Teaching with Agile-Based Exercises: A Modern Method to Teach and Learn Software Engineering

Lineu Alves Lima Filho, Dr. Paulo Marcelo Tasinaffo

***Abstract*—The quality of software applications rely heavily on Software Engineers. However, Software Engineering is a challenging subject for students, thus affecting their professional performance. Several approaches have helped students overcome these challenges. Agile method is among them and considered to be extensively used in the academic field nowadays. Although this method has introduced several benefits, the literature reports some challenges to introducing it to learners. To assess the efficiency and acceptance, the quality of the projects developed, and the ability to enhance students' interpersonal skills, we designed and applied a new approach to teach the Agile method (Scrum). This research paper presents the results we identified while applying the Teaching with Agile-Based Exercises (TABE) method at the university undergraduate level. We identified increases in the quality of projects developed and the ability for students to enhance their interpersonal skills. Our analyzes also revealed the efficiency and acceptance of the proposed method. In conclusion, the TABE method has improved student learning over the three semesters of this study; this improvement will need to be considered in future studies of education at the undergraduate level, contemporary teaching methodologies, and updating of textbooks to incorporate new teaching methodologies.**

***Index Terms*—Agile Method at Undergraduate University, Scrum Practices, Software Engineering Education, Teaching with Agile-Based Exercises.**

I. Introduction

The Information Technology (IT) market has a global expectation of growth in the next years. According to the International Data Corporation [1], approximately 50% of companies in Latin America will be active producers of software by 2025. The report also predicts that the IT market in Latin America will grow by 4.8% in 2020. In Brazil, this growth will likely be at 4.5% in the same period.

Despite this expansion, the IT market faces some challenges concerning a shortage of skills. In 2020, a deficit of 570 thousand qualified IT professionals is expected [2]. Despite the expected drop in this deficit, the shortage of IT professionals will still be at 290 thousand in 2024 [3].

Studies [4] and [5] have shown that knowledge gaps exist between what students learn at the university and what organizations value when hiring future graduates. According to a study [6], the three main challenges of organizations when hiring professionals in Brazil are: 1) low technical skills (33%), 2) lack of professional experience (23%), and 3) lack of adequate interpersonal skills (19%).

To reduce these knowledge gaps, this research study reports the use of a new approach to teaching Software Engineering (SE) through the Agile method (Scrum). It aims to 1) analyze the application and contribution of using the Teaching with Agile-Based Exercises (TABE) method, 2) increase the quality of students learning in SE, and 3) provide opportunities for students to enhance their interpersonal skills.

The research experiments involved two undergraduate programs at the Federal Institute of São Paulo (IFSP): Control and Automation Engineering (CAE) and Mechanical Engineering (ME), which both offer two computing courses: Introduction to Programming and Object-Oriented Programming. To verify our objectives, CAE used Agile-Based Exercises (Experimental Group) and ME used Traditional activities (Control Group).

Each course had 2 phases: Scrum Training and Project. The former to teach students Scrum practices and SE concepts and the latter for them to develop SE applications while practicing the knowledge acquired in agile methodology. During 3 academic semesters, 107 students from 4 courses were divided into groups of 3 to 5 members. Each group participated in 6 sprints to learn Scrum and develop a software prototype in an agile environment.

With quantitative and qualitative approaches during the analysis of the results, the proposed method was efficient in teaching Scrum and approved by the students. The method also provided an improvement in the quality of software prototypes developed by the students and granted them the opportunity to enhance their interpersonal skills.

II. Background

A. *The Agile Method*

In 2001, a group of 17 software professionals created the Agile Manifesto [7]. This document states the Agile values and principles for software development and is currently an essential reference for any Agile methodology or SE project.

The Agile method emerged to support the software development process [8], [9]. Its goal is to develop and deliver the requested product in small parts, in short and frequent iterations. During deliveries, Developers and Testers present deliverables and stakeholders comment, analyze, and may request changes in future iterations. This process increases the satisfaction of the stakeholders, who have the flexibility to change the product during its development, and reduces the expectations of the development team, who receive feedback about the product at the end of each iteration [10].

B. *Agile in Organizations*

A survey conducted by CollabNet VersionOne [11] identified the most common agile methodologies used in the organizations. 1.319 respondents from 6 continents

Lineu Alves Lima Filho, Computer Science Department, Federal Institute of São Paulo (IFSP), São José dos Campos, Brazil.
Paulo Marcelo Tasinaffo, Computer Science Division, ITA, São José dos Campos, Brazil.

answered the survey questions. They were mainly from North America (47%) and Europe (30%). Organizations and software departments varied from small to large size. 46% of professionals were from organizations of more than 5.000 people and 40% from software departments of more than 1.000 IT professionals. The survey revealed that with 54% Scrum is the most common agile methodology among the respondents, as shown in Fig 1.

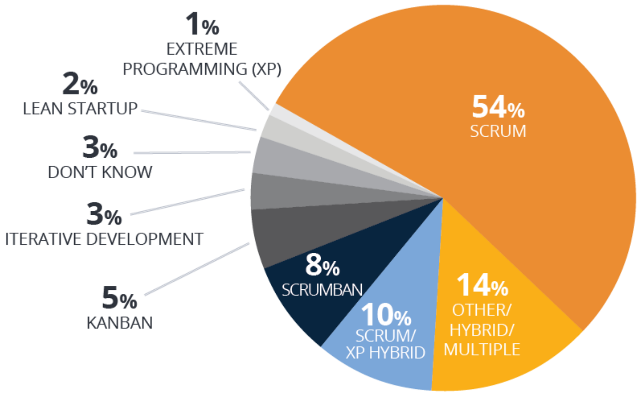


Fig 1 – Use of Agile in Industry [11]

C. *Agile in Academia*

The literature shows that the Agile method has been adopted in academia for 2 main purposes: (i) to conduct research projects; and (ii) to support the teaching and learning process.

i. *Agile in Research Projects*

A survey conducted by Aquino, Neto and Lopes [12] revealed that 25% of the respondents have already used the agile method (whether fully or partially) in research projects. 55% of researchers responded they intend to use it although they have not used it yet. Only 10% of the respondents answered that they do not want to adopt agile for conducting research.

According to Pirro [13], in her article "*How agile project management can work for your research*", published in Nature Magazine, increase in productivity, knowledge development, motivation and morale, and a reduction of misunderstandings are some of the benefits of agile research projects. Some academic research institutes [14], [15] and [16] contributed with their successful results and findings in using agile in research projects.

ii. *Agile in the Classroom*

Several educators have been using agile in supporting the teaching and learning process in the classroom. This can be seen in Salza, Musmarra and Ferrucci [17] review paper, in which they cite several research authors that have used agile in the classroom with success. Although most of the experiments were in the SE and Computer Science courses (91.5%), agile is also used in other fields of studies (8.5%).

