Perceived Benefits and Challenges of Learning Startup Methodologies for Software Engineering Students

Jorge Melegati
Free University of Bozen-Bolzano
Bolzano, Italy
jmelegatigoncalves@unibz.it

Rafael Chanin
PUCRS, School of Technology
Porto Alegre, Brazil
rafael.chanin@pucrs.br

Xiaofeng Wang
Free University of Bozen-Bolzano
Bolzano, Italy
xiaofeng.wang@unibz.it

Afonso Sales
PUCRS, School of Technology
Porto Alegre, Brazil
afonso.sales@pucrs.br

Rafael Prikladnicki
PUCRS, School of Technology
Porto Alegre, Brazil
rafaelp@pucrs.br

ABSTRACT

The need of skills other than technical from software developers is becoming evident. The DevOps movement is an example of that applied to operational tasks. Startup development methodologies focus on business activities in innovative organizations. Several universities offer courses based on these methodologies to software engineering students, mainly to improve their creativity, problem solving, and business skills. This paper investigates how software engineering students learned startup development methodologies and discusses what are the challenges and benefits in their learning process. We conducted a multi-method study in three different universities. The data was collected in two phases and analyzed using thematic analysis. Our study reveals that students realized the importance of collaboration with other courses and the importance of user involvement in development. However, students tend to over-simplify concepts, trying to adapt them to what they are familiar with. The results indicate the necessity of business education for technical students and directions for improvements.

CCS CONCEPTS

• **Social and professional topics → Software engineering education**.

KEYWORDS

Software Engineering Education; Lean Startup; Empirical Study

ACM Reference Format:
Jorge Melegati, Rafael Chanin, Xiaofeng Wang, Afonso Sales, and Rafael Prikladnicki. 2019. Perceived Benefits and Challenges of Learning Startup Methodologies for Software Engineering Students. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education (SIGCSE '19), February 27-March 2, 2019, Minneapolis, MN, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3287324.3287382

1 INTRODUCTION

Nowadays, business education is becoming so relevant that some universities across the world have already recognized the importance of providing entrepreneurial skills to engineering students [27]. The fact is that being technically competent is not enough. Understanding customers' needs and knowing how to deliver real value to people is a must in this new economy. Even though some initiatives can be perceived in the academic world, the lack of business education to software engineering students has been a noted issue for a long time. Back in 2000, Shaw [32] already observed that software engineering students rarely faced nontechnical issues. In 2009, a survey of 119 Spanish students reported that they were not receiving enough training in innovation and entrepreneurship [14]. Several courses have focused on entrepreneurship for software engineering students in the last years [13, 19] and other studies focused on the importance of business concepts for technical students [20].

Some entrepreneurship courses teach emerging methodologies for startup development like Lean Startup [19, 24, 35] and Customer Development [9]. Fitzgerald and Stol [15] argue that these methodologies are consequence of the need of a closer integration between business needs and development. However, despite of consisting interesting concepts and ideas, these methodologies can be difficult to comprehend and to apply in practice [6], which may hinder software engineering students to grasp business skills by following the courses. To the best of our knowledge, this is a problem that has not been investigated in the literature. Therefore, this study focuses on the following research question:

What are the benefits and challenges of software engineering students learning the Lean Startup approach?

To answer the research question presented, we performed a multi-method study composed of two parts. Firstly, we studied two different teams of computer science students learning by applying the Lean Startup methodology to develop a new product or service. A set of benefits and challenges in the learning process perceived by the students were identified. Secondly, we conducted a survey to a group of students from a mobile application development course that also taught Lean Startup and Customer Development concepts to further examine some specific challenges discovered in the case study. We analyzed data in both parts using thematic analysis, a technique for identifying, analyzing and reporting patterns (themes) in qualitative data [33]. The results indicate that students

SIGCSE '19, February 27-March 2, 2019, Minneapolis, MN, USA
© 2019 Association for Computing Machinery.
ACM ISBN 978-1-4503-5890-3/19/02…$15.00
https://doi.org/10.1145/3287324.3287382

understand the importance of collaboration with other courses, such as business development and marketing, and the importance of user involvement in development. However, they tend to simplify concepts, trying to adapt them to what they are familiar with, sometimes losing the main concepts behind the principles.

The remaining part of the paper is organized as follow: Section 2 presents the background on fundamental concepts. Section 3 reviews related work. Section 4 describes the research design, Section 5 presents the results, which are further interpreted and discussed in Section 6. Finally, Section 7 depicts our conclusion and final thoughts.

