On the Understanding of Experimentation Usage in Light of Lean Startup in Software Development Context

Bruna Prauchner Vargas
PUCRS – School of Technology
Porto Alegre
bruna.prauchner@acad.pucrs.br

Ingrid Signoretti
PUCRS – School of Technology
Porto Alegre
ingrid.manfrim@acad.pucrs.br

Maximilian Zorzetti
PUCRS – School of Technology
Porto Alegre
maximilian.zorzetti@acad.pucrs.br

Sabrina Marczak
PUCRS – School of Technology
Porto Alegre
sabrina.marczak@pucrs.br

Ricardo Bastos
PUCRS – School of Technology
Porto Alegre
bastos@pucrs.br

ABSTRACT

Experimentation can be used to validate software solutions, reducing waste on efforts and costs that would be spent had the solution not been validated from the beginning. Aware of experimentation benefits, a methodology named as Lean Startup have incorporated experiments and build-measure-learn loop as a manner to reduce waste from product development by focusing on efforts that will create value to customers, such as discovering the best solution. Aiming to characterize the use of experimentation in light of Lean Startup, we conducted a case study with two software development teams from a multinational information technology company. We characterize how experimentation works by teams perception, what are the benefits and challenges in the use of it. We identified in the study that experimentation was rooted in the workers' mindset and came up naturally should its need arise. Our findings indicate that the process of experimentation, albeit still a systematic effort, happens organically, and that it helps in development decisions and in reducing development efforts. Challenges are related to organizational culture and lack of belief in the approach by stakeholders.

CCS CONCEPTS

• **Software and its engineering → Agile development**.

KEYWORDS

Experimentation, Lean Startup, Agile Software Development, Case Study

ACM Reference Format:
Bruna Prauchner Vargas, Ingrid Signoretti, Maximilian Zorzetti, Sabrina Marczak, and Ricardo Bastos. 2020. On the Understanding of Experimentation Usage in Light of Lean Startup in Software Development Context. In *Evaluation and Assessment in Software Engineering (EASE 2020), April 15–17, 2020, Trondheim, Norway.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3383219.3383257

EASE 2020, April 15–17, 2020, Trondheim, Norway

ACM ISBN 978-1-4503-7731-7/20/04…$15.00
https://doi.org/10.1145/3383219.3383257

1 INTRODUCTION

The software development environment is highly dynamic and full of uncertainty in regards to market predictability, technical concerns, technological advancements, and a myriad of other aspects. Given the challenge of figuring what customers really need, the risk of developing a product that is not valuable to customers and accruing great costs is huge: asking customers what they want directly is a wasted effort as they usually can not predict what they want—there is a gap between what they do and what they say [8].

To create customer value and reduce risk, Lean Startup [9] is being used by many organizations around the world to change the way products are built and launched. Through the use of experimentation, Lean Startup helps in acquiring competitive advantages over other organizations, sometimes without even requiring an upfront development effort or technical infrastructure [8].

However, despite the interest in adopting Lean Startup principles in software development, there is little guidance available on how to introduce experimentation into an organization [13]. It gets even more difficult when considering a multinational context. Previous studies show that key challenges for running experiments at such company setting include organizational culture, technical obstacles, complex stakeholder structures, difficulties in defining success criteria, and developing experimentation skills [13].

Motivated by these issues, the goal of this study is to contribute to the understanding of experimentation in light of Lean Startup in a software development context through a case study which aims to characterize how experiments work, who gets involved in it, and what are the benefits and challenges of this approach. We studies two teams of a multinational IT company for 6 months, conducting semi-structure interviews, observations, a questionnaire, and a focus group session. Results revealed that experimentation happens organically and has several benefits, with most of its challenges rooted in organizational concerns.

The remainder of this paper is organized as follows: Section 2 describes Lean Startup and experimentation in software development. Section 3 details the setting and the data collection and analysis procedures of the case study. Section 4 reports on its results and Section 5 contrasts our findings with previous studies from literature. Section 6 concludes the paper with final remarks and thoughts on potential future work.

2 LEAN STARTUP AND EXPERIMENTATION

Lean Startup is an entrepreneurship method inspired by lean manufacturing principles that focuses on iteratively developing a viable business plan for startup companies—"human institution[s] designed to create a new product or service under conditions of extreme uncertainty" [9]. The method consists in acquiring customer feedback and using it to strategize the startup's next moves. It achieves this through its primary activity, the Build-Measure-Learn (BML) loop, in which experiments are built to measure how customers respond to an idea (e.g., a product), enabling the startup to confidently persevere on the idea or pivot to another one entirely.