To identify the most used agile methodology in education, the authors [17] performed a literature review. They analyzed approximately 200 articles and concluded that Scrum (41%) is the most common agile methodology in the education area, as shown in Fig 2. Pair Programming (19.5%) and XP (11%) are also reasonably used at universities and professors interested in these two methods reported benefits and good results in their experiments.

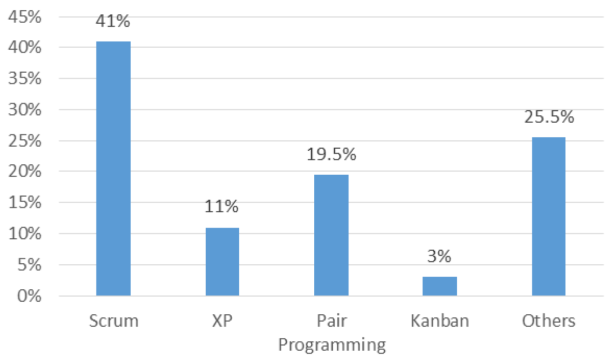


Fig 2 – Use of Agile in Education [17]

The author of this research work participated in 4 graduate courses offered by the Aeronautics Institute of Technology (ITA), which used the agile method to support the teaching-learning process. These experiences have motivated the author to apply this methodology to the undergraduate engineering programs at IFSP, where he works as a Professor in the Computer Science department.

The experiments of this research used Scrum as it is the most common methodology in both the industry (Fig 1) and academia (Fig 2).

III. The Scrum Framework

Scrum is an agile method commonly used in the software development process. Takeuchi and Nonaka [18] introduced the concept of Scrum for the first time in 1986, in an article published in the Harvard Business Review. The two Japanese business experts aimed at using this new approach to increase the speed and flexibility of companies in developing new commercial products. The name of the methodology came from an analogy of the production process with the sport of rugby, which has a play called Scrum.

In the early 1990s, Jeff Sutherland and Ken Schwaber published the article "*SCRUM Development Process - Advanced Development Methods*" in the Business Object Design and Implementation journal. They each reported their experiences in developing software using an enhanced approach, which they called Scrum. According to Schwaber [19], the name of the method was based on the analogy of Takeuchi and Nonaka. Since then, they have published most of the papers about Scrum and they are among the 17 authors that signed the Agile Manifesto.

According to the book "*The Scrum Guide*", written by Schwaber and Sutherland [10], Scrum is a framework in which people collaborate to solve complex problems, with a focus to deliver high-value products. The Scrum framework is lightweight, simple to understand, and difficult to master. It is flexible and enables the continuous improvement of products, teams, and working environment.

The Scrum framework consists of roles, events, and artifacts [10], which interact among them.

- **Roles:** Product Owner (PO), Scrum Master (SM), and Development Team (DT);
- **Events:** Sprint, Sprint Planning, Sprint Goal, Daily Scrum, Sprint Review, and Sprint Retrospective;
- **Artifacts:** Product Backlog, Sprint Backlog, and Burndown Chart.

The small number of roles aims at reducing the complexity of project management by promptly defining the role of each member.

Each event has a time-boxed duration to be completed and it may end whenever the objective is achieved, thus ensuring effective use of time.

Artifacts represent work for the Development Team and value for the stakeholders. The artifacts provide opportunities for both parties to reflect on what has been done and what needs to be done next. They share the same goal according to their roles: the Development Team pursuing continuous improvement of the process and the stakeholder(s) continuous improvement of the product.

IV. Requirements Management in Scrum

As User Stories are the main focus of this study, this section will explain its format and use. It also explains the process of a Kanban tool and its use in this research project.

- *User Story*

In the Agile method, the basic form of requirements are User Stories, which are descriptions of features that fit into a small card. According to Leffingwell [20], a User Story (US) template follows the standard shown in Fig 3. Each user story is usually small functional requirements, which will result in small product increments at the end of each sprint. It basically contains a short description of the functionality, an ID, the priority, and an estimated effort for the work to be done.

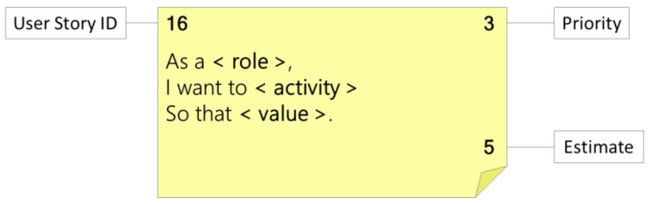


Fig 3 – User Story Template

The idea is to deliver the desired product to the stakeholder(s) in small pieces, providing stakeholder(s) and Scrum Team(s) the opportunity to share feedback (Sprint Review) and make any necessary change to the ordered product items between sprints [21], [22].

A quality user story has a set of criteria defined by the acronym INVEST (Independent, Negotiable, Valuable, Estimable, Small, and Testable). User stories can be rewritten and reviewed if the person, who writes user stories, fails to adopt these criteria [23].

The traditional activities converted into agile-based exercises in the format of user stories followed the INVEST standard of quality criteria.

- *Kanban*

Kanban is a visual management tool for managing tasks and workflows using columns and cards, as shown in Fig 4.

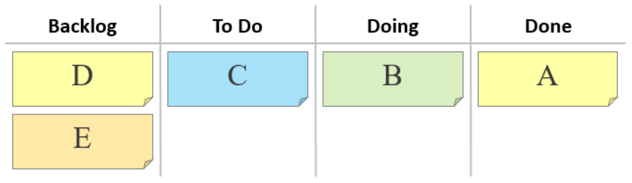


Fig 4 – Kanban Board

The board usually contains 4 columns where the user stories that will be worked on during the sprint are placed in the "Backlog" column. Once the user stories are broken into small tasks, these tasks are placed on the "To-Do" column. These tasks are then assigned to team members in a balanced way. As soon as people start working on a task, they can move it to the "Doing" state. When the task is fully performed, it is moved to the "Done" state.

V. The TABE Method

The TABE method consists in converting traditional software engineering activities into agile-based exercises. The objective is to provide students with exercises for learning computer programming according to the methodology used, in this case, Scrum.

Traditional computer programming activities, depicted in Fig 5, are usually descriptive statements of what students should do to learn how to create software.

1) Write a C++ program that receives the gross salary of an employee, and calculate and print the net salary. This is composed of the gross salary plus sales commission and tax deduction.