2 BACKGROUND

In this section, we discuss the definition of a startup and present two methodologies startup companies use: Lean Startup and Customer Development.

2.1 Startup definition

According to Blank and Dorf [4] a startup is a temporary organization that is searching for a sustainable and repeatable business model. Eric Ries [28] defines a startup as "*a human institution designed to create a new product or service under conditions of extreme uncertainty*". All startups share similar characteristics [34]: suffer from internal and external pressure; have limited resources; are young and immature; address dynamic technologies and markets; and present high market risk.

Regarding software development processes for startups, several strategies and approaches have been tested, but no significant results have been achieved [11, 34]. Since the most important goal of a startup is to find a sustainable business model before running out of resources, methods such as Customer Development [4] and Lean Startup [28] were proposed in order to address this objective. The following subsections present these methodologies' main concepts.

2.2 Customer Development

Steve Blank [3] created the Customer Development process based on the premise that most startups fail from a lack of customers rather than product development issues. He argues that there are several processes to manage product development, but there is no formal process to manage customer development.

The Customer Development process is divided into four phases:

- *Customer Discovery*: state and validate both the problem and the customer hypotheses. A startup should only move to the next phase once there are evidences that a group of customers are willing to pay to solve a given problem;
- *Customer Validation*: develop and test a sales process. If the startup is not able to validate the business model, it has to go back to the discovery phase. This move is called *pivot*;
- *Customer Creation*: once the business model is validated, it is time to grow the customer base in order to validate the business feasibility;
- *Company Building*: put formal management into place and create growth strategies to scale the business.

If a startup is not able to validate its business model before running out of resources, it will not succeed. Hence, it is very important to perform an effective customer discovery process [2]. This can be done by interviewing potential customers, using marketing strategies, such as developing landing pages or online advertisements to validate a given value proposition, or even by manually delivering the product/service to the customer (the concierge approach).

2.3 Lean Startup

Inspired by the Customer Development and Lean Manufacturing, Eric Ries proposed the Lean Startup. The method has the premise that every startup is a set of experiments to test hypotheses the startup makes about a given business idea. In order to do so, startups build minimum viable products (MVPs) [23].

An MVP is used to collect feedback from potential customers in order to identify and to model the next experiment [18]. One of the common issues with this approach is to understand what "minimal" and "viable" means [21]. The development of an MVP should focus on the smallest effort in order to test a given assumption of the business model and to obtain maximum validated learning.

Despite of the Lean Startup popularity, it is an approach proposed in a non-scientific book based on anecdotal evidences. Nevertheless, according to Frederiksen and Brem [16], there is substantial evidence in scientific literature to support the Lean Startup efficacy.

3 RELATED WORK

Chanin et al. [10] performed a systematic mapping study on software startup education and derived several interesting insights on how the Lean Startup principles are perceived by students. For instance, one of the reviewed studies (Rioja Del Rio et al. [29]) suggested the use of Business Model Canvas (BMC) [26] when working on software startup projects. The idea is that it allows students to analyze the big picture (all aspects of a business model). Otherwise they would just focus on technical issues. BMC entails the value proposition of the business, the customer segment, the channels to reach the customers, the relationships between them, key resources, key activities, key partners, revenue streams and cost structure.

Barbe [1] proposed a model that connects all aspects of a startup development: from the basic technical knowledge to the business acceleration and funding. The motivation for this model was businesses created by technical founders. If these founders lack business and/or soft skills, they will most likely fail or will need to find people with these skills. Hence, students not only learn the technical foundations for developing a software startup, but they are also exposed to the whole startup development process.

Zaina and Álvaro [38] proposed a methodology that combined user-centered design [30] and Lean Startup in order to foster innovation and entrepreneurial behavior in a software engineering course. The authors argued that computer related courses usually focused only on technical issues, and they did not instigate students to be creative and innovative. They conducted two case studies to verify the method effectiveness. The results indicate that students not only learned important business concepts, but also perceived the importance of understanding customers' needs.