The method has been reported as a way for existing software companies to innovate [2] and as a driving force to software development [3, 4, 12]. Continuous experimentation lies at the core of Lean Startup, and embracing it to develop software demands technological capabilities (e.g., continuous deployment) and organizational support (e.g., culture) [8]. For the purposes of this paper, we consider experimentation as controlled experiments that aim to test hypotheses that range from technical (e.g., optimization) to product (e.g., prospective features) to business-level concerns.

3 RESEARCH METHOD

The goal of this study is to characterize the usage of experimentation in light of Lean Startup in a software development context. We sought to identify how experimentation takes place and what are its benefits and challenges in such context. To do so, we posed the following research questions to guide our study:

RQ1 How does experimentation usage take place in light of Lean Startup in a software development context?

RQ2 What are the benefits of using experimentation in light of Lean Startup in a software development context?

RQ3 What are the challenges faced when using experimentation in light of Lean Startup in a software development context?

To answer our research questions, we conducted a case study [11] with two teams of a multinational IT company. We introduce the case setting and data collection and analysis procedures next.

3.1 Case Setting

We conducted a case study in a multinational information technology company called ORG (name omitted for confidentiality reasons) that has software product development sites in the USA (headquarters), India, and Brazil. With over 7,000 employees and responsible for about 1,200 software products, ORG's IT department started its agile transformation in 2015 and moved to the combined use of Agile, User-Centered Design, and Lean Startup principles in late 2017. The adopted approach was inspired in the Pivotal Labs[1] methodology, which proposes a "team rhythm" composed of principles and ceremonies based on the three aforementioned approaches. It also suggests the adoption of a cross-functional team composed of three main roles: Product Designer, Product Manager, and Software Engineer. Pivotal Labs' main goal is to help teams to build software products that deliver meaningful value for users and their business. It offers a framework and a starting point for any team to discuss its needs and define its own way towards software development, including roles, practices, work products, etc.

[1] http://pivotal.io/labs

We observed *in loco* two software development teams from ORG's financial department located in Brazil. Both teams were built as a sort of catalyst to prove the worth and spread the use of Pivotal Labs throughout the company and have been rated as high-performance and proficient in the use of the methodology. To achieve this, some members (enablers) underwent an immersive Pivotal Labs hands-on training at the company headquarters over the supervision of Pivotal Software Inc consulting personnel before coming back to Brazil to teach the others (learners; see more in our previous paper [12]). We describe each team's composition and product characteristics next.

- *Team A* is responsible for a software product that manages, calculates, and generates data about company projects related to equipment (e.g., peripherals and computers for personal or server use) and service delivery (e.g., machine installation, support, and replacement). The product manages general project information, such as personnel assignment and time spent on tasks. It also calculates the associated costs of services offered by the products sold by ORG and displays this information to internal ORG consumers. The application generates profit data for each project which is consumed (along with the rest of the data) by the accounting department. Team A is composed of: one Product Designer (enabler), two Product Managers (enabler and learner), and four Software Engineers (two enablers and two learners).
- *Team B* is responsible for a software product that consumes data from multiple ORG applications (including Team A's) to calculate the average cost of equipment developed in Brazil. The application generates reports for internal accounting, such as inventory reports for tax purposes. The team is also working on automating the validation process for the data coming from each source. Team B is composed of: one Product Designer (enabler), two Product Managers (enabler and learner), and four Software Engineers (two enablers and two learners).

These teams worked for 6 months in a dedicated lab that follows Pivotal Labs' collaborative work environment recommendations (e.g., single large table for pair-wise work, large screen TV for reports and news, large whiteboards for idea development and information sharing, and a meeting room that turns into an entertainment space for leisure time). The lab is located on PUCRS's campus grounds and was specifically built for ORG teams as a learning environment. The research team received a room in this lab to closely investigate the teams as part of a larger research project.

3.2 Data Collection and Analysis

We used multiple data sources to conduct the study:

- A *questionnaire* was used to collect the participants' profiles (name, role, main responsibilities, time in years working in information technology and at ORG, and whether the person participated in the Pivotal Labs immersion training—an enabler—or is being trained by those who did—a learner). Table 1 presents a summary of the participants' profiles.
- *Observation sessions* in which we both shadowed a few teams members and attended team meetings or discussions to broadly learn their approach to experimentation.

