IT Talent Shortage: Strategies to Mitigate a Blackout

Ana Carolina Moises de Souza
Norwegian University of Science and Technology
Trondheim, Norway
ana.c.m.de.souza@ntnu.no

Sheila Reinehr
Pontifícia Universidade Católica do Paraná
Curitiba, Brazil
sheila.reinehr@pucpr.br

Andreia Malucelli
Pontifícia Universidade Católica do Paraná
Curitiba, Brazil
andreia.malucelli@pucpr.br

Tayana Conte
Universidade Federal do Amazonas
Manaus, Brazil
tayanaconte@gmail.com

Sabrina Marczak
Pontifícia Universidade Católica do Rio Grande do Sul
Porto Alegre, Brazil
sabrina.marczak@pucrs.br

Letizia Jaccheri
Nowergian University of Science and Technology
Trondheim, Norway
letizia.jaccheri@ntnu.no

Fabio Binder
Pontifícia Universidade Católica do Paraná
Curitiba, Brazil
fabio.binder@pucpr.br

Tanara Lauschner
Universidade Federal do Amazonas
Manaus, Brazil
tanara@icomp.ufam.edu.br

Daniela Cruzes
Nowergian University of Science and Technology
Trondheim, Norway
daniela.s.cruzes@ntnu.no

ABSTRACT

The global Information Technology (IT) market was valued at approximately $3,110 million in 2022, marking a 7.4% increase over the previous year. Of this total, around 56% pertained to the software and services sector. This growth poses a global challenge for IT organizations due to a significant gap between the increasing demand for IT skills and the availability of qualified professionals. Currently, about 87% of organizations worldwide are experiencing the effects of this skills gap, hindering business operations. Traditional undergraduate computing programs often require more time than the current market demands can afford. To address this issue, universities and companies are developing innovative educational strategies. Our goal is to share the experience of implementing four distinct cases, three in Brazil and one in Norway, so that other universities and companies can adopt similar strategies to alleviate the IT skills shortage. Therefore, this paper presents strategies for the preparation of IT professionals to mitigate the impacts of the IT labor shortage. Each case includes an experience report on the context, structure, benefits, hard and soft skills developed, and lessons learned. Finally, we summarize the contributions of these four cases by reflecting on their differences and similarities, followed by a summary of the strategies that proved effective in collaborating with companies.

KEYWORDS

IT professional development, IT personnel shortage, software professionals shortage, university and industry partnership, innovation in software engineering education

1 INTRODUCTION

In 2022, the global Information Technology (IT) market reached a value of approximately 3,110 million dollars. Of this amount, around 56% is attributed to the software and services market, indicating a growth of 7.4% compared to the previous year [13]. To support this growth, an increasing and unmet demand for people qualified in technology has been a challenge for businesses [20]. The technology skills gap is already impacting around 87% of organizations worldwide, i.e. the shortage of qualified people in technology is hampering the running of businesses [27].

The Winter 2022 Fortune/Deloitte CEO Survey [16] revealed that approximately 71% of CEOs worldwide consider the shortage of professionals with specific skills to be the most significant business disruptor. The Fall edition of the same survey [15] shows that 94% of executives believe the IT workforce crisis is even more critical when it involves specialized positions. Many companies expect ready-to-work professionals that meet an extensive list of requirements, but this expectation often diverges from reality [8]. In Software Engineering, there is a demand for high-quality professionals, and this appears to be one of the causes of the shortage in the field [21].

In the Brazilian context, a study by BRASSCOM (Association of Information and Communication Technology and Digital Technology Companies) forecasts that there will be a gap of more than 797,000 job vacancies without qualified labor by December 2025. This same study also pointed out that universities country-wide had the capacity to graduate about one-third of this expected number, hence the concern with an eminent IT shortage [6].

Cico et al [12] argues that while Software Engineering Education has made significant progress in preparing students for their careers, this alone does not solve the overall problem of having too few IT professionals worldwide. Regular undergraduate computing courses take an amount of time that does not align with current needs, and project scenarios are often oversimplified, lacking correspondence to real-world projects. Additionally, as industry demands constantly change, higher education institutions are challenged to respond promptly to both current and future demands [24].

Currently, it takes an average of 66 days to fill a technology-related vacancy. According to the U.S. Bureau of Labor Statistics, software development will face the challenge of around 200,000 hard-to-replace vacancies every year. Contributing factors to this challenge, as outlined by Lehoufa [25], include the increasing numbers of retiring developers, unequal and limited access to Computer Science education, the "great resignation" related to the Covid-19 pandemic, skills mismatch between academia and industry, and immigration disruptions.

In addition, as described by Dong et al. [17] considering external factors such as the rapid development of technologies in the IT sector and the dynamism of the job market, employers are really actively seeking students with up-to-date knowledge, the ability to use new technologies, and transferable skills. Our future IT professionals, who are currently students, need to witness and experience the complexity and challenges inherent in developing real-world applications.

To address these challenges, universities and companies are collaborating and developing innovative educational approaches. This paper presents four different cases to enhance the preparation of IT professionals and minimize the impacts of what has become known as the IT labor shortage. The described cases showcase the hard and soft skills developed during the programs, along with their strategies, related benefits and lessons learned. These cases can assist other universities and companies in implementing initiatives to overcome the professional shortage.

The remainder of this paper is organized as follows. Section 2 outlines the educational cases which result of industry-university collaborations. Section 3 engages in a discussion on the cases relevance. Finally, Section 4 concludes the paper with a summary, limitations, and directions for further research on the topic of our experience.

2 CASES

In this section, we describe each case by providing its context, course structure, benefits, and lessons learned. A summary of the cases and strategies is also presented.

2.1 AGES

Context. Pontifical Catholic University of Rio Grande do Sul in Brazil founded the Software Engineering undergraduate course in 2014 based on two-folded reasons. First, the university is surrounded by an innovation ecosystem mainly focused in IT-based startups. The on-campus Technology Park, TECNOPUC, with over 7 thousand job positions and 1,200 companies, encourages the synergy between academia and industry in software engineering education and training. IT companies often seek for help to organize or support on-boarding and on-the-job training of novice software engineers, to design and deliver continuing education programs to keep senior software engineers up-to-date with new technologies, and also to design and offer extra-curricular training of software engineering students such as industrial hackathons and bootcamps. Second, it was also created as a response to the challenges identified by the Brazilian Federal Government report, [6] aforementioned.

The Software Engineering undergraduate course aims to graduate professionals with solid knowledge in Computing, Mathematics and Production, capable of developing software to solve complex problems, capable of analyzing and selecting technology, prepared to integrate software solutions, and able to identify new business ideas and develop innovative software-based solutions. The course was designed inspired on the ACM 2005 Computing Curricula [35] and on the national-wide Software Engineering reference curricula and guidelines [38]. It is composed of 8 terms organized into 4 years of study, and of 30 hours of classes a week.