Finally, Buffardi et al. [8] argued that it was very hard to emulate real world projects in an academic environment. When students work with "toy" projects, they end up learning technical content, but they do not experience real customer pressure, competition, and other "real life" situations. This study also brought in light (from

Nurkkala and Brandle [25]) the gaps between software engineering students' and industrial software engineers' experiences. They are sixfold: real product versus a project; long duration versus short duration; low turnover versus high turnover; high complexity versus low complexity; needs maintenance versus no maintenance; and real customers versus no customers. Hence, a methodology was proposed in order to minimize these gaps. The idea was to promote collaboration between software engineering and entrepreneurship students, who would act as customers. Even though software engineering students reported that the experience was relevant to them, the whole process just mimics a real project context. It is not ideal, but it gave students a good perception about what it took to develop a real startup. In this kind of situations, instructors need to evaluate the trade-offs. Depending on the characteristics of a course, it may be too difficult to address real projects.

In regards to challenges students face as well as benefits from courses on Lean Startup concepts, as far as the authors are concerned, there are no academic studies being undertaken.

4 RESEARCH DESIGN

In order to answer our research question, we followed a multimethod study design composed of two parts: (1) a multiple case study conducted in two sites, and (2) a survey applied to software engineering students in a third site.

4.1 Data collection

The first part consisted of a multiple case study [37] based on two cases within different educational configurations:

Case A. Two students in the end of a Computer Science course in a Brazilian university employed Lean Startup and agile methodologies in their bachelor thesis project. The group initially consisted of three students but after a few months one student dropped out and the first author acted as a mentor. The project consisted of building a website to inform students, professors and employees about events occurring in the university campus. It took approximately nine months from March to November 2016. Nonetheless, the effort was not equally distributed along the period.

Case B. Four students, one from a Master and three from a Bachelor program in Computer Science from an Italian university participated in the optional Lean Entrepreneurship course taught by the third author. The setup of the course was project-based, following a learning-by-doing style (the course was described in a previous paper [35]). They developed a project based on the business idea of their own: build a platform to connect car owners keen to rent their cars to possible renters when they were not using them. The course lasted four months, from October 2017 to January 2018.

We collected data mainly in the end of each project. For Case A, students developed a monograph (thesis) to be presented for evaluation that described all the tools used as well as the project history. The monograph had 65 pages written in Portuguese. Only the second half (starting from page 25) were valuable to this study: the remaining consisted of techniques and tools description used in the software development. The team described the product development in three iterations. Each iteration description has the following sections: building, diffusion, metrics and learning. We also included the emails exchanged between the mentor and the team. An initial set consisted of 150 emails had a team member as the sender or receiver. A subject analysis revealed that 10 of them were not related to the project. Most of them were about deadlines and meetings arrangements. We analyzed those related to the difficulties encountered by team members.

For Case B, the course ended with a retrospective session conducted by the teacher (the third author) in which the group depicted the project history. The first author also participated in this session as an observer. During the retrospective, students were asked to draw the journey during the course on an A0 paper sheet, using whatever format or notion that they deemed appropriate. Then they were asked to explain the drawn diagram. The retrospective session took more than one hour, and in the first nine minutes the team discussed internally what to drew. We recorded, and later transcribed, all conversations and discussions during the retrospective session.

The second part of this study consisted of a survey undertaken in a 2-year iOS application development course in Brazil. The class was composed of 40 students with strong software engineering background and 10 design students specialized in user experience for mobile applications.

During the course, students conducted various software development projects, classified as small tasks, nano and mini-challenges. At the end of the course, students worked on a final software development project (a big-challenge activity) which had a six-month duration. They always worked in groups (between three and six students) composed of developers and at least one designer.

The survey was performed following the guidelines proposed by Wohlin et al. [36]. Since this study focuses on software engineering students, we did not apply the survey on the design students. Therefore, the survey was sent to 40 individuals.

All students that participated in this survey have already received lessons on Lean Startup and Customer Development and have practiced and applied these concepts in various activities. The survey was intended to further explore the challenges identified in our multiple case study. In order to do so, we decided to present a small scenario that asked students how they could contribute to a project. The scenario is described as follows:

> *"An entrepreneur friend of yours comes to you with a project idea that he had not yet implemented. He wants to create an app/system that connects elderly people with technology teachers. His idea is to help people to get into the digital world (learn how to use an email, Facebook, Instagram, etc). At first, the business model would be to charge the elderly for the class and keep a commission (between 5% and 10%) and pass the remaining to the teacher. You like the idea and join the project. Before leaving, you received a task: describe, in your opinion, what and how you can help. The idea is to present the next steps and (given your expertise) how you are going to help this project."*

The idea behind this approach was to leave the floor open to students to develop their thoughts freely; we believe that if we had mentioned explicitly that we were working on a survey about Lean Startup/Customer Development, students would have been induced to answer accordingly. Students were asked to write down their answers in a text editor and send them to one of the authors.