Fig 5 - Traditional Activity

Most computer programming textbooks have traditional computer programming activities. Although it has been taught for many years with these resources, these types of activities present some disadvantages to the students' learning process.

- Primarily descriptive, stating what students should do;
- Ordered sequentially, usually by the level of difficulty that the author defined;
- Entirely focused on technical knowledge;
- Not related to any teaching methodology;
- Focused on individual work unless the teacher adapts these activities to be done in groups of students.

As Scrum has increasingly been used in the classroom (presented in section II – Agile in Academia), this research aims to present a different format of software engineering exercises based on agile methodology, as shown in Fig 6.

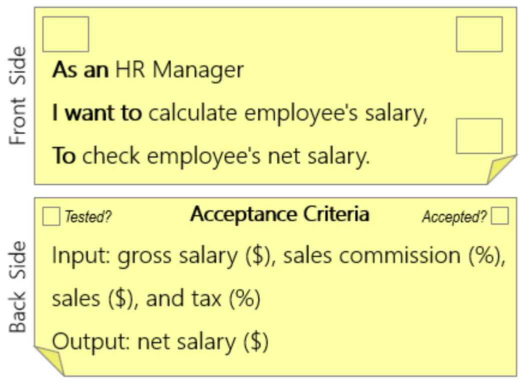

Front Side
As an HR Manager
I want to calculate employee's salary,
To check employee's net salary.
Back Side
Tested?
Acceptance Criteria
Accepted?
Input: gross salary ($), sales commission (%), sales ($), and tax (%)
Output: net salary ($)

Fig 6 – Example of Agile-Based Exercise

In the TABE method, exercises are written in cards using the US format. The front side of the card presents the requirement, and the back of the card states the acceptance criteria for the exercise to be considered done.

With this new exercise format, students need to prioritize the User Stories, aiming at delivering as much value as they can at the beginning of the project. They also need to estimate the effort of the User Stories before starting the development. The estimation effort allows teams to accommodate as many user stories as they can in a fixed time-box (sprint), optimizing team time and work.

The acceptance criteria are checked by two people: the Tester, who is a team member responsible for the testing task of the US; and the PO, who participates in the Sprint Review and, after the scrum team presentation, accepts or rejects the US implementation.

This new teaching method expects to increase student acceptance of Scrum, improve the quality of applications developed by students, and increase student interactivity, thus creating opportunities to practice interpersonal skills.

As subsidiary benefits of the agile-based exercises, they are focused on functionality, because they are written in the format of user stories. They are based on the agile method, supporting students to transition to the agile methodology while they learn software engineering. Students can reflect more on software testing as it is part of the learning process. Lastly, they can experience a real software project academically.

VI. Related Work

The identification of related work, adapted from Marques [24], consisted of a 5-stage process: (1) select the appropriate scientific databases, (2) identify the related search terms, (3) apply the predefined selection criteria, with clear inclusion and exclusion rules, (4) compare the related works with this research regarding the Scrum teaching method, and (5) analyze and synthesize the identified related works, as shown in Fig 7.

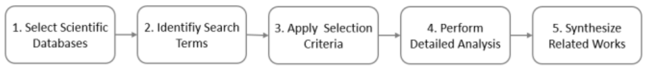

1. Select Scientific Databases
2. Identify Search Terms
3. Apply Selection Criteria
4. Perform Detailed Analysis
5. Synthesize Related Works

Fig 7 – Stages of the Literature Review

In stage 1, the databases selected were ScienceDirect, Scopus, and Web of Science.

In stage 2, the search terms were analyzed and identified. The searches performed against the databases focused on two main terms of interest: a) how undergraduate students learn the agile method and b) the existence of exercises based on agile methodology to teach SE. All search queries used title, abstract, and keywords fields. Table 1 presents the search terms identified and the results retrieved after executing the search queries by May 2020.

Table 1 - Search Terms and Results

Search Terms	ScienceDirect	Scopus	Web of Science
Agile AND Academi* AND Software AND Exercise*	4	12	7
Agile AND Academi* AND Software AND Activit*	12	55	39
Scrum AND Teach* AND Software AND Academi*	4	38	16
Total		**187**	

As the words "exercise" and "activity" are sometimes used interchangeably in computer science textbooks, they were both included in the search. The queries used the asterisk (*) wildcard character to include the terms Academic or Academia, Teach or Teaching, Exercise or Exercises, and Activity or Activities.

Stage 3, the exclusion criteria were defined as publications in which full text is not available and is not written in English, and the removal of duplicate items. The inclusion criteria included scientific publications such as published journals, conference papers, and book chapters. Also, the publications needed to present empirical evidence and be related to the university undergraduate level. After scanning the title, abstract, and keywords of 187 publications and applying the selection criteria, 8 distinct works related to this research were identified.

Stage 4 consisted of an in-depth analysis of the 8 related works. The two main goals were (1) to qualitatively verify the intersection of the related works with this research study; and (2) to certify whether the method used in this research had already been reported in the scientific community. This refinement analysis took into consideration the following 4 questions.

Q1. Did the study use Scrum in academia?
Q2. Was it at undergraduate engineering level?
Q3. Did it involve SE courses?
Q4. Did it use agile-based exercises?

The author scanned the 8 papers searching for the responses of the 4 predefined questions. The "✓" means the article fulfills the inclusion question and the "×" means the opposite. Table 2 shows the responses to the questions defined for the detailed analysis.

Table 2 – Responses to the Detailed Analysis Questions

Article	Q1	Q2	Q3	Q4	Total
[25]	✓	✓	✓	×	**3**
[26]	✓	✓	✓	×	**3**
[27]	✓	✓	✓	×	**3**
[28]	✓	✓	✓	×	**3**
[29]	✓	×	✓	×	**2**
[30]	✓	×	✓	×	**2**
[31]	×	×	×	×	**0**
[32]	×	×	×	×	**0**

Although there are several approaches to teach Scrum, this work will prioritize the analyses of the academic ones at the university undergraduate level, which have been published to the scientific community, reporting the outcome of empirical evidence.

According to the results presented in Table 2, the most relevant works that have an intersection with this research are [26], [27], [25], and [28] because they presented positive responses to Q1, Q2, and Q3.

Finally, stage 5 involved the reading and synthesizing the four related works previously identified. Next, the summaries are presented following an analysis of the findings.

Lego4Scrum [26] provides participants the opportunity of running a full-cycle of Scrum. In groups of 4 to 6 people, they receive a project requirement to build a city using paper, pencil, and Lego Bricks. The game has been applied in the academic area and successful experiment results reported [33], [34] and [35]. Several other studies in different university courses have emerged from this game. It has also been used in organizations for professional training purposes.