Among the diverse pedagogical strategies adopted in the course design, such as integration with the Computer Science Graduate Program, remote classes and 120 hours of mandatory extra-curricular activities (including internship, research assistant ship and scientific conference attendance), there is the Experimental Agency in Software Engineering, called AGES from hereon.

Program Structure. AGES main goal is to provide the Software Engineering undergraduate students with the opportunity to experience situations similar to those ones lived by industry IT professionals, such as working in teams, delivering software on-time with quality, and managing conflicts. The students work in groups, supervised by a teacher, to develop a functional prototype to customers. Candidate customers attend a call for projects managed by the AGES coordinator, a senior Software Engineering professor with almost 20 years of experience in industry as a project manager. Project proposals can vary from software development to support organizational process improvement, software evolution, development of new ideas, and proof of concept. Projects are selected based on the complexity of the expected solution, availability of the stakeholders, and awareness that the delivery is not a commercial product but rather a functional prototype that will allow the students to put in practice their theoretical learning. Once the project is selected, the AGES coordinator and the architect (a full-time hired professional) together conduct a Design Sprint session to elicit the intended scope and candidate user requirements. Each project has the duration of an academic term (4.5 months) and follows a Scrum-like development framework organized into 15-days long sprints. Teams are comprised of 10-15 students, often self-organized into two squads (cross-functional teams responsible for a certain feature or set of features) [23]. Each project team includes students from the 4 expected roles: unitary and functional tester, developer, developer leader and/or software architect, and Scrum master. Each role is played by a student enrolled in one of the 4 Agency courses. For instance, a student playing a software architect role needs to be enrolled in the AGES course level 3, a third year course. These roles were defined intending to promote that the student puts into practice the knowledge acquired up-to-the-moment in her program, integrating content from several courses. For instance, the student playing the developer role (level 2, a second year student) will perform requirement elicitation and specification and user experience design tasks, from the respective Requirements Engineering and User Experience courses. Each AGES course requires 162 hours per term to be completed, organized into 9 hours per week (3 hours in person, as part of the classwork, and 6 hours extra-class).

Project Development Practices. Once the student is allocated to a certain project and the team is formed when the school term starts, the students are welcomed to the Agency, introduced to the AGES Scrum-like development framework, and assessment and grading schema. They are next introduced to the project goal and stakeholders. Their first task of this Set-Up sprint is to understand the project scope and define and negotiate with the stakeholders the technologies to be used. If necessary, the Agency architect will help the team to study these technologies, tutoring during the process.

Next, at the Warm-up sprint, the team needs to revisit the previously elicited user requirements, specify them, and design mockups to validate the specified user requirements with the stakeholders. Once these requirements are approved, the team needs to design the software architecture and organize the product backlog and tasks. From now one, 5-6 development sprints are conducted including coding, unit and functional testing, and integration testing. At the end of each sprint, there is a sprint review in which the stakeholders approve the developed requirements and a retrospective session to identify what can be improved. The last sprint is dedicated to tackle technical debt and prepare the final delivery to the customer (e.g., code packing, documentation writing), including the final presentation to all Agency project teams.

Each team needs to follow the time-bound Scrum-like framework but is free to decide on the development practices and artifacts. For instance, a certain team might opt to specify the user requirements using behavior-driven development scenarios [37] or UML use cases, both techniques introduced in the Requirements Engineering course, or any other specification technique of their choice, including none. The teams are expected to revisit their decisions during the sprint retrospective and take action to continuously improve their practices adoption.

Performance Assessment and Portfolio Development. The student performance assessment is comprised of several assignments, namely: periodic progress status reports, detailed reports, one-to-one meetings with the teacher, and participation at the Agency retrospective.

The periodic progress status report needs to be handed out at the end of each sprint stating the student's contribution to the project development following a set of pre-defined self-evaluation criteria. These periodic reports are input for two detailed reports delivered by the student; one at mid-term and one at the end of the term. They include a consolidated description of the student's contributions and a critical analysis of how their hard skills have evolved. The students also participate twice during the term in a half-an-hour one-a-one meeting with the teacher in charge of the project to reflect upon their soft skills and career development, mirroring what companies often offer to their employees–a long-term view on how current job actions and behavior are contributing to one's career development. Last but not least, the Agency retrospective seeks for feedback on how the Scrum-like process, performance assessment, project selection, infrastructure, and other aspects can be improved, offering the student a chance to reflect upon an organizational level (rather than a project level or individual level).

Lessons Learned. Inspecting the history of the AGES' project documentation and the students' final reports we identified that, in average, the teams deliver most of the agreed features and the stakeholders express to be satisfied with the functional prototype. The delivered software mostly follows programming good practices and standards discussed in Programming courses and integration testing is the mostly left behind activity.

- Team Challenges: Quite often the students point out their difficulties on on-boarding level 1 students and addressing action plans defined during the sprint retrospective. Communication is among the most-difficult-to-deal issue for diverse reasons, e.g., different working hours extra class and maturity of individuals to grasp the relevance of a timely and clear communication. And picking technologies that team members are no experts on to work is among the decisions teams most regret.
- Stakeholders Participation: A few projects in AGES history report about the lack of participation from the stakeholders and of infrastructure to implement the designed software solution (i.e., no AWS licenses).
- Skills Development: Time management, tasks prioritization, and conflict management are also among the cited skills developed during the AGES courses participation. They say that they realize that a good professional needs to excel in hard and soft skills and that the teachers' mentorship helps them deal with job hunting anxiety and with building trust in themselves.

Overall, the students report that AGES offers them the opportunity to experience working collaboratively in teams, learn how to provide feedback, and commit to attending the customer expectations.

2.2 Customer-Driven Project Course

Context. The software engineering course, known as Customer-Driven Project, was established at the Norwegian University of Science and Technology in 1974. Over the years, approximately 3650 students have completed this course. The Customer-Driven Project Course aims to equip students with software engineering hard and soft skills by participating in collaborative group work. This involves applying skills acquired in previous semesters to a real-world development projects. Following a Project-Based Learning methodology [36] [3], the Customer-Driven Project Course encompasses all stages of a software development project, covering project management, pre-study, requirements analysis, design, programming, testing, and documentation. The main result and benefit is the delivery of a minimum viable product (MVP) to the selected companies [28]. In this course, a variety of Samsungizes and sectors are selected.

Course Structure. At the start of every calendar year, the course coordinator invites companies, also called customers, to submit proposals. In recent years, these proposals comprise questions regarding project definition, its connection to sustainability and diversity, technical and managerial contacts, and customer expectations for the project. Following submission, the teaching assistants team, composed of current PhD students, conducts a peer review of the proposals. During this peer review, the technical viability, details of the project description, solution innovation, and contribution to society are evaluated. When the selection process ends, the customer receives a notification of whether or not they have been accepted.