4.2 Data analysis

The data analysis conducted in both parts employed mainly thematic analysis, following the scheme proposed by Braun and Clarke [7]. Although suggested to psychological studies, this approach has been commonly used in Software Engineering research [12].

In the first part, we labeled relevant pieces of data and initial themes emerged. The labeled excerpts were then grouped together. Through a comparison between them, we made some improvements. In addition, we conducted a cross-case analysis to compare the two cases related to their setup and results according to the themes identified in each individual case. In the second part, we analyzed the survey responses under the themes identified in the case studies.

5 RESULTS

This section presents our results. Section 5.1 presents findings from the multiple case study: a detailed description of each case and a cross-case analysis. Section 5.2 displays the survey results.

5.1 Multiple-case study

Case A. In this case, students never mentioned a business model for their project. Although Lean Startup can be useful to non-profit organizations, there was no concerns about how to support the project financially. One possibility that students mentioned in emails were that the project could substitute the pre-existent university events page. In this sense, also in the emails, there was a concern from the mentor about why users should switch to the new product. Regarding design and user experience, in the thesis, the text mentioned that suggestions made during first iteration focused on design aspects and still in the third iteration: *"most of the critics [...] were related to the experience provided by the site and that it was not attractive enough"*. Finally, for marketing and distribution, the difficulties were users acquisition and engagement. The first was present in each interaction description and they also wrote: *"with acquired learning, it was clear the importance of time spent in marketing"*. The latter was present in an email from one of team members: *"engagement: one of the challenges is to make the user to come back and give her interaction ways that make her want to come back every week to see all the events"*.

In the thesis, students made a conceptual error: they used the word "hypothesis" to describe a list of features to "create an interface". They confused hypothesis testing with building incrementally. A possible explanation is that the Build-Measure-Learn cycle first step is to formulate hypothesis, so they felt obligated to write a hypothesis while reporting this interaction. Nevertheless, they followed a cyclical process as described in their thesis (Figure 1).

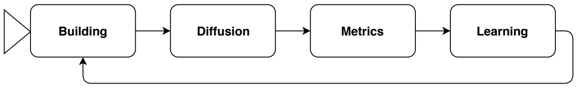

Figure 1: A schematic view of Group A's thesis.

It is interesting to show pieces of data where students highlight their learning: *"that demonstrated the importance of keeping development always in touch with the final user"* or *"the importance that the client represents in the process"*. Additionally, in their conclusions, they stated that: *"not always the concept about a project or a product idea by developers point of view is, in fact, a need to the target users"*.

Case B. From a business perspective, the second group mentioned that the mentor drew attention to the need of a specific insurance to make the product viable. Although such insurance was fundamental to the product viability, the group had already started developing software architecture. Regarding design and UX, they said that they still needed help to understand *"what the consumer wants"*. They also mentioned that they were not able to create an interesting logo that could enhance product adoption.

The tendency to follow a straight path is clear in the picture the second group drew (Figure 2). Although classes in the course advocated the Build-Measure-Learn cycle from the Lean Startup methodology [28], the depicted flow still recalls the waterfall process. The group also mentioned in the retrospective while discussing the insurance: *"we started the implementation of the website after the form. We didn't think about it"*. During the retrospective, the team also made a conceptual error: when asked what an MVP was, they replied: *"MVP is just the minimum set of features we should put in our platform"*. Instead, according to Lean Startup, MVP is the minimum set of features one should implement to test a hypothesis that may not even be in the final product.

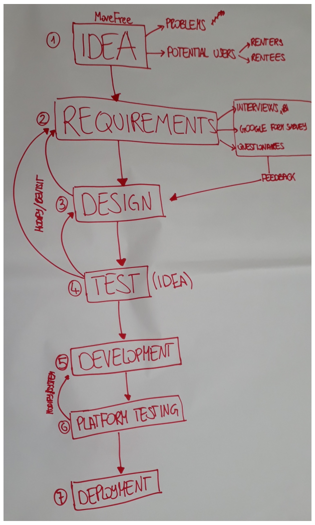

Figure 2: Flow draw by the second group.

As takeaways from the course, this group mentioned: *"what the consumer wants is more important than you implementing"* and *"if it does not put value to the market, there's no sense to making it"*.