SCRUMIA [27] follows the same principles of Lego4Scrum, with only one difference. It removes the Lego bricks from the resources to make it accessible to low-income academic institutions. Its focus is to teach the full-cycle of Scrum to undergraduate students at public universities (government-funded).

The Scrum Game [25] is a prototype of a web-based system designed for academic purposes. It aims to be a complementary resource in SE courses to help students to learn the Scrum practices. The author described the implementation of the game and reported the results of the experiments while applying it to undergraduate students. They responded to a questionnaire to evaluate the game experience and 95% of the students (21 participants) answered they would have a better understanding of Scrum if the game was part of a course.

SCRUMI [28] is an electronic board game with questions and answers. It introduces key concepts of the Scrum framework in the academic environment. It helps students from organizing the scrum team up to reviewing the sprint backlog. This game is suitable for participants who have never used Scrum before and to be used at the very beginning of a course when students have to form groups and understand the basic concepts of the method.

The literature review showed that Scrum is mainly taught through the use of educational games. They all try to simulate environments in which participants learn the Scrum method. Also, all 8 related works had negative answers to Q4. Thus, there is no evidence that agile-based exercises designed with the US model in mind have been reported to the scientific community.

The objective of this work is to introduce a novel approach to teach Scrum and SE. We will convert traditional software engineering activities into agile-based exercises based on the US design, in which the literature review has not found any related work. It will certainly be a valuable addition to the approaches currently used in the academic environment. It may also lead authors to review and update textbooks to follow a more contemporary teaching approach.

VII. Methodology

A. *Research Environment*

The Federal Institute of São Paulo (IFSP) is a public government-funded educational institution, located in the state of São Paulo. It currently has 37 campus in operation which offers professional, undergraduate and graduate programs.

The research experiments were conducted in the campus located in the city of São José dos Campos, in the state of São Paulo. The undergraduate programs involved were Control and Automation Engineering and Mechanical Engineering, which both offer two computing courses: Introduction to Programming and Object-Oriented Programming, as shown in Fig 8. Both computer programming courses adopt the C++ programming language as the former is about structured programming and the latter uses Object-Oriented Paradigm.

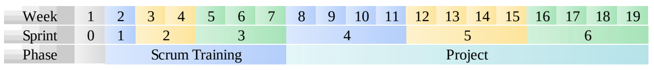
Control and Automation (Experimental)	Introduction to Programming PRCE1 2019-S1 TABE Method	Object-Oriented Programming PRCE2 2019-S2 TABE Method
Mechanical (Control)	Introduction to Programming APRM2 2018-S2 Traditional Method	Object-Oriented Programming PCOM3 2019-S1 Traditional Method

Fig 8 – Outline of the Courses

As for the gender of the participants, in the Mechanical Engineering program, 73% of students (38) were male and 27% (14) female. Their age ranged between 17 and 28 years old. In the Control and Automation Engineering program, 69% of students (38) were male and 31% (17) female. Their age ranged between 17 and 31 years old. Gender and age in both engineering programs were very similar.

B. *Applying the Method*

The courses were divided into two phases named Scrum Training and Project. Then, each phase was subdivided into three sprints. Fig 9 shows the semester planning week by week, the duration of each sprint, and the phases of the course.



Week	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
Sprint	0	1	2		3			4				5				6			
Phase		Scrum Training						Project											

Fig 9 – Course Planning Timeline

During the Scrum Training phase, students learned the Scrum practices presented in section III, which were carefully planned and introduced during weeks 2, 3, and 5, as shown in Table 3. These practices were conceptually explained during three short presentations of approximately 20 minutes each. The main purpose of sprints 1 to 3 was to apply the Scrum practices and the software engineering concepts presented through the classes.

During the Project phase, students applied the Scrum practices while they developed and tested a software engineering application. The main aim in this phase was to provide students the opportunity to not only develop a software engineering project, but also manage it using the Scrum framework.

Table 3 – Scrum Practices Introduction Outline

Week	Category	Scrum Practice
2	Role	Team Member
	Activity	Sprint
	Activity	Sprint Planning
	Artifact	Sprint Backlog
3	Role	Scrum Master
	Activity	Weekly Meeting
	Activity	Sprint Retrospective
	Artifact	Product Backlog
5	Role	Product Owner
	Activity	Project Goal
	Activity	Sprint Review
	Artifact	Burndown Chart

Three Scrum practices were adapted to suit the academic environment, as shown in Table 3. The "Development Team" role was replaced by the "Team Member" (TM) role as all members, including the SM, were responsible for the development and test of the software prototype. Only the project phase used the term "Project Goal", originally called "Sprint Goal". Each scrum team set a project goal for the software project phase. As the course classes were weekly, the "Daily Scrum" became the "Weekly Scrum" so that all scrum teams could hold meetings in person during the class.

During sprints 1 to 3, the **Experimental Group** received a list of user stories (Agile-Based Exercises) and the **Control Group** received a list of descriptive activities (Traditional), as presented in section V. The content of the exercises were the same for both groups, differentiating only the format.

The following sections will detail how students learned the Scrum practices during sprints 1 to 3 (Scrum Training phase), and will explain the development lifecycle and the type of projects students developed during sprints 4 to 6 (Project phase).

- *Sprint 1*

As the first class was devoted to course presentation and welcome students to the academic institution, sprint 1 started in week 2 and ended in week 3, lasting 1 week.

The first four Scrum practices were introduced to the students in week 2. Apart from these four practices, the concepts of Scrum Team (ST), User Story (US), and Planning Poker were also explained to the students.

- o **Scrum Team** – students self-organized them into Scrum Teams, with no interference from the professor or assistant. Each ST was composed of 3 to 5 team members. At this time, all team members were designated simply as team members. Professor emphasized collaboration, cooperation, and a balanced workload among team members in a ST. Its main objective was to deliver value in a short period of time.
- o **Team Members** – the TM role replaced the Development Team role to emphasize that all students needed to work on the development and test tasks, even the student playing the SM role, later introduced in sprint 2.
- o **Sprint** – as students received an explanation and access to the Course Planning Timeline (Fig 9), they were aware of all sprint duration.
- o **User Story** – the concept of US was introduced to the students, as explained in section IV.
- o **Planning Poker** – the concept of story points was introduced and the Fibonacci-like format (0.5, 1, 2, 3, 5, 8, 13, 20, 40, and 100) adopted. After that, students practiced the Planning Poker while estimating the US efforts of this sprint. Professor also emphasized that story points should be used to balance the workload among team members.
- o **Sprint Planning** – students prioritized the Sprint Backlog and estimated the effort of user stories. Then, they broke the user stories into tasks and split them among the team members. This first Sprint Planning was done in the classroom and questions and doubts were resolved promptly, avoiding any misunderstanding regarding the concepts just introduced.
- o **Sprint Backlog** – it was a spreadsheet that students needed to use during each sprint to follow up on the ST progress. The first Sprint Backlog was filled out in class during the Sprint Planning meeting of Sprint 1.