Before the school second semester begins (the Fall term), the teaching team forms student groups, each with a maximum of 7 students. This process is crucial, as groups are structured based on backgrounds and experiences, aiming to mix those with and without Artificial Intelligence knowledge, for example. The student groups are carefully curated to provide a well-rounded student experience, ensuring a balance of gender, country of origin (national or international), and technical expertise within each group.

The official start of the Customer-Driven Project Course iteration with students occurs in the Fall of each year and lasts three months. Each group is randomly assigned to a customer proposal. On the first day, an Introductory Lecture is held, where students and customers are invited to learn about the course requirements. This provides an opportunity for the teaching team, students, and customers to get to know each other and start the project. Meetings, communication tools, intellectual property, and expectations are decided collaboratively between students and customers.

Each student spends around 20 hours a week for this course, with a dedication of 350 hours. In this course, students can experience diverse aspects of the software development process. For instance, students should explore different roles and transition between software engineering specialties. In the course itself, students can assume different positions, such as group leaders, social activity facilitators, and sustainability experts. Group leaders participate in bi-weekly meetings with the teaching team to align on the course structure, share experiences, and monitor group progress.

Development Practices. The practices in this course aim to enhance students' software engineering, project management and cooperation skills through practical scenarios [17]. The challenges include decision-making for large systems, collaborative problem-solving, task delegation, adaptation to non-ideal conditions, and effective documentation. Students have the freedom to choose their development methodology, often opting for Agile methodologies [19]. The course guides students through a preliminary study, requiring a comprehensive understanding of the problem, describing the current system, and exploring potential solutions. It emphasizes writing detailed requirements specifications aligned with business requirements. The course also guides students in designing a modular software architecture, and currently integrating artificial intelligence tools, implementing software security, and testing practices. Previous programming knowledge is crucial. Depending on the customer proposal, students may need to acquire new programming languages. The last phase involves external documentation, including user and installation guides.

Performance Assessment. At the semester's end, students must submit a project report, source code, and presentation. The final project report covers reporting on customer communication, customer feedback, diversity management, sustainability practices, the role of artificial intelligence in the project and suggestions for course improvement. Self-reflection is encouraged to discuss any issues affecting the project. The external examiner judges the ultimate grade, considering the feedback from customers and supervisors. The evaluation comprises five categories:

- Product (30%): Assessing how well the group met project objectives, project usability, and project value to the customer.
- Team Dynamics (20%): Assessing role rotation, collective decision-making, and problem resolution within the team.
- Project and Process (20%): Evaluating the planning and management of scope, time, communication, resources, and risks, along with adherence to processes.
- Documentation (15%): Assessing the project report quality, logical flow, format, structure, writing style, and readability.
- Innovation (15%): Evaluating considerations for emerging technological trends, diversity dimensions, sustainability, and students' critical reflection and innovative documentation solutions.

The Customer-Driven Project Course focuses on hands-on experience in all software development phases, emphasizing project organization, execution, effective documentation, and real-world customer presentation skills. The course provides a diverse learning environment, incorporating group work, supervision meetings, presentations, and a few lectures.

Interdisciplinary topics like Diversity and Sustainability in software engineering are recent additions to the course curriculum, illustrating ongoing efforts made yearly to introduce interdisciplinary topics, supported by materials and workshops for students and customers addressing societal challenges [30],[11],[14].

Lessons Learned. Through these 50 years of this course existence much we have learned and improved, we describe the lessons learned as following:

- Advancements in Software Engineering Education: From following advances in technology stacks and software development process, to integrating interdisciplinary topics within software engineering practices, this course is often improved by the input and market requirements of our industry partners. The case presented here is the ultimate version of this course.
- Customer Feedback and Enhancements: Since we started to collect mid-term customer feedback we learned that: a) customers has limited time to offer in this course, avoid overwhelming meetings is necessary; b) customer might not be able to dedicate time to learn about interdisciplinary topics, so they expect the students to share knowledge with them; c) most of the customer that submit their proposals for the first time, submit afterwards as well.
- Proposals Scope: Regarding the scope of proposals, we have been facing challenges with those that propose solutions solely based on artificial intelligence models rather than software engineering projects. Out of the 18 proposals received this year, at least 8 suggested solutions that require knowledge of Artificial Intelligence models. We are learning to balance projects with students' backgrounds and course expectations as well as integrate both expertise in the course.
- Soft skills Development: Students are expected to exercise soft skills. A demanding characteristic is for students to be pro active, that means not rely on the supervisor to solve their problems with the customer. As supervisors and courses coordinators, we are still learning how to equip and make the students confident to enhance their skills on this matter.

Students overall satisfaction about this course is over 4 points (in a scale from 1 to 5 in which 1 means very unsatisfied and 5 means very satisfied).

2.3 Apple Developer Academy

Context. Aiming to expand the training of developers for its own application ecosystem, Apple has established partnerships with universities worldwide to implement an environment called the Apple Developer Academy. The case reported here is implemented in collaboration with the Pontifical Catholic University of Paraná in Brazil.

Since 2013, the two institutions have established a partnership to offer students the opportunity to develop skills and competencies to produce applications to be made available at the Apple virtual store. In its most recent format, the Apple Developer Academy takes place in two years and welcomes 50 students in each cohort. By 2024, 400 students have successfully completed the Apple Developer Academy.

Program Structure. The physical environment is designed to facilitate communication, learning, and creativity in the format of a software studio [7]. The software studio is a concept borrowed from architecture and industrial design as a response to the perceived gap of skills between what undergraduate courses deliver and what industry demands. The central idea behind the studio is derived from the École des Beaux-Arts and the Bauhaus [34], based on reflective practice and the use of mentoring instead of traditional lectures.

The teaching method used in the Apple Developer Academy is Challenge-Based Learning (CBL), inspired on the Problem-Based Learning method defined to support medical education [41], which encourages students to develop technical and behavioral skills while performing challenges and reflective activities. This method has been proven effective in teaching mobile software development skills [5]. Nichols et al. [31] explain that CBL is an environment for investigative reflection on teaching and learning, a flexible framework with multiple entry points, a nonproprietary scalable model, a framework that focuses on global challenges with local solutions, an authentic connection between academic disciplines and real-world experience, a framework for developing 21st-century skills, a process that places the students as responsible for their learning, and requires students to develop and deploy solutions in a real-world environment.

CBL is organized into three phases: Engage (Big Idea, Essential Question, and Challenge); Investigate (Guiding Questions, Guiding Activities and Resources, and Analysis); and Act (Solution, Implementation, and Evaluation). Students are encouraged to reflect on their learning through reflexive practice at every step of the way.

The challenges are designed to provide growing difficulty levels throughout the Apple Developer Academy. Some are performed individually, while others are in groups, as illustrated in Figure 1. Smaller circles represent shorter challenges and larger circles represent more extended challenges. Mentors help students navigate through their learning path along the program. The mentors are specialists in technology, programming, game design, software engineering, soundtrack, etc.