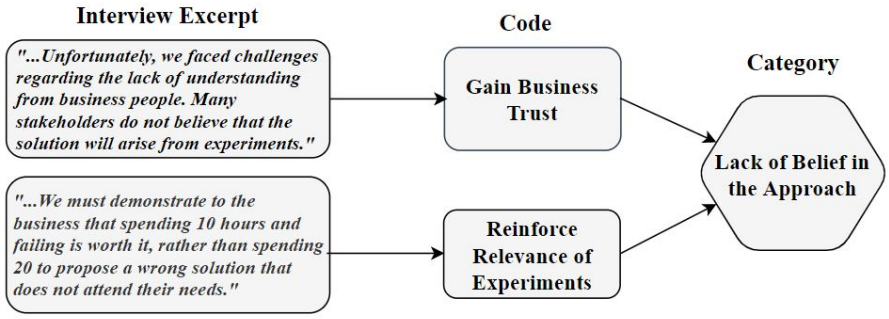

Figure 1: Code analysis example

Table 1: Participants' Profiles

ID	Role	Training	IT Exp.	ORG Exp.
P1	Software Engineer	Enabler	10	4
P2	Product Manager	Enabler	19	0.5
P3	Product Designer	Enabler	27	10
P4	Software Engineer	Enabler	21	8
P5	Software Engineer	Learner	7	7
P6	Product Manager	Enabler	21	6
P7	Product Designer	Enabler	5	4
P8	Product Manager	Learner	23	10.5
P9	Software Engineer	Enabler	20	11
P10	Software Engineer	Enabler	5	5
P11	Software Engineer	Learner	6	1
P12	Software Engineer	Learner	15	11
P13	Product Manager	Learner	16	7.5
P14	Software Engineer	Learner	5.5	4

- One *focus group* session was conducted to have the participants illustrate how experimentation takes place in their team's context. The participants also pointed out benefits and challenges of experimentation. The session was attended by eight enablers: two Product Designers, two Product Managers, and four Software Engineers. It lasted approximately 1.5 hours and was recorded and transcribed for analysis.
- A *semi-structured interview* was used to confirm the data collected from the focus group session and to discover more benefits and challenges of experimentation. The interview was conducted with three participants: a Software Engineer (P10) from Team A, a Product Designer (P7) from Team B, and a Product Manager (P6) from Team B. We looked to contrast their perception of experimentation across the different roles. The interview lasted 30 minutes and was recorded and transcribed for analysis.

Regarding data analysis, we conducted the content analysis procedure by Krippendorff [7] using a qualitative approach to ethnographic content analysis, focusing on the narrative description of the situations, settings, and the perspective of the actors involved in the phenomena. As we used recording/coding units, we organized the analysis into the following steps: organization and pre-analysis, reading and categorization, and recording the results. Using the Atlas.TI[2] tool, we first read the dataset, extracted text excerpts, and marked them as codes (Figure 1). These codes were revisited and grouped into larger codes, eventually forming categories.

In regards to empirical validity, we conducted a series of reviews with senior researchers in order to mitigate validity threats. Two senior researchers reviewed the questionnaire and interview scripts and assisted in defining question content, order, and terminology. As for the iterative analysis process, it was conducted by two researchers and continuously reviewed by two senior researchers.

4 CASE STUDY RESULTS

In this section we answer our research questions based on the findings of our case study.

4.1 RQ1: How does experimentation usage take place in light of Lean Startup in a software development context?

ORG is undergoing a transformation process to a combined approach of Agile, User-Centered Design, and Lean Startup methodologies. The adoption of this approach included a set of new activities and techniques derived from the aforementioned methodologies. The most significant change was the inclusion of the Build-Measure-Learn cycle, and the usage of experimentation as a consequence. Our study arises from the use of these Lean Startup concepts, specifically experimentation, in the context of the adoption of the combined approach as perceived by Team A and B.

