Continuous Experimentation in Finnish startups – A descriptive case study of A Grid and Maria 01 communities

Vihtori Mäntylä

School of Science

Thesis submitted for examination for the degree of Master of Science in Technology.
Espoo 30.12.2021

Supervisor

Dr Fabian Fagerholm

Advisor

MSc Bettina Lehtelä

Aalto University
School of Science

Aalto University
School of Science

Aalto University, P.O. BOX 11000, 00076 AALTO
www.aalto.fi
Abstract of the master's thesis

Author Vihtori Mäntylä

Title Continuous Experimentation in Finnish startups – A descriptive case study of A Grid and Maria 01 communities

Degree programme Information Networks

Major Information Networks **Code of major** SCI3047

Supervisor Dr Fabian Fagerholm

Advisor MSc Bettina Lehtelä

Date 30.12.2021 **Number of pages** 66+6 **Language** English

Abstract

Startups are more likely to fail than succeed. Continuous experimentation has been proposed as a practice that can help startups succeed where others have failed.

This study evaluates the product development practices used by five Finnish startups and describes the factors that could affect the adoption of continuous experimentation.

The study uses a descriptive multiple case study approach and the primary data collection method is semi-structured interviews. The study uses qualitative data analysis to answer the three research questions (RQs). RQ1: What product development practices are the startup companies in the A Grid and Maria 01 communities using? RQ2: How do startup companies in the A Grid and Maria 01 communities use data in product development? RQ3: What factors affect the adoption of continuous experimentation in startup companies in the A Grid and Maria 01 communities?

The case study data shows that all companies are using lean or agile software development practices. All companies had set up continuous integration and continuous deployment systems and were able to produce new releases with little manual effort. The product management practices used by the companies were found to be fairly unstructured and the most common model for planning the product development work appeared to be weekly or biweekly meetings. Experimentation was found to be common among the case companies. However, only one company was doing experimentation systematically and using the experimental data in a systematic way. Continuous experimentation was not used by any of the case companies and only one interviewee recognized the practice when asked.

The case study indicates that the case companies are unlikely to adopt continuous experimentation, or any other practice, unless three conditions are met. First, a company employee must have prior experience of the practice. Second, the adoption of the practices must solve a real problem experienced by the company. Third, adopting the practice must provide perceived value to the company almost immediately. The thesis proposes teaching continuous experimentation in university or as a part of entrepreneurship training as a possible way for making the practice more widely adopted.

Keywords Continuous experimentation, software development, startup, product development, software engineering

Aalto University
School of Science

Aalto-yliopisto, PL 11000, 00076 AALTO
www.aalto.fi
Diplomityön tiivistelmä

Tekijä Vihtori Mäntylä

Työn nimi Jatkuva kokeilu suomalaisissa startup-yrityksissä – Kuvaileva tapaustutkimus A Grid ja Maria 01 yhteisöistä

Koulutusohjelma Information Networks

Pääaine Information Networks **Pääaineen koodi** SCI3047

Työn valvoja TkT Fabian Fagerholm

Työn ohjaaja MSc Bettina Lehtelä

Päivämäärä 30.12.2021 **Sivumäärä** 66+6 **Kieli** Englanti

Tiivistelmä

Startupit epäonnistuvat todennäköisemmin kuin onnistuvat. Jatkuvaa kokeilua on ehdotettu menetelmäksi, joka voi auttaa startuppeja onnistumaan siinä missä muut ovat epäonnistuneet.

Tämä diplomityö tutkii viiden suomalaisen startupin käyttämiä tuotekehitysmenetelmiä ja kuvaa tekiköitä, jotka voivat vaikuttaa jatkuvan kokeilun käyttöönottoon.