Nano challenges are short-term challenges performed individually, while the others are group activities. As can be supposed, a business challenge focuses on the business side of the market. Besides developing technical and soft skills, the business challenge aims to develop marketing and sales skills through a monetization step. Students receive a small amount of money to promote their apps. They create the app, publish it in the app store, and monitor the acquisition of users, and the revenue numbers. This skill is essential to the app development market.

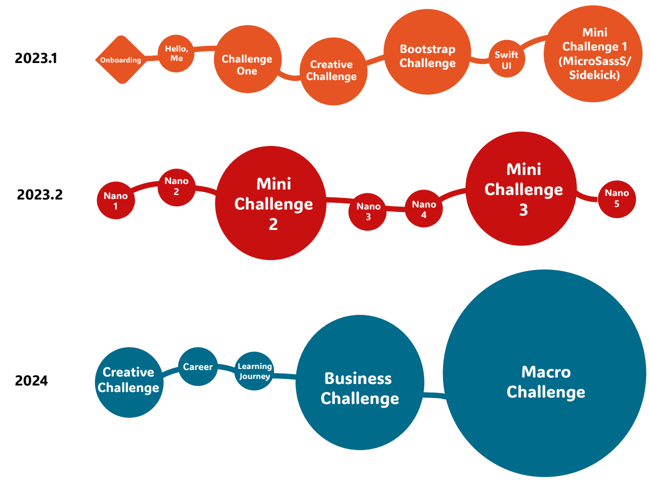

Figure 1: The Apple Developer Academy two-year program structure.

Application and Selection Process – A Critical Success Factor. One of the program's critical success factors is the recruitment process. Each cohort receives about 600 applications, of which 50 students are selected to participate in the program.

To join the Apple Developer Academy, applicants undergo a four-stage selection process. First, the applicants must send a video, telling what makes them unique or the best idea they have ever had, and their portfolio, which can be composed of real-world products or mock-ups. Afterward, they have a written exam in which they can choose from a programming exam, a design exam, or both. The third step is a two-day hackathon, where they are initially paired to select an idea for a product and then are rotated among the groups until they finish the challenge. One interesting feature of this step is that the group that presents the solution is not the same group that starts with the idea. By the end of the third stage, applicants are narrowed to about 100 people. The last step is a 30-minute interview with the project coordinator. After these four steps, the coordinator and the mentors meet to analyze all the results and select the 50 students that will participate in the program.

This selection process is critical to ensure skills that fit the Program goals and format. In the first cohort (2013-2014), 27 students had to be dismissed from the Apple Developer Academy because they did not demonstrate the expected profile and background for the activities. In the 2015-2016 cohort, with the selection process improved, only two students were dismissed. Since the process was changed (to the 4 stage process previously described), only five people have been dismissed from the program for poor performance in 8 years-time.

Each of the selected students receives a laptop, a smartphone, and a monthly stipend. At the end of the program, they can keep their equipment for personal use.

Close Monitoring, Feedback, and Reflections – Another Critical Success Factor. Another critical success factor is the assessment process. During the program, students are constantly monitored and mentored. They receive a task (challenge) and have limited time to perform. Mentors are right away available to answer questions and to help define the best course of action for completing the challenge. The assignments are not graded once the monitoring and feedback processes are continuous.

Another critical point is using reflective practice during the learning process [18]. Reflective practice can occur in four formats: reflection-in-action, knowing-in-action, reflection-on-action, and reflective conversations with the material. Dors et al. [18] explain that *reflection-in-action* is the reflective form of knowing-in-action, which means, the reflection during the problem-solving process. In the reflection-in-action process, doing and thinking are complementary since doing extends thinking in the tests, moves, and probes of experimental action, and reflection feeds on doing and its results. Once the problem-solving process is finished, a practitioner may consider what she could have done differently or would do differently next time. This is called *reflection-on- action* or reflecting on experience. It can take place after the fact or during a pause, during the action, in both cases. The reflection has no direct connection to present action. *Knowing-in-action* is the knowledge built into and revealed by one's performance of everyday routines of action. It sometimes is labeled as intuition, instinct or motor skills. Finally, *conversations with the material* refers to the process of evolving understanding developed through the design of the app and its artifacts. Transformation on how to solve the problem emerges through the discussion about and learning from designed artifacts.

A longitudinal study conducted over the three most recent cohorts [1] shows that reflective practice helps develop the hard and soft skills demanded by the app development market. The following soft skills were observed: communication, active listening, conflict management, decision-making, collaboration, motivation, creativity, self-confidence, and synthesis. The students also learned the following hard skills: code reuse-oriented development, software architecture, database development, app prototyping using different platforms such as MIRO, FIGMA, or Adobe Package, programming logic, algorithm foundations and programming, object-oriented programming, and Swift.

Our experience with this initiative has shown us that it helps develop the profile of the 21st-century professional, as well as helps to improve the ecosystem of applications for mobile devices, aiming to reduce the labor shortage in this sector. Over 50% of the graduated students are currently working in developing mobile applications of all sorts as per our most recent survey on their career status.

Lessons Learned. In Apple Developer Academy we had the following lessons learned:

- Student Fit and Selection Process: Not all students are well-suited for this program. The selection process continues to evolve to identify candidates who are motivated, self-manageable, resilient, and responsible. This ensures that the students selected are more likely to succeed and benefit from the program's unique challenges and learning methods.
- Interdisciplinary Learning: Students from computer science and design backgrounds often have differing interests, with computer science students typically less interested in design, and design students less interested in programming. However, both groups benefit from exposure to the other discipline. At a minimum, they gain a basic understanding that facilitates better collaboration. Some students even develop a professional level of expertise in the secondary discipline.
- Challenge Based Learning (CBL) and Knowledge Gaps: One of the primary weaknesses of CBL is the potential for knowledge gaps due to the absence of a traditional syllabus. This issue is mitigated by having students engage in a variety of challenges, each building on the previous ones, thereby progressively filling in these gaps and reinforcing their learning.
- Comprehensive Business Challenge: Understanding the entire process of app development, publishing, and monitoring is a significant advantage provided by the business challenge. Although not all students enjoy the post-development phases, they universally acknowledge the value of this experience. Many consider it the richest and most informative part of their academic journey.
- Collaboration and Team Dynamics: Having students work with different peers for each challenge enhances their collaborative skills. This practice not only fosters teamwork but also prepares students for real-world professional environments where working with diverse teams is essential.

2.4 SUPER Project

Context. A different case is occurring in another university. The SUPER Project is a collaboration between Federal University of Amazonas and Samsung, aimed at fostering training and research in technological fields at Federal University of Amazonas. This project was initiated in response to a significant demand for skilled professionals in the fields of Informatics and Engineering, particularly to drive development and innovation within the region. Federal University of Amazonas is situated in a unique region known for its diverse natural environment. In an effort to conserve this rich natural environment, the government encourages the growth of the technological industry as an alternative to traditional extractive and livestock farming practices. Nevertheless, there is a pressing need to augment the number of IT professionals within the region.