The introduction of **experimentation is highly associated with the Build-Measure-Learn cycle** as a software engineer highlighted — *"The experiment is built-in on Build-Measure-Learn."* (P4). He also adds about the outcomes that the experiment can generate — *"When you enter the Build-Measure-Learn cycle, the moment you learn from the experiment, this knowledge can reorder your problem priority or your solution list. You can also discard a problem because you can't solve it. These are all possible outcomes."*

[2] atlasti.com

The participants mentioned that the **Build-Measure-Learn cycle with experimentation at the core is not just for software**. In the following example, the participants illustrate how the experimentation approach encourages solutions that might not even be software-related — *"First, we analyze the problem. For example, our users were facing problems when using spreadsheets with a substantial amount of data. They were spending a lot of time in this one application, usually about three days between calculating what they needed and waiting for the results. Based on that, our assumption was that maybe the problem was with the host machine's performance and not with the application that generates the spreadsheet. So we defined a hypothesis that a computer with better specifications would be able to handle the workload, and got to confirm it. After that, we suggested that the host machine needed some new parts and we managed to solve the problem without having to write a single line of code."* (P9)

As mentioned before, the application of the BML cycle as a development approach was a huge change to their software development process. The participants mentioned that **the cycle is ubiquitous** — *"We argue that the cycle is in everything."* (P6, P7, P10) and Team A's Product Manager supplements — *"We use BML all the time in any part of our process. For example, a user interview. If we are defining the interview script, we are building something. We measure the script's value by observing the interview's findings, trying to understand if we collected the right data or not. Maybe the stakeholders answered the question correctly, but the question was flawed from the beginning. This process allows us to learn from our failures and to create better, more accurate scripts afterwards. BML is applicable to any product development activity."* (P2)

The participants also reveal **the systematic way in which experimentation is executed** by them — *"At the beginning of it all, we define an assumption and derive a hypothesis from that assumption. For example, our hypothesis for that problem was: do the calculations take a long time because of the current computer specifications? From the hypothesis, we can define our next steps to conduct the experimentation."* (P6)

Another example mentioned by the team — *"Our users asked to not send a wrong token, otherwise the application will fail. A token is a numeric code and our application supports a list of tokens. Our end sends another list of tokens and the users have to do the matching themselves. Using the Build-Measure-Learn cycle, we followed the systematic steps for experimentation and considered as a hypothesis the question: can projects fail because of a token? And this lead us to explore the whole token cycle during the experiment."* (P2)

The participants also said how they define the assumptions using different techniques — *"We use a technique called 'How might we do this', and then we use another called 'We know we are right if we do this' to create an assumption of how we would know if we were right, to know if we were going to get things right. Only from that, one can define the metrics."* (P2) A Product Manager mentioned another technique — *"You continuously check if you are consistent and sound... If the strategy you thought of in the short, medium, or long term is still valid. We do a lot of this in a technique we call 'Now, near, next'."* (P6)

The participants also mentioned the need to measure and validate hypotheses — *"We define metrics to validate our hypothesis. We also use some specific indicators as objectives and key results or key performance indicators to measure the success of our experiments."* (P8) Besides the quantitative indicators of product success, the participants declared that qualitative feedback is also useful — *"It is not possible to gather quantitative metrics most of the time. So stakeholder feedback is a valuable indicator that the product is aligned with their needs."* (P6)

Although the experiment conduction is systematic, following sequential steps (e.g., assumption, hypothesis, experiment), the act of using or applying the experiments is organic in the development context. It is not an event or an activity defined for an specific step. The participants mentioned that conduct experiments is almost natural during the product development.

4.2 RQ2: What are the benefits of using experimentation in Lean Startup in a Large-Scale Software Development Context?

The benefits mentioned by the participants were organized into categories as they emerged during the coding process.

One of the benefits reported by the participants is that now they have **focus on the problem understanding** rather than only refining software requirements — *"By discussing the problem we can consider different solutions. They almost come up naturally."* (P2) Another participant reinforces this claim — *"When you say that you are going to deliver a requirement, it is what has been set and that is it. When you change the nomenclature to experiment terms, you end up not committing to that."* (P9) As a consequence, the participants mentioned that using experimentation increases **perceived added value** in the product — *"We are actually aggregating value to our products. We explore the problem that business brings to us, and by the end of it, we address their needs in the product."* (P7)

The participants mentioned that they call the development of features experiments. In their opinion, this takes the burden off of having to deliver a closed solution, enabling them to conduct experiments and search for a better solution — *"Within the company you have a requirement and a feature perspective. The business area asks for a feature or shows up with the requirements ready. For example, 'I want a button on the side of a screen and I want to sign the client up'."* (P9) — *"When you change this nomenclature to experiment, for example, 'I want to perform an experiment to sign the customers up through another platform', it allows you to not have 100% assertiveness that it will work."* (P9)