Cross-case analysis. Although students had similar team sizes and backgrounds, they followed really different processes. The first group performed a cyclical process trying to formulate hypothesis and to validate them, that is, following a Build-Measure-Learn cycle. They also focused on customer feedback that comprehends a good part of the methodology [17]. Meanwhile, the second group basically followed a straight plan (having the idea, gathering requirements, designing and testing the idea, development, platform testing and deployment), even after participating in classes about the Lean Startup methodology. In software startups, Giardino et al. [18] already saw this plan driven approach as waste and a reason for startups' failure. Meanwhile, Bosch et al. [6] argue that agile and lean practices are better for early stage software startups.

A possible explanation is that first group students was more engaged in learning the methodology or really developing a viable product. Another possibility would be that they are better students than the second group. Even if these hypotheses are correct, an element should have been more important to guide a group: the mentor. For the second group, the mentor had an important act telling them about the insurance issue. However, students did not mention anything else about her performance. For the first group, the first author acted as a mentor highlighting the importance of customer feedback and hypothesis testing, guiding the students to focus on them. The second group mentioned another interesting point: *"it's different from the other courses: you're given [a task] and deliver [it]; that is, there's no market connection".*

There were, though, some similarities between the two cases. First, neither teams worried about business issues: Case A students did not come up with a clear business model and Case B students ignored concerns about insurance and continued developing the technical solution. Of course, the academic environment may be the reason of such insouciance about business. Second, both teams recognized the importance of validation and customer feedback.

Thematic analysis identified 11 themes categorized in the two categories: *Benefits* and *Challenges*, as described in Table 1.

Table 1: Themes found in first phase.

Category	Theme
Benefits	Business learning Design learning Feedback importance Marketing learning Success based on metrics Team building learning User experience learning
Challenges	Business perspective ignored Conceptual errors

The benefits include students learning regarding different areas:

- **Design:** for instance, all suggestions in the first iteration of Case A concerned design issues;
- **Team building:** like a student in Case B mentioned: *"people from different backgrounds can give different opinions"*;
- **Marketing and distribution:** as Case A thesis states: *"with the acquired knowledge it is clear the importance of time dedicated to distribution"*;
- **Business:** Case A thesis states: *"not always the idea of determined project or idea from the perspective of developers is, in fact, a necessity of targeted users".*

Besides that, several excerpts show the importance of user feedback during projects development. For instance, in Case A's students wrote: *"Although the alerts page visits had a peak after launching, the feature was abandoned by users, generating almost no visits. This shows that possibly it wouldn't be interest to invest in its development."*

The challenges comprises the avoidance to tackle business issues as mentioned earlier, the tendency to follow a straight plan focusing on technical challenges and difficulty to understand some concepts, like MVP. In this regard, in Case A, students used hypothesis as a synonym of prototype. Besides that, team members from Case B defined MVP as *"just the minimum set of features we should put in our platform"* instead of the correct concept of validation.

5.2 Survey results

From the 40 students surveyed, 37 answered (92,5% response rate). In the large majority (28 times), students focused on the *validation* of the idea, highlighting the **business learning** the course enabled. In their answers, students generally described how they would perform validation. The most common technique was *interview*, mentioned 11 times. In a student words: *"To interview the target audience and to understand what is its real needs and difficulties.".* Other tools mentioned were prototype (6), concierge (2) and talking to close people, like relatives (2). However, several students (6) did not detail how they would perform the validation they proposed. In summary, there is no learning cycle, in Lean Startup terms, in this process. Even though some learning happens during validation, this approach resembles a traditional software/business requirement process. These answers also illustrate **conceptual errors** students make. Figure 3 summarizes this idea: students understand validation as a step to get to the implementation, and not as a learning process.

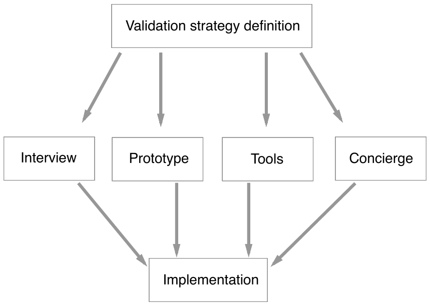

Figure 3: Students' perceptions on validation.