Tämä työ on kuvaileva monen tapauksen tapaustutkimus, jossa ensisijaisena tiedonkeruumenetelmänä käytetään teemahaastatteluita. Tutkimuksessa käytetään kvalitatiivista data-analyysiä selvittämään vastaukset kolmeen tutkimuskysymykseen (TK). TK1: Mitä tuotekehitysmenetelmiä startup-yritykset A Grid ja Maria 01 yhteisöissä käyttävät? TK2: Miten startup-yritykset A Grid ja Maria 01 yhteisöissä käyttävät dataa osana tuotekehitystä? TK3: Mitkä tekijät vaikuttavat jatkuvan kokeilun käyttöönottoon A Grid ja Maria 01 yhteisöihin kuuluvissa startup-yrityksissä?

Tapaustutkimuksen aineisto osoittaa kaikkien tutkimusessa tutkittujen yritysten käyttävän ketteriä ohjelmistokehitysmenetelmiä. Kaikki yritykset olivat myös ottaneet käyttöön jatkuvan integraation ja jatkuvan toimituksen mahdollistavat järjestelmät. Yritykset pystyivät tuottamaan uusia tuotantojulkaisuja ilman suurta käsin tehtävää työmäärää. Tutkimukseen osallistuneiden yritysten käyttämät tuotehallinnan menetelmät eivät vaikuttaneet hyvin jäsennellyiltä, ja yritykset käyttivät lähinnä viikottaisia tai kahden viikon välein toistuvia palavereja tuotekehitystyön suunnitteluun. Erilaiset kokeilut olivat yleisiä yritysten keskuudessa, mutta vain yhdellä yrityksellä kokeilu ja datan käyttö oli järjestelmällistä. Jatkuvaa kokeilua ei käyttänyt yksikään tutkimukseen osallistunut yritys. Ainoastaan yksi haastateltava tunnisti menetelmän kysyttäessä.

Tapaustutkimus indikoi, että yritykset eivät todennäköisesti ottaisi jatkuvaa kokeilua tai mitään muutakaan menetelmää käyttöön, ellei seuraavat kolme ehtoa täyty. Ensimmäiseksi, yrityksen työntekijällä tulee olla aikaisempaa kokemusta menetelmän käytöstä. Toiseksi, menetelmän käyttöönoton tulee ratkaista jokin oikea yrityksen kokema ongelma. Kolmanneksi, menetelmän käyttöönoton tulee tuottaa yritykselle havaittavaa hyötyä miltein välittömästi käyttöönosta lähtien. Ehdotamme jatkuvan kokeilun opettamista yliopistossa tai yrittäjäkoulutuksen osana jatkuvan kokeilun laajemman käyttöönoton saavuttamiseksi.

Avainsanat Jatkuva kokeilu, ohjelmistokehitys, startup, tuotekehitys, ohjelmistotuotanto

Preface

Thank you Dr Fabian Fagerholm and MSc Bettina Lehtelä for the numerous Friday morning discussions that kept me going through the thesis process.

Thank you Pia for making me do this.

Kulosaari, 20.12.2021
Vihtori V. O. Mäntylä

Contents

1 Introduction

Building software based products is easier than ever before. Novel tools and frameworks developed, used, and shared by the most successful tech companies of modern era are available for anyone to study and use. Combined with vast amount of free training material available online allows anyone with an internet connection to learn how to build their own product ideas into reality. Even though writing software has become more accessible than ever before, companies are still struggling to build products that produce actual customer value. Software product companies may be able to implement their vision only to find out that the initially identified customers do not want to use the product.

This mismatch of product features and customer needs is one of the most common reasons of software project failures (Klotins, Unterkalmsteiner, and Gorschek, 2019). While waterfall model of software development has been widely replaced by iterative agile methods, building the right product appears to be a feat that a majority of startups fail to achieve. Already a decade a go, Eric Ries in his book *The Lean startup* (Ries, 2011) attempted to provide a recipe for building product that better match the market demand. He proposes that companies should release early and often in order to collect as much data about the market as possible. This data is then used to validate businesses' underlying assumptions and guide further development of the product. Even though Ries provides multiple examples of successful use of the method, the book falls short in explaining how to identify, design, and run the experiments required by his method.