Yet another benefit of using experimentation is **having room to fail** up front during development — *"We used to work based on sprints and release plans: there was no room whatsoever for experiments and failure. With our new continuous development and release approach, we can explore, test, and pivot candidate solutions. Our time slots give room for value-driven development."* (P1) A Product Manager adds — *"Product development is uncertain and very susceptible to failure. The experimentation as a core of BML gives us room to fail but also allows us to fail and fix quickly. We do not need to wait until the end of the iteration to discover that we do not understand the stakeholder needs. The conduction of experiments gives this information as early as possible."* (P2)

The aspect of **reducing development effort** was an easily perceived benefit — *"Using this example of registering a customer, suppose the experiment has been validated and customers want to sign up.*

Based on that, do I need the whole registration module implemented? Or do I just need a part of it?" (P9) Team B's Product Designer illustrates this in practice through an example — *"We received a bunch of change requests from the business to be implemented. We received predefined solutions. So, we decided to investigate first and conduct an experiment, mocking up data and creating prototypes. We discovered that they have one minor problem that was fixed just by including a new field in the application. We are sure that we avoided wasting significant effort into developing the predefined solutions by executing an experiment beforehand."* (P7) In addition to reducing development effort, this notion of not having to necessarily implement everything that is requested can leave room for the business to generate new insights.

We also identified **rapid feedback and validation** as another benefit provided by the use of experimentation — *"We can easily and in less time conclude whether it is worth to follow that path or if we have to pivot the solution."* (P9)

4.3 RQ3: What are the challenges faced when using experimentation in Lean Startup in a Large-Scale Software Development Context?

The **lack of belief in the approach** was cited as a challenge by the Product Designer from Team B, which calls attention to the difficulty of making people understand — *"It is hard to sell this idea, to make people buy it. The idea that doing the experiments we can get the solution they want. Not everyone at the company believes this, that it is possible to come up with the ideal solution for the business, because they think they already have the ideal solution."* (P7) and a Software Engineer adds — *"It's also a challenge to be able to communicate this to the business [department]. Why are we running an experiment instead of doing what they asked?"* (P9)

Another point of concern for the participants is the fact that the current **organizational culture does not welcome failure**. A Product Designer considers that failure is not well seen — *"It's something that we haven't talked about, the issue of failures, how we work... You break problems, analyze, do the experiments... we treat failure as a learning process. Today, I see in the company that sometimes failure is an awful thing. So, we still have to work on the mindset and account that somehow from the project management perspective."* (P3) and a Software Engineer adds — *"Failure is not being understood as a part of the process of learning. This is one of the most challenging points: interpreting a failure as worthwhile."* (P9)

Another challenges is related to **top-down decision about the problems priority**. Currently, these decisions comes from the three major company areas — *"It comes from business, from what we think makes sense, and from the product development area. We understand that there are some priority issues that can not be avoided. Some priorities are imposed on us and we have difficulty in understanding their value. However, even top-down prioritization, we can re-prioritize easily and fit in a good way in the backlog."* (P7) The teams reported that this happens because of team maturity, since the two teams are made up of seniors.

Although this lack of decision-making in the problem prioritization, **puts in risk the teams' autonomy**. The participants mentioned this as a challenge because it implies the factor of having the autonomy to made teams decisions about their scope — *"We must have this free pass to make our own decisions. Decide the solution, what is the best for the user - of course, with the agreement. The point is that the team is the owner of the product, and this decision is ours. We need to make others [business, customers, etc] understand that we are now co-owners of a software product."* (P5)

5 DISCUSSION

In this section we discuss our findings in light of existent studies from literature per research question.

RQ1. How does experimentation usage take place in light of Lean Startup in a software development context?

We identified that experimentation is highly associated to the Build-Measure-Learn cycle, as teams embrace the experimentation that is present at its core. Models proposed for continuous experimentation also follow the BML cycle [1, 3, 10]. The difference between our findings and the literature is that the latter's models define stages, phases, or levels, while our case reports that experimentation is done organically by the teams: it is present in their way of thinking. When there is a problem to solve, the team considers it essential to brainstorm solutions and think about what the root cause of the problem is, requiring patience, time, and availability (precious resources that not every company can afford) to analyze what needs to be solved rather than simply developing software without caring if the user's problem will actually be solved.