The SUPER Project distinguishes itself by involving not just one, but a total of eleven distinct undergraduate programs, spanning across two separate campuses. On the main campus, a diverse array of programs partake in the project, including Computer Science, Electrical Engineering, Computer Engineering, Software Engineering, Production Engineering, and Design. Meanwhile, on the second campus, the courses are Information Systems, Production Engineering, and Software Engineering.

While Federal University of Amazonas offers various informatics and engineering courses, numerous undergraduates encounter challenges in completing their programs due to financial and technical constraints. Conversely, Samsung and several research and development institutes are in need of professionals with expertise in specific areas. This project seeks to address these dual challenges by supporting undergraduates in completing their courses and simultaneously preparing future professionals for the specific areas of focus at Samsung.

The Program Structure. The SUPER Project offers 140 courses opportunities in technological fields and has been designed to cover the entire duration of the student's stay at the university. Each part of this project is called Work Packages (WP), ranging tasks from basic training to specific and advanced levels. This allows students to complement their current studies in addition to the university's regular courses. The design of this project enables master's and doctorate students to collaborate and share their knowledge among the participants of the project. The breakdown of the WPs is as follows:

- WP1 - Common Basic Training: Covers basic knowledge common to the courses involved in the project, such as mathematics, physics, technical English, and socio-emotional skills.
- WP2 - Specific Basic Training: Training and improvement courses in the fundamental principles of the fields of Computer Science, Engineering, and Design.
- WP3 - Advanced Training: Opportunities for participation in qualification and innovation actions in strategic areas, leading to high-impact scientific and technological development.

In the context of Computer Science and Software Engineering courses, more specific tasks are applied on WP2 and WP3 that involve students at different levels:

- WP2 tasks – This group of tasks has a significant impact on the connection of novice students (first two years) with undergraduate programs in Computer Science and Software Engineering. The literature demonstrates that this is the most critical period for a student's retention in the university. To achieve this goal, it is necessary to implement a set of pedagogical actions, which include: 1) organizing orientation courses for newcomers with the purpose of minimizing deficiencies from high school education; 2) providing improvement courses to complement subjects with higher failure rates in the initial periods of these programs; 3) promoting scientific and technological initiation in projects of low complexity but that stimulate curiosity and interest in the field of Computing; 4) generating and sharing scientific knowledge about computer education; and 5) adapting existing physical spaces to create an interactive learning environment with practical activities, challenging projects, and technologies that spark student interest. The WP2 tasks include:
  - Scientific and technological initiation in Computing (ICT Jr.)
  - Peer instruction in CS courses - 7 instructors and 10 teaching assistants to assist students in the more challenging computing courses.
- WP3 tasks – The significance of these tasks lies in the need to complement the academic education of senior students (last two years) in Computer Science and Software Engineering undergraduate programs with emerging content demanded by the Information Technology job market. To address this gap, this WP aims to provide training courses where students can learn about emerging technologies in the field of Computing and apply them in scientific research projects. These WP3 tasks encompassing three different programs, each with a distinct focus:
  - Engineering Innovative Software Products
  - Machine Learning Models Applied to Health Care
  - Web Development Applied to Educational Tools

Project Impact and Achievements. In the first three years of its existence, this project has offered 4840 hours of courses that have impacted more than 3000 students. We have also achieved one of our goals, which was to reduce the dropout rate by 50%, compared to the 2018 figures, in each undergraduate course involved in the project. There has been a 50% increase in the total number of graduates compared to the cumulative total from 2014 to 2018, reaching a total of 771 students. While this project is generating an increasing number of competent graduates, it also plays a role in reducing the disparity between demand and supply in the IT job market. Achieving these goals fills the gap in the widespread shortage of IT professionals on a global scale [13] [27].

Annually, the project hosts a week of activities open to the community. During the event, students present their projects, engaging locals in technology and inspiring new participants. This event has positively impacted 1500 people over the past two years.

Scientific contributions have also been highlighted, providing opportunities for students to be part of the national and international scientific academic landscape. More than 300 reports on research and technological development have been delivered by the students.

Lessons Learned. In these four years of SUPER project, we have observed the significant impact of the project on the University as a whole, and specifically on many undergraduate students. Due to the SUPER project, these students have become prouder of their courses and the opportunities available in the industry.

- Financial Support Benefits: The project not only equips students for new positions in the industry but also allows them to be full-time students during specific periods, as they receive sufficient funding to cover their personal expenses (and sometimes support their families). They know they don't need to rush into accepting a position in the industry at the "wrong time" due to financial pressures. The project has allowed them to choose the right time to transition to the industry, and they know they are more qualified.
- Course Design Insights: Regarding the training courses, we learned that: (a) Sometimes, a new technology that is highly hyped will not endure, so we need to choose the technologies for advanced courses carefully; (b) Advanced courses should provide a strong foundation in a specific subject, not just focus on the latest technologies; (c) Often, students need better training in intermediate courses to be able to learn advanced subjects effectively.
- Student Selection Criteria: Regarding better student selection, we learned that (a) The selection process should not only consider technical aspects, such as course grades; (b)

Table 1: Cases Summary

Cases	AGES	Apple Developer Academy	Customer-Driven Project Course	SUPER Project
Methodology	Active Learning [10]	Challenge Based Learning [5]	Project-based Learning with real customers [3]	Active Learning [10]
Structure	648 hours in total divided in 4 terms	1200 hours of technical specialized knowledge beyond the formal academic syllabus	350 hours of a course integrated within the Computer Science academic syllabus	980 hours of complementary courses to enrich the existing academic syllabus
Strategies	Real-world software engineering selected projects from various industry sectors	Industry support for learning their specific technology	Real-world software engineering projects in various industry sectors	Industry incentives for research and academic infrastructure
Benefits	1) Opportunities for students to experience the same living environment as in the industry	1) Increase the number of specialized professionals 2) Continuous student development.	1) Students partnership and networking with companies 2) Exposure to industry-specific challenges.	1) Increase the number of graduated students and scientific development 2) Funding research projects with practical applications.
Graduated	2160 students	400 students	3650 students	3000 students

Interviews that explore the reasons behind a student's candidacy for a specific task help us better understand the student's context and motivations; (c) Discussing the focus of WP3 tasks in some classes and linking the course content with the task objectives helps students gain a better perspective of the WP3 Task, resulting in better candidates.

Table 1 outlines the methodologies, structures, strategies, and benefits associated with three distinct cases: AGES, Apple Developer Academy, Customer Driven Project Course, and SUPER Project. The data includes insights into the number of graduated students and the benefits on academy and industry collaboration.

3 DISCUSSION

In this section, we discuss the strategies, similarities and differences between the cases, the implications for software engineering education practice and research, as well as future work.