- *Sprint 2*

- o **Sprint Retrospective** – as soon as the professor presented this event, students responded to the two questions regarding sprint 1 and they did the same at the end of all 6 sprints. Students used an electronic form to send their responses. After all students responded, each ST received the responses from their members for analysis and discussion.
- o **Scrum Master** – professor explained and emphasized the SM's responsibilities. Then each ST elected a member of their team to be the SM.
- o **Weekly Meeting** – as this is a 2-week sprint, students exercised the weekly meeting in week 4. They answered the three questions (What did I do yesterday? What will I do today? Do I see any impediment?) within their scrum teams.
- o **Product Backlog** – the Product Backlog was used as a US repository in this sprint. Students prioritized and estimated the effort of all user stories in the Product Backlog. During the sprint, scrum teams iterated between the Product Backlog and the Sprint Backlog. Each time the STfinished a US, they moved another one from the Product Backlog to the Sprint Backlog.

- *Sprint 3*

- o **Product Owner** – in sprints 1 and 2 the professor responsible for the course implicitly played the PO role. In this sprint 3, the students were aware the professor was the PO [36]. The objective was to introduce the importance of the PO role in the project and to exercise the communication between SM and PO. During the project phase, from sprints 4 to 6, professors from the institution and experienced students played this role. They were invited and voluntarily accepted to participate in the project. The experienced students were in the 5th year of the engineering program and already had at least one year of experience as a Software Engineer.
- o **Project Goal** – each ST defined their own Project Goal according to the topic of the software project chosen in this sprint. The Project Goal was defined for the entire Project phase, which consisted of 3 sprints of 4 weeks each.
- o **Sprint Review** – during sprints 1 and 2, students informally presented their artifacts as the user stories were independent. In this sprint 3, students learned the concept of a sprint review and they presented their projects following the Scrum ceremonies.
- o **Burndown Chart** – professor presented this artifact and explained its use. In this sprint, students developed small to mid-sized software engineering projects and they put into practice the burndown chart to follow up their progress.

- *Sprints 4 to 6*

From sprints 4 to 6, students developed and tested a software application while they executed the full lifecycle of the Scrum framework used in this research, as shown Fig 10. The main objective was to: a) increase the quality of applications produced by the students academically; and b) provide more opportunities for students to develop or improve their interpersonal skills.

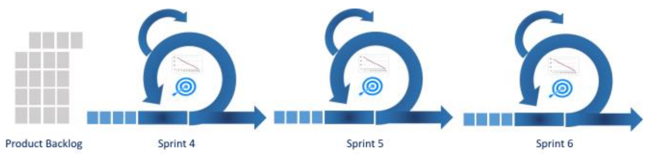


Fig 10 – Sprint Overview of the Project Phase

Each ST received a distinct software engineering project. All projects shared the same purpose: "develop an application to assess the quality control of a car". The only variable for each project was the type of car (hatchback, sedan, SUV, convertible, wagon, luxury, sports car, hybrid car, electric car, and diesel car).

VIII. Results

Hypothesis 1 was evaluated based on (i) a two-tailed independent samples t-test, which used an individual student assessment results of the Scrum Practices; (ii) the observation method conducted by the professor; and (iii) a questionnaire voluntarily answered by the students about their opinion on the efficiency of the proposed method.

The multiple-choice assessment in (i) was applied for both experimental and control groups at the end of the 7th class, six weeks after the students had been studying and using the Scrum Practices. The assessment consisted of questions from the book "*Scrum and Agile in Projects: Complete Guide*" [37]. It assessed the students' ability to retain and apply the knowledge acquired from week 2 to 7. The observation in (ii) occurred during the Scrum Training phase, from week 2 to 7. The questionnaire in (iii) was available online through an electronic form for the students to respond between weeks 18 and 19.

Hypothesis 2 was evaluated based on a two-tailed independent samples t-test. To evaluate software quality, H2 considered only user stories developed and tested with success by the end of each sprint during the Project phase, between week 8 and 19. This hypothesis used empirical data to study its validity.

Hypothesis 3 was evaluated quantitatively through a questionnaire and qualitatively by the observation method during the Project phase (between weeks 8 to 19). The students voluntarily responded to a questionnaire after presenting the final product of the software project. The professor responsible for the course and the invited participants, who played the PO role during the project phase, observed all scrum teams and took notes about the most relevant findings.

With a significance level of $\alpha = 0.05$ for hypotheses H1 and H2, the null hypothesis was rejected for both H1 and H2. As for hypothesis H3, the triangulation involving the observations of the course professor, Product Owners, and the students' responses to the questionnaire are presented and discussed.

A. *The TABE Method*

The results of hypothesis 1 are presented in this section in the following order: (i) Statistical analysis; (ii) Professor's observations; and (iii) Students' opinions.

i. *Statistical Analysis*

The 55 students who learnt the scrum practices through the TABE method (M = 8.17, SD = 0.82) compared to the 52 students exposed to traditional method (M = 7.11, SD 1.27) had significantly better averages, t(50) = 2.6167, p = 0.0146, with 95% CI = [0.2267, 1.8876].

The students' average result of each Scrum practice in both groups is shown in Table 4.

Table 4 – Students' Average Results by Scrum Practices

Scrum Practices	Experimental	Control
Team Member	9.8	9.4
Scrum Master	7.8	4.6
Product Owner	7.8	6.4
Sprint	7.4	4.6
User Story	8.6	4.2
Story Points	7.2	3.8
Weekly Meeting	8.4	6.2
Sprint Planning	6.4	3.6
Planning Poker	6.2	3.8
Sprint Retrospective	7.4	5.6
Sprint Review	7.6	5.8
Sprint Backlog	8.4	6.2
Product Backlog	6.6	5.0
Burndown Chart	6.6	4.6

Between the Experimental Group (TABE) and the Control Group (Traditional), the most striking average difference is the User Story (2.4). This is comprehended due to the experimental group receiving programming exercises in the form of user stories.