Another interesting finding is that experimentation and BML are not restricted to software solutions. The teams reported that, as they are working in a problem-oriented mindset, their focus is on solving stakeholder problems, be it through software or non-software solutions.

Both teams were following a consistent strategy in applying experimentation in a systematic manner by: defining assumptions, transforming them into hypothesis, defining metrics, executing the experiment, collecting metrics, and validating said assumptions. Fagerholm et al. [3] and Lindgren and Münch [8] reinforce that experimentation must be systematic and continuous.

Regarding problem exploration, Björk, Ljungblad, and Bosch [1] argue on the need to think and understand the problem first by using experimentation. This is directly aligned with the problem-oriented approach that allows the teams to explore the problem and devise different ideas for solutions, which can in turn be used as hypothesis for conducting experiments, allowing the teams to pivot to and persevere on the best metrically-validated solutions. Furthermore, in moderation, hypothesis should be appreciated and encouraged in an attempt to dismiss the belief that mistakes are harmful and instead acknowledge them as a part of the expected learning process.

Björk, Ljungblad, and Bosch [1] mention the importance of having two types of metrics—quantitative and qualitative. The teams confirmed the use of both metric types, although we identified a preference for quantitative metrics in our interactions with them.

RQ2. What are the benefits of using experimentation in light of Lean Startup in a software development context?

Concerning benefits, the teams pointed out that the focus on the problem understanding was one of the most impactful changes.

They claim that by first discussing and thinking about the problem helps in recognizing how the experiments need to be conducted. As mentioned before, Björk, Ljungblad, and Bosch [1] already address problem understanding as the first stage of their model, emphasizing the relevance of thinking first and executing later.

Gutbrod, Münch, and Tichy [5] report that experimentation enables a better understanding of customers' needs, priorities, behaviors, and continuously better prioritization of development activities. The teams corroborate on this by reporting that using experimentation in the software development process results in products that are more aligned with stakeholders needs, increasing their perceived added value. This due to experimentation allowing for more room for failure, enabling the search for a better solution.

Reduced development effort was also mentioned by the teams as a benefit of using experimentation, in accordance with Yaman et al. [13] and the core lean manufacturing principle of reducing waste.

RQ3. What are the challenges faced when using experimentation in light of Lean Startup in a software development context?

The challenges perceived by the teams are associated with organizational issues, such as lack of belief in the development approach and resistance in accepting failures as an important part of the learning process. Lindgren and Münch [8] faced a similar challenge with the organizational culture of their case setting. Yaman et al. [13] claimed that proposing this type of change in organizations is sometimes difficult due to complex organizational structure. Cultural changes, such as working habits [6], are part of the concerns but they are perceived as manageable at the development team level. Notably, we found that the ORG teams perceive the change of existing nomenclature to experiment-related terms as a beneficial factor in improving organization support.

Additionally, both teams reported difficulties with top-down stakeholder directives that impact the priorities of their work backlog. This has significant negative repercussions on the teams' autonomy and hinders their ability to organically investigate problems.

6 CONCLUSION, LIMITATIONS, AND FUTURE WORK

We reported through a case study how experimentation in light of Lean Startup takes place in a multinational information technology company through the lenses of two software development teams from the company's financial department. To accomplish that, we used a questionnaire to map the participants' profiles, observation sessions to broadly learn the teams' approach to experimentation, and a focus group session along with a semi-structured interview to characterize the usage of experimentation in software development and explore its benefits and challenges.

The case study results revealed that experimentation is conducted in an organic manner—the experiments themselves are still systematic, but are not predetermined in a process. Regarding benefits, the teams report that experimentation helps them to focus on the problem understanding, as well as having room for failure, reductions in development effort, and rapid feedback and validation. As for challenges, lack of belief in the experimentation approach from business people, failure-averse culture, top-down directives on problem priority, and reduced team autonomy were the main adversities identified.

We are aware that our research may have some limitations. Inherent to any empirical study is concerns with construct validity, as to whether the scenario of the study is representative of the real world, and external validity, as to whether our findings are generally applicable. Although we cannot claim that our results are applicable to other distinct scenarios, as we observed two teams with team members performing distinct roles and coming from different backgrounds in a real setting that promotes collaboration, we argue that the nature of our case study helps to ease this limitation.