As another example of **conceptual errors**, five respondents criticized the idea without proposing to validate it. One student said that he would not participate in the project: *"I think the idea will not work, because doing an app to connect people that do not know how to use technology will not be effective to reach them, because they will never succeed to use this app without previous knowledge. The app will not reach its target audience. I suggest to pivot the idea.".* It is understandable to have personal opinions about an idea, but, the Lean Startup process makes it clear that one should look for facts, and not opinions. This outcome showed us how personal opinions and beliefs have a significant influence on startup projects.

Five times students focused on promote themselves to the fictional job and three of them would not even test the idea. For instance, *"I could contribute by developing the project, helping with brainstorm dynamics, prototypes ideas, etc. I could also help to put the platform website up, the meeting schedule service and an iOS app, organizing and managing the team if necessary.".*

Three students highlighted the importance of **listening to user feedback**. One student mentioned that *"it is highly important, as soon as we develop some features, to test with the audience in order to receive feedback so possible changes could be made".*

Marketing and distribution concerns were mentioned 3 times. For instance, one student said: *"Besides that, I would do ads in games that elderly people use to play [some mobile games] and advertise the platform there".*

Students mentioned concerns about **user experience** 6 times. One student described: *"To facilitate the app user flow and to avoid that the user loses herself, I would leave the minimum of options available in the screens. Besides that, I would put some gamification elements to make the user experience more pleasant".*

6 DISCUSSION

Our results suggest that a Lean Startup or Customer Development course indeed makes software engineering students aware of business related concerns of a software product like marketing. In our case studies, both groups tried to validate their business assumptions about users, although to different levels of success. Moreover, Case A also faced several issues about user interface and distribution. One reason that Team B have not have confronted them may be the shorter period spent in the project. Students' statements about learning also corroborate this observation.

Nevertheless, students face difficulties to avoid a straight plan focusing on technical challenges. They had already absorbed the plan-driven idea of "getting things done". From the first phase, two facts support this finding: from Case A, the observation from the first author as a mentor that the group tended to focus on developing the solution and, from Case B, the flow students drew. In the second phase, such an issue happens when students described technical plans on what to be done next ignoring a validation stage. Even when they suggested a validation stage, they do not think it in a cyclical manner: adding a validation as an initial stage to be performed before implementation.

In addition, students still struggle to understand concepts such as MVP and validation, generally using them as synonyms of simpler and already known concepts: MVPs as prototypes and validations as interviews. For example, students in Case B defined incorrectly an MVP as a product with a small set of features, ignoring the concept of business hypothesis, the practice core. In Case A, students wrongly used the term "hypothesis" to designate a prototype. For validation, Case B students and many survey respondents described the employment of interviews without further details.

Based on our findings, we suggest that software engineering students to be immersed in the concept of software business and customer relationships earlier in their formation process. More practical courses with real users or customers could be a solution. Such courses would be in line with the move from product to services seen in software products where "companies must transition from working with planned releases with detailed requirement specifications to continuously experimenting with customers" [5]. Since Case A's results were slightly better in than Case B (students performed a cyclical process focused on customer feedback), some indications could be gathered from Case A which suggest how to improve students' business capabilities. In this case, the mentor encouraged students to work in cycles and real users interacted with the platform. The problems they had with imprecise concepts and the lack of concern about the project's business sustainability should be the focus for improvement in the future courses.

6.1 Threats to validity

Runeson and Höst [31] describe a common scheme to assess threats to validity when reporting a case study composed by four aspects: construct validity, internal validity, external validity and reliability.

Construct validity reflects "to what extent the operational measures that are studied really represent what the researcher have in mind" [31]. The use of multiple information sources for both case studies reduces this issue as suggested by Yin [37].

Internal validity is related to causal relationships and represents the possibility of other factors not taken into account also explain a consequence. Although this study is exploratory, we used pieces of data to support each claim in data analysis and explored alternative explanations like students drivers. Triangulation of data from different sources within each case study also improved internal validity [22]. Another threat was that the first author was the first group mentor and it could have biased the data analysis. A detailed procedure and the results presentation to the other coauthors who were not involved in the cases mitigated this issue.

The use of three student groups in three different scenarios in two countries improved **external validity**.

The use of thematic analysis improved the **reliability** which goal is "to minimize errors and biases in the study", that is, if another researcher performs the same study in the future, she reaches the same results [37]. Besides that, the paper describes all steps performed in data collection and analysis.