The second-largest difference was Scrum Master and Sprint Backlog (tied with 2.2). Most of the scrum teams in the control group simply ignored the SM role and a few of them had members playing it, but this person was uncertain about the responsibilities of this role. Few scrum teams in the experimental group also had doubts about the duties of the SM role, but with less impact because the TABE method helped in the organization of tasks. The Control Group had difficulties in managing the Sprint Backlog, leaving it outdated most of the time during the sprint. The absence of the SM role in most scrum teams and the need to transfer descriptive activities into user stories contributed to these results.

The third-largest difference was the Sprint Planning (1.8). Again, the extra work the Control Group had when transferring descriptive activities into user stories reflected in this result. Their Sprint Planning meeting lasted longer than the Experimental Group although they were more confident in what they had to do after the meeting.

ii. *Professor's Observations*

The following are the most relevant findings during the professor's observations in class while students performed programming activities using the scrum method.

In the experimental group, the Accepted Criteria of the user stories was the most commented and valued feature by students compared to traditional software engineering activities. It reduced the students' expectations of what they should do to deliver correct activities (product increment) that match the requirements (user stories). In the control group, most scrum teams were more concerned about understanding the requirements (descriptive exercises) rather than analyzing other resources (e.g. accepted criteria and the PO) that they had available.

All scrum teams in the control group converted the traditional exercises into user stories. They reported that the conversion process led them to a better understanding of the requirements and it also raised questions among the team members. A TM stated "*It* [the process conversion] *made us talk to members of our group more often than we usually do in other courses*", with the other team members agreeing with the statement. They also revealed that they feared not delivering the development artifacts on time, because of the time spent on the process conversion. They found the activity of writing user stories very challenging since they asked the professor several questions about it.

iii. Students' Opinions

At the end of PRCE1 and PRCE2 courses (experimental groups), students voluntarily answered a questionnaire about their opinion on TABE and Traditional methods. They responded to two questions and were able to leave comments or even justify their choice. Question one asked if students had learned software engineering with traditional activities before. Question two requested their opinion about the most efficient learning method, between TABE and Traditional.

Only students who responded "yes" to the first question were considered in the analysis because they were able to compare both methods. As responding to the questionnaire was voluntary, 37 of the 55 students participated in this survey. These, 24 students answered the first question positively, and thus only this group had their answers considered in Fig 11, which shows students' preferred method. 92% of the students (22 students) responded that the TABE method facilitated the learning process of SE and only 8% (2 students) said that the two methods have relevant benefits.

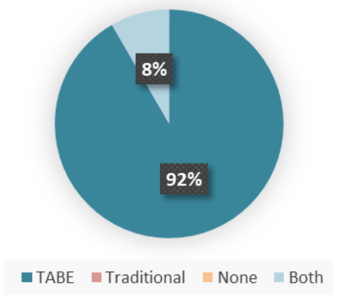


Fig 11 – Students' Preferred Method

Most students mentioned that the TABE method helped in the application of Scrum practices due to the type of exercises presented between sprint 1 and 3 (Scrum Training phase). For instance, a student commented: "*By seeing how User Stories were written at the beginning of the semester* [sprints 1 to 3] *helped us write our User Stories during the project*".

Therefore, the statistical analysis, the professor's observations, and students' opinions about the method showed support for the first hypothesis, (H1: the TABE method helps students learn Scrum practices).

B. Software Quality

Students who were exposed to the TABE method (M = 10.88, SD = 1.20) developed and tested software applications with more quality than students exposed to traditional method (M = 9.38, SD 1.54), t(32) = 3.0645, p = 0.0046, df = 30, 95% CI = [0.5004, 2.4996].

The number of user stories each ST delivered during the Project phase (sprints 4 to 6) in both experimental and control group is presented in Table 5. The user stories needed to be developed and tested with success by each ST and accepted by the PO during the Sprint Review. Only user stories that met both criteria previously mentioned were considered for quality metrics.

Table 5 – Number of User Stories Delivered by Each Group

Experimental Group		Control Group	
Scrum Team	**US Develired**	**Scrum Team**	**US Develired**
ST01	11	ST01	12
ST02	11	ST02	8
ST03	10	ST03	8
ST04	11	ST04	8
ST05	13	ST05	11
ST06	11	ST06	12
ST07	13	ST07	12
ST08	10	ST08	8
ST09	12	ST09	9
ST10	11	ST10	9
ST11	10	ST11	8
ST12	12	ST12	8
ST13	9	ST13	9
ST14	11	ST14	9
ST15	9	ST15	9
ST16	10	ST16	10

According to the observations of the professor responsible for the courses, the experimental group presented projects with a high degree of completeness, increasing customer satisfaction (Product Owners).

This was also recognized by the POs while they watched the scrum teams' project presentations during the Sprint Review of each sprint.

The PO responsible for the Hybrid Car project stated: "*Scrum* [framework] *really helped both* [experimental and control] *groups. Both of them presented good projects, but the scrum teams in the experimental group delivered quite complete solutions*".

Another PO, who was in charge of the Electric Car project, stated: "*In the two projects* [an experimental and a control group] *in which I was the PO, the scrum teams presented good projects. However, the experimental group stood out in the organization of the sprints and the development of their prototype. The prototype of the control group would still need some final adjustments*".

The other POs also reported comments similar to the ones presented above. Therefore, the statistical results and observations support the second hypothesis (H2: The TABE method helped students increase their software quality).

C. Interpersonal Skills

A triangulation was carried out involving the observations of the course professor, Product Owners, and students. It qualitatively identified the main findings regarding the use and enhancement of the students' interpersonal skills. The observations took place during the Project phase. The professor took notes right after classes. The POs wrote down their observations during the Sprint Planning and Sprint Review meetings. The students responded to a questionnaire about the interpersonal skills before each Sprint Review.

According to the National Curricular Guidelines for the Undergraduate Engineering Course, published by the Ministry of Education [38], the body that regulates the creation and standardization of engineering courses in

Brazil, the interpersonal skills in Table 6, explicitly mentioned in the document, are essential for undergraduate engineers to exercise their professional career.

Table 6 – Skills Required by Graduate Engineers

Skills		
Autonomy	Creativity	Proactivity
Collaboration	Critical Thinking	Problem solving
Commitment	Ethic	Reflection
Communication	Innovation	Self-learning
Cooperative	Leadership	Teamwork

Students were asked to rate how much of each interpersonal skill they practiced on a Likert scale from 1 (very little) to 5 (very much). All interpersonal skills were exercised satisfactorily by both experimental and control groups, as presented in Fig 12. According to the students' responses, the five skills that showed the greatest contrast were self-learning, reflection, autonomy, critical thinking, and leadership.