3.1 The strategies

We aimed to share the experiences of conducting four distinct cases in four different institutions in Brazil (three cases) and Norway (one case). Each case has demonstrated its unique contribution to reducing the IT shortage. The benefits and challenges of the strategies to mitigate the IT shortage are discussed as following:

3.1.1 Brazilian Governmental Support. Policies and laws that incentive the IT industry can attract more talent and investment, potentially addressing shortages. Programs like Brazilian's laws called "Lei do Bem" [32] and "Lei de Informatica" [33] aim to stimulate the IT sector by supporting professional training, encouraging startups, and fostering partnerships between academia and industry. These initiatives, supported by fiscal incentives for Research and Development (RD), have shown promise in enhancing technological advancement, similar to efforts seen in other OECD (Organisation for Economic Co-operation and Development) countries [29].

3.1.2 Industry Support. The aforementioned Brazilian laws allow industry to invest in RD in general. By choosing to invest in educational programs companies help to leverage individuals to pursue IT quality education and careers [40], contributing to reduce the IT professional shortage. In Norway, the companies volunteers by submitting the proposals to be developed in the course.

3.1.3 Industry Partnership on Real-world Proposals. Partnering with industry allows students to work on real-world technological proposals, providing valuable experience and potentially increasing the pool of high skilled IT professionals [12]. Examples like AGES and the Customer-Driven Project Course illustrate how working on real-world projects enables students to apply their skills, interact with industry professionals, and gain practical experience. This approach exposes students to industry-specific challenges and tools, thereby enhancing their employability.

3.1.4 Industry-Academia Collaboration. Building trusting relationships between universities and industry partners can facilitate knowledge exchange, internship opportunities, and tailored programs that address industry needs, ultimately helping to bridge the IT skills gap. These collaborations extend beyond individual projects. It establishes long-term partnerships that benefit both academia and industry while also ensuring a continuous flow of skilled graduates and cutting-edge research.

3.1.5 Program Development. Designing programs integrated into students' graduation requirements and offering financial support for further education can incentive more individuals to pursue and remain in the IT field.

3.2 Similarities and Differences

The four cases - AGES, Customer Driven Project, Apple Academy, and SUPER Samsung - share similarities in their goals of providing students with practical, hands-on experience in software development and addressing the IT shortage through specialized training. They all aim to bridge the gap between academia and industry, preparing students for real-world challenges and increasing their employability. They emphasize the importance of collaboration between students and industry partners, fostering relationships and exposure that enhance students' career prospects. The following paragraphs brings a recap of the cases:

AGES program aims to produce professionals capable of developing software solutions for complex problems, analyzing and selecting technology, integrating software solutions, and generating innovative software-based solutions. The program spans 4 terms over 4 years, with 9 hours of weekly classes (3 hours in person and 6 hours extra-class). AGES provides students with hands-on experience akin to industry professionals, allowing them to work in teams, deliver software on time with quality, and manage conflicts. It also selects project proposals that vary from software development to support organizational process improvement, software evolution, development of new ideas, and proof of concept.

Customer-Driven Project has contributed to the local ecosystem of industry and academia for the past 50 years in Norway. It actively involves students with the industry, serving as an exemplary model for courses that integrate real software development problems [36]. For many students it is their first project and for companies it is an opportunity to test an idea, develop an MVP or improve their current solutions. Participating in real-world projects establishes relationships and exposure for students, increasing their chances of being hired not only by the clients involved, but also by other companies. This study [40] supports the effectiveness of real-world projects in software engineering education, confirming that 88% of participants started or changed careers after completing the program.

The Apple Academy refers to a specialization in the company's specific technology. It is crucial due to its complexity and the limited resources available for learning about the company's technology. The Apple Academy is executed worldwide by other universities and is recognized for creating opportunities for students to develop applications and solutions based on this company's technology [2]. From the perspective of addressing the IT shortage, this project enables students to work on application development, open startups, and contributes to an increase in the number of specialized professionals in the company's technology.

SUPER Samsung is a broader case that encompasses course programs within Federal University of Amazon. It is supported by a company interested in increasing the number of IT professionals in the region. SUPER Samsung is organized to benefit students in continuing their undergraduate courses, while offering basic, specific and advanced training beyond the regular course program. Another benefit is related to scientific contributions. For example, research projects such as a technique to support the design of Internet of Things solutions [39] and the creation of a tool to detect related posts on developer forums [26] are funded by SUPER Samsung. These research projects create opportunities for students to diversify their careers in industry and academia.

It is important to note the differences between the cases. AGES focuses on training complete professionals, capable of developing innovative software solutions, with a wide range of project proposals. These proposals are developed in 4 terms in which the student experiences a role in the software development process. The Customer Oriented Project, on the other hand, offers students opportunities to work on real-world projects for a semester focusing on a selected customer proposal. Apple Academy specializes in technology from a specific company outside of the current degree program, offering students unique opportunities to develop applications and solutions based on that technology. SUPER Samsung, in turn, offers a broader program within a university environment, supported by a company that aims to increase the number of IT professionals in the region by providing additional training beyond regular course programs and funding research projects that benefit both industry and academia.

3.3 Implications for SEE practice and research

While efforts in collaboration between industry and academia have occurred at many universities and across various program levels, there is still a need to reemphasize the implications of these collaborations for software engineering education [22]. Carver and Prikladnicki highlighted the importance of engaging with companies to understand their real-world problems and identify the Industry-Academia Collaboration (IAC) patterns being utilized. They also advised industrial practitioners to seek out researchers and students, as they may be able to assist in solving their issues [9].

Guttorm et al demonstrated that project-based learning is successful in reinforcing students' learning goals and turning them from theoretical concepts to applicable realities [36]. Students feel motivated and engaged by the idea that the project will be used. In the case of the Apple Academy, there is also an opportunity to develop its own applications and innovate solutions for society. Hands-on experience is truly necessary and still in demand.

Programs like the Apple Academy offer specialized training in specific technologies or industry domains. This highlights the importance of offering niche programs to cater to the diverse interests and career goals of students, as well as the evolving needs of the software industry.

Initiatives like SUPER Samsung, which fund research projects benefiting both industry and academia, emphasize the importance of collaboration between academia and industry in advancing software engineering research. Such collaborations can lead to the development of innovative solutions and technologies that address real-world challenges.

In summary, all cases offer a laboratory for research into different phenomena linked to software engineering education across various topics. Studies have been conducted to understand learning methodologies' outcomes for students [4], students' perceptions of new topics [39], interdisciplinary integration [14],[11],[30], trending topics in the field [12], and reflective practices [18],[1].

3.4 Future work

In terms of measuring the actual impact of the presented cases, future work for this study involves conducting in-depth analysis of the current employment status of alumni. Our goal is to conduct research to understand how these cases contributed to students securing employment after completing their courses.

Further research on the impact of partnerships with industry partners is also underway. We aim to investigate the critical success factors from the industry's perspective in terms of their business and the employment opportunities provided to these students.