7 CONCLUSIONS

This study focused on the challenges and benefits of startup methodologies courses for software engineering students. We performed a multiple-case study and a survey with students in three different contexts from two countries. Our results indicate that students understood the importance of business concepts and developed several soft skills, but still struggle to understand key aspects of the Lean Startup methodology, usually over-simplifying them, and experience a tendency to follow a straight path focused on technical success.

Based on the difficulties found, we proposed some insights that could be used to create or improve business and innovation related courses taught to computer science and similar areas. Lecturers should be concerned with important concepts, like MVP and validation, and perform a deeper inspection to check students' understanding of these concepts. They should also organize their practical courses to encourage students to follow a cyclical path avoiding the tendency to pursue a straight path.

Our results stimulate further studies on introducing business and innovation education into computer science and software engineering courses. For instance, it would be interesting to verify whether introducing these concepts in the beginning of the course, when students are less biased, could bring better learning results.

ACKNOWLEDGMENT

This work is partially funded by FAPERGS (17/2551-0001/205-4). The authors would like to thank the students who participated in this study.

REFERENCES

[1] D.F. Barbe. 2010. A Model of Cross Disciplinary Education, Technology Transfer and Teaching Non-Technical Skills for Engineers. In *Transforming Engineering Education: Creating Interdisciplinary Skills for Complex Global Environments*. IEEE Computer Society, Dublin, Ireland, 1–32.

[2] T. Batova, D. Clark, and D. Card. 2016. Challenges of lean customer discovery as invention. In *2016 IEEE International Professional Communication Conference (IPCC)*. 1–5.

[3] S. Blank. 2013. *The Four Steps to the Epiphany: Successful Strategies for Products That Win.* K&S Ranch, Incorporated.

[4] S. Blank and B. Dorf. 2012. *The Startup Owner's Manual: The Step-by-step Guide for Building a Great Company.* K&S Ranch, Incorporated.

[5] J. Bosch. 2016. Speed, Data, and Ecosystems: The Future of Software Engineering. *IEEE Software* 33, 1 (jan 2016), 82–88.

[6] J. Bosch, H.H. Olsson, J. Björk, and J. Ljungblad. 2013. The Early Stage Software Startup Development Model: A Framework for Operationalizing Lean Principles in Software Startups. *Lean Enterprise Software and Systems* (2013), 1–15.

[7] V. Braun and V. Clarke. 2006. Using thematic analysis in psychology. *Qualitative Research in Psychology* 3, 2 (jan 2006), 77–101.

[8] K. Buffardi, C. Robb, and D. Rahn. 2017. Tech startups: realistic software engineering projects with interdisciplinary collaboration. *Journal of Computing Sciences in Colleges* 32, 4 (2017), 93–98.

[9] R. Chanin, L. Pompermaier, K. Fraga, A. Sales, and R. Prikladnicki. 2017. Applying Customer Development for Software Requirements in a Startup Development Program. In *Proceedings - 2017 IEEE/ACM 1st International Workshop on Software Engineering for Startups, SoftStart 2017*. 2–5.

[10] R. Chanin, A. Sales, L. Pompermaier, and R. Prikladnicki. 2018. A Systematic Mapping Study On Software Startups Education. In *2018 Evaluation and Assessment in Software Engineering, EASE 2018*. 163–168.

[11] G. Coleman and R. O'Connor. 2008. An investigation into software development process formation in software startups. *Journal of Enterprise Information Management* 21, 6 (2008), 633–648.

[12] D. S. Cruzes and T. Dyba. 2011. Recommended Steps for Thematic Synthesis in Software Engineering. *2011 International Symposium on Empirical Software Engineering and Measurement* 7491 (2011), 275–284.

[13] E.F.Z. da Cruz and A. Alvaro. 2013. Introduction of entrepreneurship and innovation subjects in a computer science course in Brazil. In *2013 IEEE Frontiers in Education Conference (FIE)*. 1881–1887.

[14] M. Edwards, L. M. Sánchez-Ruiz, E. Tovar-Caro, and E. Ballester-Sarrias. 2009. Engineering students' perceptions of innovation and entrepreneurship competences. *Proceedings - Frontiers in Education Conference, FIE* (2009).

[15] B. Fitzgerald and K.J. Stol. 2017. Continuous software engineering: A roadmap and agenda. *Journal of Systems and Software* 123 (2017), 176–189.

[16] D.L. Frederiksen and A. Brem. 2017. How do entrepreneurs think they create value? A scientific reflection of Eric Ries' Lean Startup approach. *International Entrepreneurship and Management Journal* 13, 1 (2017), 169–189.