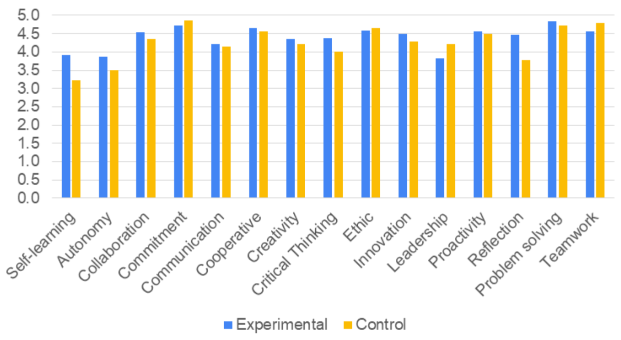


Fig 12 - Average of Each Skill per Group

The level of self-learning and autonomy of the experimental group was higher than the control group. The students' responses match with the observations from the Product Owners and the professor when they mentioned that the experimental group had stronger communication, teamwork and collaboration skills than the control group.

The experimental group also presented a higher level of reflection and critical thinking skills. The TABE method and the academic environment simulating a real software project led students to reflect and think critically about the type and quality of the software they developed. This is supported by the POs observations when they stated that the experimental group was more committed to the software project. The POs mentioned that most scrum teams were very interested to further understand the requirements during the Sprint Planning and concerned about the quality of the deliverables in the Sprint Review.

The leadership was lower in the experimental group and this result is positive in this context. The Agile method emphasizes the horizontal management style, where members are empowered to make important decisions without approval from a manager. Most scrum teams in the experimental group had an active SM who guided their team members successfully. Thus, the reduced leadership in the experimental group revealed that the team members equally collaborated and cooperated within their scrum teams during the software project.

Therefore, the triangulation involving the observations of the course professor, Product Owners, and students about the applied interpersonal skills during the course showed support for the third hypothesis, (H3: the TABE method enhances students' interpersonal skills).

IX. Conclusion

Due to the results achieved with the application of the TABE method in this research, it is believed that the proposed method can also be successfully applied in other undergraduate courses.

It is also believed that there is a need to revise the textbooks used in teaching SE, converting traditional activities into exercises based on the agile method. This will provide a contemporary teaching method, focused on activities that add value to students and provide them opportunities to develop not only technical knowledge, but also interpersonal skills.

In light of Takeuchi and Nonaka's statement [18], "*What we need today is constant innovation in a world of constant change*"; pointing out that the educational field needs to change to innovate the teaching and learning process, and to help address the three main challenges of organizations in Brazil, the main contributions of this research work are:

1) Design of a methodology to teach SE through the use of exercises based on the agile method;
2) Improvement in the quality of software developed and tested by students; and
3) Adequate academic environment for students to develop and enhance their interpersonal skills.

X. Future Work

To improve the teaching of the agile method (Scrum framework) through the TABE method, to which the proposed work is addressed, the following are some suggestions for future work:

- Apply the TABE method to other courses at undergraduate engineering programs, apart from the computer science core disciplines;
- Use the proposed method in a corporate training environment and compare its results with this research study;
- Create a learning environment using Teaching with Agile-Based Exercises (TABE) with Game-Based Learning (GBL) methodologies to evaluate ST productivity;
- Evaluate each ST using the Tuckman Model to allow students to clearly view their learning progress;
- Convert traditional computer programming activities into ATDD and apply it to teach SE;
- Apply the proposed method to other agile development methods, such as Pair Programming and XP;
- Write and publish a book with Agile-Based Exercises and explain how to use them in the academic environment.

References

[1] International Data Corporation (IDC), “IDC Webinar IDC FutureScape Worldwide 2020 Predictions LatAm Implications,” *IDC*, 2019. [Online]. Available: http://www.idclatin.com/2019/Events/futureScape2020/LA_Predictions_SP.pdf. [Accessed: 10-May-2020].

[2] ABES, “Mercado de TI na América Latina deve registrar aumento de 4,8% em 2020, segundo a IDC,” *Associação Brasileira de Empresas*

de Software (ABES), 2020. [Online]. Available: http://www.abessoftware.com.br/noticias/mercado-de-ti-na-america-latina-deve-registrar-aumento-de-48-em-2020-segundo-a-idc. [Accessed: 01-Feb-2020].

[3] Brasscom, "Profissão TI: corrida contra o tempo," *Brasscom*, 2020. [Online]. Available: https://brasscom.org.br/profissao-ti-corrida-contra-o-tempo/. [Accessed: 01-Feb-2020].

[4] I. W. E. F. Agenda, "New Vision for Education: Fostering Social and Emotional Learning through Technology," *World Econ. Forum*, no. March, p. 36, 2016.

[5] S. Valstar, S. Krause-Levy, A. MacEdo, W. G. Griswold, and L. Porter, "Faculty views on the goals of an undergraduate cs education and the academia-industry gap," *Annu. Conf. Innov. Technol. Comput. Sci. Educ. ITiCSE*, pp. 577–583, 2020.

[6] P. Natividade, "Três em cada dez candidatos não têm habilidades técnicas exigidas," *Correio*, 2018. [Online]. Available: https://www.correio24horas.com.br/noticia/nid/procura-se-profissional-qualificado-tres-em-cada-dez-candidatos-nao-tem-habilidades-tecnicas-exigidas/. [Accessed: 01-Feb-2020].

[7] K. Beck *et al.*, "Manifesto for Agile Software Development," 2001. [Online]. Available: https://agilemanifesto.org/. [Accessed: 13-Apr-2020].

[8] A. Yadav, "AGILE : Software development model," *Int. J. Eng. Tech. Res.*, no. 3, pp. 11–17, 2015.

[9] K. S. Lohit and R. Wagh, "An Empirical Study of Agile Software Development with SCRUM Model," *Int. J. Eng. Tech. Res.*, no. 4, pp. 60–63, 2016.

[10] K. Schwaber and J. Sutherland, "The Scrum Guide: The Definitive The Rules of the Game," *Scrum.Org and ScrumInc*, no. November, p. 19, 2017.

[11] CollabNet VersionOne, "The 13th annual STATE OF AGILE Report - 2018," *CollabNet | VersionOne*, vol. 13, p. 16, 2019.

[12] N. M. R. Aquino, A. G. S. S. Neto, and H. S. Lopes, "A study on the perception of researchers about the application of agile software development methods in research," *Commun. Comput. Inf. Sci.*, vol. 802, pp. 137–145, 2018.

[13] L. Pirro, "How agile project management can work for your research," *Nature*, no. April, 2019.