Seeking to deepen the understanding of this phenomenon and expand to other scenarios, our next step aims to replicate this study with other teams with ORG and perhaps other organizations in order to contrast the findings and look for additional and rich examples to further our understanding on the matter.

ACKNOWLEDGMENTS

We thank the study participants and acknowledge that this research is sponsored by Dell Brazil using incentives of the Brazilian Informatics Law (Law no 8.2.48, year 1991).

REFERENCES

[1] Jens Björk, Jens Ljungblad, and Jan Bosch. 2013. Lean Product Development in Early Stage Startups.. In *Proceedings of the Workshop on Second-Order Quantifier Elimination and Related Topics (CEUR Workshop Proceedings)*. CEUR-WS, 13. http://ceur-ws.org/Vol-1095/paper_02.pdf

[2] Henry Edison, Xiaofeng Wang, and Pekka Abrahamsson. 2015. Lean Startup: Why Large Software Companies Should Care. In *Scientific Workshop Proceedings of the XP2015 (XP '15 workshops)*. Association for Computing Machinery, New York, NY, USA, Article Article 2, 7 pages. https://doi.org/10.1145/2764979.2764981

[3] Fabian Fagerholm, Alejandro Guinea Sanchez, Hanna Mäenpää, and Jürgen Münch. 2017. The RIGHT Model for Continuous Experimentation. *Journal of Systems and Software* 123 (January 2017), 292 – 305. https://doi.org/10.1016/j.jss.2016.03.034

[4] Benjamin Grossman-Kahn and Ryan Rosensweig. 2012. Skip the silver bullet: driving innovation through small bets and diverse practices. *Leading Through Design* (2012), 815.

[5] Matthias Gutbrod, Jürgen Münch, and Matthias Tichy. 2017. How Do Software Startups Approach Experimentation? Empirical Results from a Qualitative Interview Study. In *Product-Focused Software Process Improvement (PROFES 2017)*, Michael Felderer, Daniel Méndez Fernández, Burak Turhan, Marcos Kalinowski, Federica Sarro, and Dietmar Winkler (Eds.). Springer International Publishing, Cham, 8. https://doi.org/10.1007/978-3-319-69926-4_21

[6] T. Karvonen, H. Sharp, and L. Barroca. 2018. Enterprise Agility: Why is Transformation So Hard?. In *International Conference on Agile Software Development*. Springer International Publishing, Porto, Portugal, 131–145.

[7] Klaus Krippendorff. 2013. *Content Analysis - 3rd Edition: an Introduction to Its Methodology*. SAGE Publications, Inc.

[8] Eveliina Lindgren and Jürgen Münch. 2016. Raising the Odds of Success: The Current State of Experimentation in Product Development. *Information and Software Technology* 77 (04 2016), 80–91. https://doi.org/10.1016/j.infsof.2016.04.008

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

[10] Olli Rissanen and Jürgen Münch. 2015. Continuous Experimentation in the B2B Domain: A Case Study. In *2015 IEEE/ACM 2nd International Workshop on Rapid Continuous Software Engineering (RCoSE 2015)*. IEEE, Florence, Italy, 12–18. https://doi.org/10.1109/RCoSE.2015.10

[11] Per Runeson and Martin Höst. 2008. Guidelines for Conducting and Reporting Case Study Research in Software Engineering. *Empirical Software Engineering* 14 (2008), 131. https://doi.org/10.1007/s10664-008-9102-8

[12] I. Signoretti, S. Marczak, L. Salerno, A. d. Lara, and R. Bastos. 2019. Boosting Agile by Using User-Centered Design and Lean Startup: A Case Study of the Adoption of the Combined Approach in Software Development. In *International Symposium on Empirical Software Engineering and Measurement (ESEM), Porto de Galinhas, Brazil*. IEEE, 1–6. https://doi.org/10.1109/ESEM.2019.8870154

[13] Sezin Gizem Yaman, Myriam Munezero, Jürgen Münch, Fabian Fagerholm, Ossi Syd, Mika Aaltola, Christina Palmu, and Tomi Männistö. 2017. Introducing Continuous Experimentation in Large Software-Intensive Product and Service Organisations. *Journal of Systems and Software* 133 (Nov 2017), 195 – 211. https://doi.org/10.1016/j.jss.2017.07.009