Continuous experimentation (CE) is a new approach for software product development that builds on Ries' ideas and proposes a more scientific approach for running experiments in cycles (Fitzgerald and Stol, 2017). Scientific approach for experiments mandates that experiments should be used to validate a hypothesis and the validation should be done based on predefined metrics. Furthermore, CE takes a holistic approach to software development by considering the roles and activities of the organisation as a whole instead of treating the software development function as an isolated entity. Companies practicing CE have reported improved product quality and business performance as benefits of the new approach (Fabijan, Dmitriev, Olsson, et al., 2017). In addition to individual case studies, similar benefits are reported from a randomized control trial of Italian startups (Camuffo et al., 2020).

1.1 Problem statement

This thesis studies the use of experimentation in Finnish startup companies located in two communities: A Grid and Maria 01. The A Grid community is located at the Aalto University campus in Espoo. The Maria 01 is a community-driven initiative co-owned by the City of Helsinki, located close to central Helsinki. Previous studies (Lindgren and Münch, 2015; Lindgren and Münch, 2016) have found that startups are likely to run experiments as a part of their product development process. However, the experiments are rarely planned nor run in an organized and systematic manner characteristic to CE (Lindgren and Münch, 2015). Furthermore, analysis of 88

case studies by Klotins (Klotins, Unterkalmsteiner, and Gorschek, 2019) found that startup companies are often reluctant to incorporate industry best practices, methods, and frameworks proposed by researchers into their business and product development processes. The goal of this thesis is to describe the state of experimentation in target companies as well as understand the evolution of the experimentation process.

1.2 Structure of the thesis

This thesis is structured as follows. In Section 2, we describe the previous work related to the Lean Startup model (Ries, 2011), continuous software engineering (Fitzgerald and Stol, 2017), and continuous experimentation (Fagerholm et al., 2017). In Section 3, we describe the case study approach and qualitative data analysis practices used in the study. In Section 4, we describe the results of our case study based on the case company interviews. In Section 5, we discuss the relevance of our findings and compare of our results with the previous studies. In Section 6, we conclude our work and propose future work based on our findings.

2 Background

In this section, we use previous work to describe what a startup is and why startups fail. The ideas from Lean Startup (Ries, 2011) are connected to academic studies and continuous experimentation. We provide a brief introduction to continuous software engineering and continuous experimentation. We describe two major models of continuous experimentation: HYPEX (Olsson and Bosch, 2014) and RIGHT (Fagerholm et al., 2017). Finally, we describe the current understanding of the adoption of continuous experimentation based on previous studies.

2.1 Startup

A startup is not just another company doing business. A startup is a specific kind of company with special features that make it distinctly different from traditional enterprises.

> A startup is a human institution designed to create new product or service under conditions of extreme uncertainty.
> —Eric Ries in The Lean Startup (Ries, 2011)

Ries' definition of a startup is useful when trying to understand what kind of a company a startup is and what defines their success or failure. First of all, the company is developing a completely new product. Importantly, building a new kind of product means that nobody else has done anything similar before. At the same time, the company does not know exactly what kind of the product it is building, and it is also unknown who is going to buy the product once it is built (Ries, 2011).

Startup statistics show that startups fail more often than they succeed. A recent combined report from Cerdeira and Kotashev (2021) states that 9 out of 10 startups fail. The success rate had actually improved from 11 out of 12 from 2019. Neither of the numbers promise a good future for a new startup. The same report states that 34% of the failures were due to failing to find product-market fit, 16% due to finance problems, and 18% due to team problems. Product-market fit, i.e. failure in requirements engineering, is the leading cause of product failure reported in previous studies as well (Klotins, Unterkalmsteiner, Chatzipetrou, et al., 2021). Ries (2011) has come to the same conclusion stating that companies often fail by succeeding to build the wrong product. This, he claims to be a result of trying to use traditional management methods in an environment that calls for new kind of management tools.