[17] A. Ghezzi, A. Cavallaro, A. Rangone, and R. Balocco. 2015. A Comparative Study on the Impact of Business Model Design & Lean Startup Approach versus Traditional Business Plan on Mobile Startups Performance. *Proceedings of the 17th International Conference on Enterprise Information Systems* (2015), 196–203.

[18] C. Giardino, X. Wang, and P. Abrahamsson. 2014. Why early-stage software startups fail: A behavioral framework. *Lecture Notes in Business Information Processing* 182 LNBIP (2014), 27–41.

[19] R. Harms. 2015. Self-regulated learning, team learning and project performance in entrepreneurship education: Learning in a lean startup environment. *Technological Forecasting and Social Change* 100 (2015), 21–28.

[20] J. Kontio, M. Ahokas, P. Poyry, J. Warsta, M.M. Makela, and P. Tyrvainen. 2006. Software Business Education for Software Engineers: Towards an Integrated Curriculum. *19th Conference on Software Engineering Education and Training Workshops (CSEETW'06)* (2006), 4–7.

[21] V. Lenarduzzi and D. Taibi. 2016. MVP Explained: A Systematic Mapping Study on the Definitions of Minimal Viable Product. In *2016 42th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*. 112–119.

[22] T.C. Lethbridge, S.E. Sim, and J. Singer. 2005. Studying software engineers: Data collection techniques for software field studies. *Empirical Software Engineering* 10, 3 (2005), 311–341.

[23] S.-C. Li. 2007. The Role of Value Proposition and Value Co-Production in New Internet Startups: How New Venture e-Businesses Achieve Competitive Advantage. In *Portland Int. Center for Management of Engineering and Technology (PICMET)*. 1126–1132.

[24] C. Nobel. 2011. Teaching a 'Lean Startup' Strategy. *Harvard Business School* (2011), 1–2.

[25] T. Nurkkala and S. Brandle. 2011. Software studio: teaching professional software engineering. In *Proceedings of the 42nd ACM Technical Symposium on Computer Science Education*. Dallas, TX, USA, 153–158.

[26] A. Osterwalder and Y. Pigneur. 2010. *Business model generation: a handbook for visionaries, game changers, and challengers.* John Wiley & Sons.

[27] J. Porter, J. Morgan, R. Lester, A. Steele, J. Vanegas, and R. Hill. 2015. A course in innovative product design: A collaboration between architecture, business, and engineering. In *Proceedings of 2015 IEEE Frontiers in Education Conference (FIE)*. IEEE Computer Society, El Paso, TX, USA, 1–5.

[28] E. Ries. 2011. *The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses.* Crown Business.

[29] C. Rioja Del Rio, A. Morgado-Estevez, and J.J. Dominguez-Jimenez. 2014. Entrepreneurship and lean manufacturing for software engineering. In *SEFI Annual Conference 2014*. Birmingham, UK.

[30] J. Rubin and D. Chisnell. 2008. *Handbook of usability testing: how to plan, design and conduct effective tests.* John Wiley & Sons.

[31] P. Runeson and M. Höst. 2009. Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering* 14, 2 (2009), 131–164.

[32] M. Shaw. 2000. Software engineering education. In *Proceedings of the conference on The future of Software engineering - ICSE '00*. ACM Press, New York, New York, USA, 371–380.

[33] A. Strauss, J.M. Corbin, and J. Corbin. 1998. *Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory.* SAGE Publications.

[34] S. M. Sutton. 2000. The role of process in software start-up. *IEEE Software* 17, 4 (Jul 2000), 33–39.

[35] X. Wang, D. Khanna, and P. Abrahamsson. 2016. Teaching Lean Startup at University: An Experience Report. In *International Workshop on Software Startups (IWSS) co-located with 22nd ICE/IEEE International Technology Management Conference.*

[36] C. Wohlin, P. Runeson, M. Höst, M.C. Ohlsson, B. Regnell, and A. Wesslén. 2012. *Experimentation in software engineering.* Springer Science & Business Media.

[37] R.K. Yin. 2003. *Case Study Research: Design and Methods.* SAGE Publications.

[38] L.A.M. Zaina and A. Alvaro. 2015. A Design Methodology for User-centered Innovation in the Software Development Area. *Journal of Systems and Software* 110, C (2015), 155–177.