received during the project development?	0%	0%	13%	7%	80%	5	4.67	0.15

Of the students, 73% (11 out of 15) used Skype as a communication tool, and 53% (8 out of 15) used Google Meet (among these, 4 used both tools). These tools were rated useful and easy to use and understand (93% of responses at levels 4 or 5). Regarding project management tools, 67.7% (11 out of 15) used Jira (1 of them already had contact with the tool), and 32.3% (5 out of 15) used Trello (2 had already used it). Based on the evaluation obtained, there were no significant difficulties in using these tools, as they were rated positively (80% of responses at levels 4 or 5) for their usefulness and ease of use and understanding. The results of the previously analyzed questions were confirmed in the answers to the question, "Did your team face any challenges related to communication

barriers, scheduling meetings, using different tools?". The result of this question indicates that students rarely experienced difficulties in using these tools.

The experience of being part of a distributed team was also perceived as very positive, with no relevant difficulties being identified in terms of the support they received. Of the students, 80% rated the support they received during the project development as very positive.

Students were also asked about the most significant challenges (difficulties) they faced during the project development. Seventeen responses were obtained (some students identified more than one difficulty). The results are shown in Figure 8.

The most frequently mentioned difficulty concerned technical difficulties associated with using the OutSystems LCDP (14 times mentioned in 17 responses). This may be associated with the fact that 78.6% of respondents had no previous contact with the platform before starting this project. It also reflected some difficulties in the initial adaptation to the OutSystems LCDP, corroborating the result obtained in the question "How was your adaptation to OutSystems development platform?" in Table 6.

6.2. Threats to Validity

The number of participants (i.e., 15 students from four HEIs and one software house) might represent a possible limitation to the validity of the results of this study.

Another limitation may be related to the fact that all students used the same methodology, i.e., the agile Scrum methodology and the OutSystems low-code development platform. Although, on the one hand, this approach allowed a greater number of students to validate its use, on the other hand, it does not allow comparisons between different approaches to better identify the factors that most contributed to these results.

7. Conclusions

The present study took a different approach from previous work on this topic in trying to prepare students for the job market in terms of SE practices in the context of end-of-course projects. This approach was chosen to reduce the gap between SE education and the needs and practices in the software industry. Students were integrated into mixed (students, teachers, and IT professionals from a software house) distributed teams using the agile Scrum methodology and the OutSystems LCDP, providing students with the experience of an almost real scenario. The case study presented illustrates the work methodology and the results obtained.

The results from the survey, answered by 15 students from four HEIs who developed their projects following this methodology, allow us to confirm the five research questions identified in the study, as follows: The development of software applications proposed by external entities (and not just an academic project) contribute to greater student motivation and commitment; the involvement of software houses during the software development process promotes greater student commitment; students recognize that integration into distributed teams contributes to the development of some of their soft skills; the use of Scrum and OutSystems LCDP seems to fit this work methodology well; and the integration of students in distributed work teams contributes to improving their communication practices, team collaboration, task allocation and distribution and usage of collaboration tools.

Industry–academia collaborations, such as the one described here, may contribute positively to good preparation for students to be part of high-performance teams. Training students to work proactively in high-performance teams is a crucial aspect of making them competitive in the labor market. This close relationship, between polytechnics/universities and leading companies, in addition to increasing the added value for the preparation of students for the job market, also strongly increases their employability rates, provided that the companies involved operate in vanguard areas. This relationship naturally becomes a win-win relationship as companies can also recruit young graduates who are better prepared for their hiring requirements. It also becomes a motivating factor not only for

students (primary beneficiaries) but also for companies (which recruit well-prepared human resources) and for polytechnics/universities (graduates with high employability).

This study shows that there are benefits for teachers, students and practitioners in following this form of collaboration. Some of the actionable items, useful for both teachers and practitioners to align better software engineering teaching with industrial requirements are: experimentation and sharing practical knowledge about agile methodologies and software development tools used by the industry; sharing experience to test and document the artifacts produced; sharing experience about best software development practices; better understanding of how to deal with different perspectives in a distributed team work environment and deal with cutting edge agile project management tools.

To ensure communication between the different stakeholders, an important aspect to safeguard in agile methodologies, the time made available by all is a necessary cost. Therefore, crucially both sides (industry and academia) should be committed to implementing this approach. From another perspective, this strategy is demanding for teachers/supervisors (on the academic side) and tutors/mentors (on the company side) because it requires time to be regularly dedicated to the process of monitoring the project. The strategy is also demanding for students because it requires a continuous work pace and commitment to the goals established in each sprint. In other words, it is essential that everyone is committed to the strategy and that they allocate time to perform their own roles. Despite these costs, in our experience, the benefits that can be achieved in improving the preparation of the respective students for the job market compensate for the time dedicated by all those involved (academy, industry, students and other stakeholders involved in this approach).