[14] P. M. Tasinaffo, G. S. Gonçalves, A. M. da Cunha, and L. A. Vieira Dias, "Using Monte Carlo method to estimate the behavior of neural training between balanced and unbalanced data in classification of patterns," vol. 7, no. 2, 2018.

[15] I. Ribeiro Lima, T. de Castro Freire, and H. A. X. Costa, "Adapting and Using Scrum in a Software Research and Development Laboratory," *Rev. Sist. Informação da FSMA*, vol. 9, no., pp. 16–23, 2012.

[16] M. Hicks and J. S. Foster, "Adapting Scrum to Managing a Research Group," *Commun. ACM*, no. October, pp. 1–9, 2010.

[17] P. Salza, P. Musmarra, and F. Ferrucci, "Agile Methodologies in Education: A Review," *Agil. Lean Concepts Teach. Learn.*, no. January, 2019.

[18] H. Takeuchi and I. Nonaka, "The New New Product Development Game: Stop Running the Relay Race and Take Up Rugby," *Harv. Bus. Rev.*, vol. 64, no. 1, pp. 137–147, 1986.

[19] K. Schwaber, "SCRUM Development Process," *Bus. Object Des. Implement.*, no. February 1986, pp. 117–134, 1997.

[20] D. Leffingwell, *Agile Software Requirements: Lean Requirements Practices for Teams, Programs, and the Enterprise*. Upper Saddle River, NJ: Addison-Wesley Professional, 2010.

[21] M. Cohn, *User stories applied: For agile software development*. Boston: Addison-Wesley Professional, 2004.

[22] J. Patton and P. Economy, *User story mapping: discover the whole story, build the right product*. Sebastopol, CA: " O'Reilly Media, Inc.," 2014.

[23] W. Lynch, "Scrum: INVEST in Good Stories by Achieving SMART Tasks," *Medium*, 2019. [Online]. Available: https://medium.com/@warren2lynch/scrum-invest-in-good-stories-and-smart-tasks-63f8432f7840. [Accessed: 22-Apr-2020].

[24] J. C. Marques, "MACRE-SAR: Um Modelo Ágil para Especificação de Requisitos de Software em Ambientes Regulados," Instituto Tecnológico de Aeronáutica, 2016.

[25] A. Gkritsi, "Scrum Game: An Agile Software Management Game," p. 96, 2011.

[26] A. Krivitsky, "Scrum Simulation with LEGO Bricks," *LEGO4SCRUM*, 2011.

[27] C. G. Von Wangenheim, R. Savi, and A. F. Borgatto, "SCRUMIA - An educational game for teaching SCRUM in computing courses," *J. Syst. Softw.*, vol. 86, no. 10, pp. 2675–2687, 2013.

[28] A. D. De Souza, R. D. Seabra, J. M. Ribeiro, and L. E. D. S. Rodrigues, "SCRUMI: A Board serious virtual game for teaching the SCRUM framework," *Proc. - 2017 IEEE/ACM 39th Int. Conf. Softw. Eng. Companion, ICSE-C 2017*, pp. 319–321, 2017.

[29] J. M. Fernandes and S. M. Sousa, "PlayScrum - A Card Game to Learn the Scrum Agile Method," in *2010 Second International Conference on Games and Virtual Worlds for Serious Applications*, 2010, pp. 52–59.

[30] W. L. Lee, "SCRUM-X: An interactive and experiential learning platform for teaching Scrum," *7th Int. Multi-Conference Complexity, Informatics Cybern. IMCIC 2016 7th Int. Conf. Soc. Inf. Technol. ICSIT 2016 - Proc.*, vol. 2, pp. 192–197, 2016.

[31] W. C. Wake, "Scrum from Hell - Developed for the Scrum Gathering in Denver," *XP123*, 2004. [Online]. Available: https://xp123.com/articles/scrum-from-hell/. [Accessed: 05-May-2020].

[32] B. Gloger, "The Ball Point Game," *Scrumology*, 2008. [Online]. Available: https://scrumology.com/from-the-archives-the-ball-point-game/. [Accessed: 05-May-2020].

[33] M. Paasivaara, V. Heikkilä, C. Lassenius, and T. Toivola, "Teaching Students Scrum Using LEGO Blocks," *36th Int. Conf. Softw. Eng. ICSE Companion 2014 - Proc.*, pp. 382–391, 2014.

[34] J. P. Steghöfer, H. Burden, H. Alahyari, and D. Haneberg, "No silver brick: Opportunities and limitations of teaching Scrum with Lego workshops," *J. Syst. Softw.*, vol. 131, pp. 230–247, 2017.

[35] J. de Castro Martins *et al.*, "Using big data, internet of things, and agile for crises management," in *Information Technology-New Generations*, Las Vegas: Springer, 2018, pp. 373–382.

[36] A. Delhij, R. van Solingen, and W. Wijnands, "The eduScrum Guide: the rules of the game," p. 21, 2015.

[37] F. Cruz, *Scrum e Agile em Projetos: Guia Completo*, 2nd ed. Rio de Janeiro: Brasport, 2018.

[38] Ministério da Educação, "Diretrizes Curriculares - Cursos de Graduação em Engenharia," *Portal do Ministério da Educação*, 2019. [Online]. Available: http://portal.mec.gov.br/escola-de-gestores-da-educacao-basica/323-secretarias-112877938/orgaos-vinculados-82187207/12991-diretrizes-curriculares-cursos-de-graduacao. [Accessed: 30-Jan-2020].

Lineu Alves Lima Filho is a Professor of Computer Science at the Federal Institute of São Paulo (IFSP), campus São José dos Campos/SP. He has more than 20 years of experience in the educational field. He holds a degree in Computer Science from the University of Taubaté (1997) and a degree in Education from CEETEPS (2016). He worked as a Microsoft Certified Trainer (MCT) for 7 years. His research interests include software engineering, database systems, agile methods, agile teaching methods, scrum, project-based learning, game-based learning, and instructional design.

Paulo Marcelo Tasinaffo is graduated in Mechanical Engineering from the Federal University of Itajubá/MG (UNIFEI, 1996), obtaining a Master's degree in Mechanical Engineering from the same Institution in 1998. In 2003, he received a Doctorate in Space Engineering and Technology from the National Institute of Space Research (INPE) in Brazil. He is currently a full professor of the Aeronautical Technological Institute (ITA) in São José dos Campos/SP also in Brazil, with experience in Aerospace and Computer Engineering and with emphasis on Artificial Neural Networks. Prof. Tasinaffo works mainly in the following subjects: modeling of non-linear dynamic systems, computational mathematics, artificial intelligence, expert systems, intelligent agents, neural control structures, evolutionary computation, and stochastic processes.