Regarding academic research opportunities, ongoing projects are exploring how the methodologies applied in the cases help students achieve their learning goals. Additionally, we are examining how emerging topics such as sustainability, diversity, and artificial intelligence are perceived by both students and industry partners. All four cases have proven to be excellent arenas for research and innovation

4 FINAL CONSIDERATIONS

Aiming to share cases that provide strategies to mitigate the IT Talent Shortage, this paper offers insights into how the industry and academia can form diverse types of partnerships. The partnerships presented here covers strategies such as: a) industry support for learning their specific technology; b) real-world software engineering projects in various industry sectors; c) industry incentives for research and academic infrastructure. By providing details on the context, structure, and outcomes of each case, we hope to motivate other universities and the IT industry to develop similar strategies.

Despite efforts to address the IT professional shortage, challenges persist. The cases described in this article, although promising, may not be universally applicable. The effectiveness of these strategies may be influenced by regional variations, institutional resources and specific needs of the industrial partner. Furthermore, the evolving nature of technology and industry demands requires continuous adaptation, and the success of these programs may depend on continuous evaluation and improvement. To further address the IT professional crisis, future research should focus on the scalability and adaptability of cases presented to diverse educational institutions and industries. Longitudinal studies can assess the sustained impact of these initiatives on students' career trajectories and the needs of the IT industry. Exploring new collaborative models and frameworks that bridge the gap between academia and industry could contribute to a more dynamic and responsive educational ecosystem.

Furthermore, research into the role of emerging technologies, such as artificial intelligence and virtual reality, in improving IT education and preparing students for real-world challenges is crucial. Understanding the impact of global events, such as the Covid-19 pandemic, on the IT job market and identifying strategies to mitigate disruptions can inform future education and workforce planning.

In conclusion, addressing the IT Talent shortage requires a multifaceted approach that involves collaboration between universities and industry, innovative educational programs, and ongoing research to adapt to the evolving technology and employment needs. The cases presented provide insight and serve as starting points for further exploration and development in this area.

5 ACKNOWLEDGMENTS

We would like to thank the opportunity to share academic and industry initiatives during the Software Engineering Practices and Experiences Exchange between Norway and Brazil, and also for providing financial support through the Norwegian Directorate for Higher Education and Skills - UTFORSK. The Brazilian authors Tayana Conte, Sabrina Marczak, Andreia Malucelli, and Sheila Reinehr would like to thank CNPq for the financial support (Bolsa de Produtividade process no. 314797/2023-8, 313181/2021-7, 313975/2021-3, and 314064/2021-4, respectively). The SUPER project is the result of the Research and Development (RD) project 001/2020, signed with Federal University of Amazonas and FAEPI, Brazil, which has funding from Samsung, using resources from the Informatics Law for the Western Amazon (Federal Law nº 8.387/1991), and its disclosure is in accordance with article 39 of Decree No. 10.521/2020. The authors would like to thank the Apple Developer Academy for the opportunity to conduct the research.

REFERENCES

[1] Aline Andrade, Alessandro Maciel Schmidt, Tania Mara Dors, Regina Albuquerque, Fabio Binder, Dilmeire Vosgerau, Andreia Malucelli, and Sheila Reinehr. 2023. Education, Innovation and Software Production: The Contributions of the Reflective Practice in a Software Studio. *The Journal of Software Engineering Research and Development* 11 (2023), 7:1 – 7:12. Issue 7.

[2] Apple. 2021. Coders, Designers, and Entrepreneurs Thrive Thanks to Apple Developer Academy. https://www.apple.com/newsroom/2021/05/coders-designers-and-entrepreneurs-thrive-thanks-to-apple-developer-academy

[3] Brigid Barron, Daniel L. Schwartz, Nancy Vye, Allison L. Moore, Anthony J. Petrosino, Linda K. Zech, and John D. Bransford. 1998. Doing with Understanding: Lessons from Research on Problem- and Project-Based Learning. *The Journal of the Learning Sciences* 7 (1998), 271–311.

[4] Fabio Vinicius Binder, Regina Albuquerque, Sheila Reinehr, and Andreia Malucelli. 2020. Innovation and active learning for training mobile app developers. In *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering: Software Engineering Education and Training* (Seoul, South Korea) *(ICSE-SEET '20)*. Association for Computing Machinery, New York, NY, USA, 151–161. https://doi.org/10.1145/3377814.3381713

[5] Fabio Vinicius Binder, Mark Nichols, Sheila Reinehr, and Andreia Malucelli. 2017. Challenge based learning applied to mobile software development teaching. In *2017 IEEE 30th Conference on Software Engineering Education and Training (CSEE&T)*. IEEE, Savannah, GA, USA, 2017, 57–64.

[6] Brasscom. 2023. Demanda de Talentos em TIC e Estratégia TCEM. https://brasscom.org.br/pdfs/demanda-de-talentos-em-tic-e-estrategia-tcem/

[7] Christopher N. Bull, Jon Whittle, and L. Cruickshank. 2013. Studios in Software Engineering Education: Towards an Evaluable Model. In *Proc. Int'l Conf. on Software Engineering*. IEEE, San Francisco, USA, 1063–1072.

[8] Peter Cappelli. 2019. Your Approach to Hiring Is All Wrong. https://hbr.org/2019/05/your-approach-to-hiring-is-all-wrong

[9] Jeffrey C Carver and Rafael Prikladnicki. 2018. Industry–academia collaboration in software engineering. *IEEE Software* 35, 5 (2018), 120–124.

[10] Michael Christie and Erik de Graaff. 2017. The philosophical and pedagogical underpinnings of Active Learning in Engineering Education. *European Journal of Engineering Education* 42 (2017), 16 – 5.

[11] Orges Cico, Letizia Jaccheri, and Anh Nguyen Duc. 2021. Software Sustainability in Customer-Driven Courses. In *2021 IEEE/ACM International Workshop on Body of Knowledge for Software Sustainability (BoKSS)*. IEEE, Madrid, Spain, 15–22.

[12] Orges Cico, Letizia Jaccheri, Anh Nguyen-Duc, and He Zhang. 2021. Exploring the intersection between software industry and Software Engineering education - A systematic mapping of Software Engineering Trends. *Journal of Systems and Software* 172 (2021), 110736. https://doi.org/10.1016/j.jss.2020.110736

[13] Associação Brasileira das Empresas de Software. 2023. *Mercado Brasileiro de Software: Panorama e Tendências 2023*. Technical Report. ABES, São Paulo, Brasil.

[14] Ana Carolina Moises de Souza, Kaban Koochakpour, Sofia Papavlasopoulou, and Letizia Jaccheri. 2024. Exploring Social Sustainability Alignment in Software Development Projects. In *Proceedings of the 19th International Conference on Evaluation of Novel Approaches to Software Engineering - Volume 1: ENASE*. INSTICC, SciTePress, Angers, France, 250–261. https://doi.org/10.5220/0012739600003687

[15] Deloitte. 2022. Fall 2022 Fortune/Deloitte CEO Survey. https://www2.deloitte.com/content/dam/Deloitte/us/Documents/us-ceo-survey-overview-fall-2022.pdf

[16] Deloitte. 2022. Winter 2022 Fortune/Deloitte CEO Survey. https://www2.deloitte.com/content/dam/Deloitte/us/Documents/about-deloitte/us-fortune-deloitte-CEO-survey-winter-2022-highlights-final.pdf

[17] Dapeng Dong, Robbie Butler, and J. Herbert. 2022. Evaluation of the effectiveness of developing real-world software projects as a motivational device for bridging theory and practice. *Journal of Further and Higher Education* 46 (2022), 1275–1289.