Author Contributions: Conceptualization, methodology, formal analysis, writing—original draft preparation, writing—review and editing: J.M., F.R. and P.G.; validation, investigation: J.M., F.R., P.G., A.M., D.F. and H.V. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Raghavan, J.; Towhidnejad, M. Challenges/Issues in a industry-academic collaboration. In Proceedings of the 2006 Annual Conference & Exposition, Chicago, IL, USA, 18–21 June 2006.
2. Garousi, V.; Pfahl, D.; Fernandes, J.M.; Felderer, M.; Mäntylä, M.V.; Shepherd, D.; Arcuri, A.; Coşkunçay, A.; Tekinerdogan, B. Characterizing industry-academia collaborations in software engineering: Evidence from 101 projects. *Empir. Softw. Eng.* **2019**, *24*, 2540–2602. [CrossRef]
3. Dutra, A.C.S.; Prikladnicki, R.; Conte, T. What Are the Main Characteristics of High Performance Teams for Software Development? In Proceedings of the 17th International Conference on Enterprise Information Systems—Volume 2, Barcelona, Spain, 27–30 April 2015; pp. 145–152.
4. Metrôlho, J.C.; Ribeiro, F.R. Holistic Analysis of the Effectiveness of a Software Engineering Teaching Approach. *Int. J. Adv. Softw.* **2019**, *12*, 46–55.
5. Serban, C.; Vescan, A. Towards an Evaluation Process around Active Learning based Methods. In Proceedings of the Frontiers in Education Conference, Uppsala, Sweden, 21–24 October 2020; Volume 2020, pp. 1–7.
6. Gary, K. Project-Based Learning. *Computer* **2015**, *48*, 98–100. [CrossRef]
7. Brungel, R.; Ruckert, J.; Friedrich, C.M. Project-Based Learning in a Machine Learning Course with Differentiated Industrial Projects for Various Computer Science Master Programs. In Proceedings of the 2020 IEEE 32nd Conference on Software Engineering Education and Training, CSEE and T 2020, Munich, Germany, 9–12 November 2020; pp. 50–54.
8. Rodrigues, P.; Souza, M.; Figueiredo, E. Games and gamification in software engineering education: A survey with educators. In Proceedings of the Proceedings—Frontiers in Education Conference, FIE, San Jose, CA, USA, 3–6 October 2018; Volume 2018, pp. 1–9.

9. Malhotra, R.; Massoudi, M.; Jindal, R. An innovative approach: Coupling project-based learning and game-based learning approach in teaching software engineering course. In Proceedings of the 2020 IEEE International Conference on Technology, Engineering, Management for Societal Impact Using Marketing, Entrepreneurship and Talent, TEMSMET 2020, Bengaluru, India, 10 December 2020.
10. Hamou-Lhadj, A.; Gherbi, A.; Nandigam, J. The Impact of the Model-Driven Approach to Software Engineering on Software Engineering Education. In Proceedings of the 2009 Sixth International Conference on Information Technology: New Generations, Las Vegas, NV, USA, 27–29 April 2009; pp. 719–724.
11. Gren, L. A Flipped Classroom Approach to Teaching Empirical Software Engineering. *IEEE Trans. Educ.* **2020**, *63*, 155–163. [CrossRef]
12. Garcia, R.E.; Messias Correia, R.C.; Olivete, C.; Costacurta Brandi, A.; Prates, J.M. Teaching and learning software project management: A hands-on approach. In Proceedings of the 2015 IEEE Frontiers in Education Conference (FIE), El Paso, TX, USA, 21–24 October 2015; pp. 1–7.
13. Igaki, H.; Fukuyasu, N.; Saiki, S.; Matsumoto, S.; Kusumoto, S. Quantitative Assessment with Using Ticket Driven Development for Teaching Scrum Framework. In *Proceedings of the 36th International Conference on Software Engineering*; Association for Computing Machinery: New York, NY, USA, 2014; pp. 372–381.
14. Paasivaara, M.; Durasiewicz, S.; Lassenius, C. Using Scrum in Distributed Agile Development: A Multiple Case Study. In Proceedings of the 2009 Fourth IEEE International Conference on Global Software Engineering, Limerick, Ireland, 13–16 July 2009; pp. 195–204.
15. Cico, O.; Jaccheri, L.; Nguyen-Duc, A.; Zhang, H. Exploring the intersection between software industry and Software Engineering education—A systematic mapping of Software Engineering Trends. *J. Syst. Softw.* **2021**, *172*, 110736. [CrossRef]
16. Oguz, D.; Oguz, K. Perspectives on the Gap between the Software Industry and the Software Engineering Education. *IEEE Access* **2019**, *7*, 117527–117543. [CrossRef]
17. Rybnicek, R.; Königsgruber, R. What makes industry–university collaboration succeed? A systematic review of the literature. *J. Bus. Econ.* **2019**, *89*, 221–250. [CrossRef]
18. Rajalo, S.; Vadi, M. University-industry innovation collaboration: Reconceptualization. *Technovation* **2017**, *62–63*, 42–54. [CrossRef]
19. Myoken, Y. The role of geographical proximity in university and industry collaboration: Case study of Japanese companies in the UK. *Int. J. Technol. Transf. Commer.* **2013**, *12*, 43–61. [CrossRef]
20. Laursen, K.; Reichstein, T.; Salter, A. Exploring the Effect of Geographical Proximity and University Quality on University–Industry Collaboration in the United Kingdom. *Reg. Stud.* **2011**, *45*, 507–523. [CrossRef]
21. Figueiredo, N.; Fernandes, C. Cooperation University–Industry: A Systematic Literature Review. *Int. J. Innov. Technol. Manag.* **2020**, *78*, 2130001. [CrossRef]
22. Ankrah, S.; AL-Tabbaa, O. Universities–industry collaboration: A systematic review. *Scand. J. Manag.* **2015**, *31*, 387–408. [CrossRef]
23. Wong, W.E. Industry Involvement in an Undergraduate Software Engineering Project Course: Everybody Wins. In Proceedings of the 120th ASEE Annual Conference and Exposition, Atlanta, GA, USA, 23–26 June 2013.
24. Tuzun, E.; Erdogmus, H.; Ozbilgin, I.G. Are Computer Science and Engineering Graduates Ready for the Software Industry? Experiences from an Industrial Student Training Program. In Proceedings of the 2018 IEEE/ACM 40th International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET), Gothenburg, Sweden, 25 May 2018–3 June 2018; pp. 68–77.