[18] Tania Mara Dors, Frederick MC Van Amstel, Fabio Binder, Sheila Reinehr, and Andreia Malucelli. 2020. Reflective Practice in Software Development studios: findings from an ethnographic study. In *2020 IEEE 32nd conference on software engineering education and training (CSEE&T)*. IEEE, Munich, Germany, 1–10.

[19] Henry Edison, Xiaofeng Wang, and Kieran Conboy. 2022. Comparing Methods for Large-Scale Agile Software Development: A Systematic Literature Review. *IEEE Transactions on Software Engineering* 48 (2022), 2709–2731.

[20] European Commission. 2023. European Year of Skills 2023. https://commission.europa.eu/strategy-and-policy/priorities-2019-2024/europe-fit-digital-age/european-year-skills-2023_en

[21] Vahid Garousi, Gürkan Giray, Ebru Tuzun, Cagatay Catal, and Michael Felderer. 2019. Closing the gap between software engineering education and industrial needs. *IEEE Software* 37, 2 (2019), 68–77.

[22] Vahid Garousi, Dietmar Pfahl, João M Fernandes, Michael Felderer, Mika V Mäntylä, David Shepherd, Andrea Arcuri, Ahmet Coşkunçay, and Bedir Tekinerdogan. 2019. Characterizing industry-academia collaborations in software engineering: evidence from 101 projects. *Empirical Software Engineering* 24 (2019), 2540–2602.

[23] Henrik Kniberg. 2014. Spotify engineering culture (part 1). https://engineering.atspotify.com/2014/03/spotify-engineering-culture-part-1/

[24] Marco Kuhrmann, Joyce Nakatumba-Nabende, Rolf-Helge Pfeiffer, Paolo Tell, Jil Klünder, Tayana Conte, Stephen G. MacDonell, and Regina Hebig. 2019. Walking Through the Method Zoo: Does Higher Education Really Meet Software Industry Demands?. In *Proc. of the Int'l Conf. on Software Engineering: Software Engineering*

Education and Training. IEEE, Montreal, QC, Canada, 1–11.

[25] Meryam Lehoufa. 2023. The Global IT Talent Shortage: How to Prevent This Growing Crisis from Affecting Your Business. https://www.redaktcms.com/blog

[26] Márcia Lima, Igor Steinmacher, Denae Ford, Evangeline Liu, Grace Vorreuter, Tayana Conte, and Bruno Gadelha. 2023. Looking for related posts on GitHub discussions. *PeerJ Comput. Sci.* 9 (2023), 1–38.

[27] McKinsey. 2021. Five Fifty: The skillful corporation. https://www.mckinsey.com/capabilities/people-and-organizational-performance/our-insights/five-fifty-the-skillful-corporation

[28] Maleknaz Nayebi, S. Kabeer, G. Ruhe, Chris Carlson, and Francis Chew. 2017. Hybrid Labels Are the New Measure! *IEEE Software* 35 (2017), 54–57. https://doi.org/10.1109/MS.2017.4541048

[29] Thomas Neubig, Fernando Galindo-Rueda, Silvia Appelt, Chiara Criscuolo, and Matej Bajgar. 2016. Fiscal incentives for R&D and innovation in a diverse world. https://www.oecd-ilibrary.org/content/paper/5jlr9stckfs0-en

[30] Anh Nguyen-Duc, Letizia Jaccheri, and Pekka Abrahamsson. 2019. An empirical study on female participation in software project courses. In *2019 IEEE/ACM 41st International Conference on Software Engineering: Companion Proceedings (ICSE-Companion)*. IEEE, Montreal, QC, Canada, 240–241.

[31] M. Nichols, K. Cator, M. Torres, and D. Henderson. 2016. *Challenge Based Learning User Guide.* Digital Promise, Redwood City, CA.

[32] Government of Brazil. 2023. Lei do Bem. https://www.planalto.gov.br/ccivil_03/_ato2004-2006/2005/lei/L11196compilado.htm

[33] Planalto. 2019. Lei n.º 13.969, de 26 de dezembro de 2019. https://www.planalto.gov.br/ccivil_03/_ato2019-2022/2019/lei/l13969.htm

[34] D. A. Schön. 1983. *Reflective Practitioner: How Professionals Think in Action.* Basic Books Inc., New York.

[35] Russell Shackelford, Andrew McGettrick, Robert Sloan, Heikki Topi, Gordon Davies, Reza Kamali, James Cross, John Impagliazzo, Richard LeBlanc, and Barry Lunt. 2006. Computing Curricula 2005: The Overview Report. *SIGCSE Bull.* 38, 1 (mar 2006), 456–457.

[36] Guttorm Sindre, Michail Giannakos, Birgit R. Krogstie, Robin Munkvold, and Trond Aalberg. 2018. Project-Based Learning in IT Education: Definitions and Qualities. *Uniped* 41, 2 (2018), 147–163.

[37] John Ferguson Smart and Jan Molak. 2023. *BDD in Action: Behavior-driven development for the whole software lifecycle.* Manning, Shelter Island, New York.

[38] Sociedade Brasileira de Computação (SBC). 2023. Currículos de Referência. https://www.sbc.org.br/documentos-da-sbc/category/131-curriculos-de-referencia

[39] Eriky Souza, Carla Façanha, Conceição Rocha, Max Barbosa, Sabrina Souza, Genildo Gomes, Tanara Lauschner, Bruno Gadelha, and Tayana Conte. 2023. IoThinking: A Design Thinking technique to support the design of IoT systems. In *Proceedings of the XXXVII Brazilian Symposium on Software Engineering* (Campo Grande, Brazil) *(SBES '23)*. Association for Computing Machinery, New York, NY, USA, 458–463. https://doi.org/10.1145/3613372.3613422

[40] Caio Steglich, Anielle Lisboa, Rafael Prikladnicki, Sabrina Marczak, Michael da Costa Móra, Alejandro Olchik, Nelice Heck, Yasser Rachid, and Guilherme Ghidorsi. 2020. Agile Accelerator Program: From Industry-Academia Collaboration to Effective Agile Training. In *Proceedings of the XXXIV Brazilian Symposium on Software Engineering* (Natal, Brazil) *(SBES '20)*. Association for Computing Machinery, New York, NY, USA, 21–30. https://doi.org/10.1145/3422392.3422485

[41] Diana Wood. 2003. Problem-Based Learning. *The BMJ* 326, 7384 (2003), 328–330.