Ries proposes the Lean Startup (Ries, 2011) method for building successful startups. The method relies on five principles: entrepreneurs are everywhere, entrepreneurship is management, validated learning, build-measure-learn, and innovation accounting. The first principle suggests that the Lean Startup method should be applicable also outside startups. The second principle suggests that even though old management tools may not be applicable in startup environment, there needs to be some form of management nevertheless. The last three principles describe the

product development activities Ries claims to work well in the uncertain world of startups (Ries, 2011).

Validated learning is all about learning as much as possible. Ries (2011) claims that a company should aim to take every possible opportunity to learn more no matter what they do. He proposes that instead of assuming the state of world, the entrepreneur should validate the facts with scientific rigor.

Build-Measure-Learn (BML) is the core activity Ries (2011) proposes to use in his Lean Startup methods. The idea of BML is to build changes, measure the effects of the changes, and finally learn from the data. For example, change a color of a button, measure how often the user clicks the button after the change versus before the change, learn that the new blue button is clicked more often than a red button. With this new validated data at your disposal, the next cycle could include changing all buttons from red to blue. Ries (2011)) explains that the idea of BML is to have a fast feedback loop for validated learning.

To ensure a fast feedback loop, Ries proposes using the BML loop together with minimum viable product objects (MVP). The MVP objects need not be complete products, they can be the smallest possible change that can be put through a BML loop for validated learning. The idea of MVP is the polar opposite of the traditional product development practice of only allowing the complete product to be used by customers. Ries goes on and explains that these MVP items should be used to test the fundamental business hypotheses of a company. If the users like the MVP, they would probably like the finished product as well. If the MVP is not liked, the company may need to work on their hypotheses and find a MVP that produces better value for the customer i.e. pivot (Ries, 2011).

The last principle, innovation accounting, states that the a startup should be able to measure its progress. Ries introduces a number of metrics that should be used to measure real progress. He also warns against using vanity metrics that may look good on slides, but provide no actionable information. He proposes that split-tests together with cohort analysis should be used to properly compare two implementations. He also argues that, in the spirit of a scientific experiment, the split test experimentation should produce actionable metrics that are defined before the test has started. Ries also advocates that the data should be accessible and auditable. Anyone should be able to look at the data and verify that the data matches reality. Quality data allows the startup to do data based decisions (Ries, 2011).

Arguably, one major shortcoming of the Lean Startup (Ries, 2011) was that there were no proper instructions for setting up the model in actual startups. The book had many case studies, but mostly the method was describing the activities on a higher level instead describing the architecture needed to support the innovation accounting process. Running experiments was central part of the Lean Startup method, yet little guidance was offered for selecting the experiments to be run.

While the Lean Startup (Ries, 2011) is not an academic paper, the ideas behind the concept has been studied extensively since the book was published in 2011. Indeed, most of the ideas from the Lean Startup model can be found in the field of continuous software engineering (Fitzgerald and Stol, 2017) and continuous experimentation (CE) (Olsson and Bosch, 2014, Fagerholm et al., 2017). Camuffo et al. (2020)

conducted a randomised study and found evidence that the scientific experimentation approach is beneficial for startups. At last, the ideas from the Lean Startup were formalized (Fagerholm et al., 2017; Olsson and Bosch, 2014) and tested (Camuffo et al., 2020) with scientific rigor.

2.2 Continuous *

For a long time, software was built by first spending an extensive amount of time and resources in the planning phase where the complete product specification was carefully crafted and documented before the project moved into an implementation stage. The assumption of this big up front planning approach was that the skilled architects, business personnel, and software engineers would be able to specify how an ideal product would look like. Unfortunately, it turns out to be most difficult to know what kind of product the end users, or customers, actually want to use. Furthermore, some crucial aspects of the products may become apparent only during the implementation, requiring the initial plans to be amended. Maintaining the product specification and documentation takes up a lot of time and it had become apparent that the waterfall model was not compatible with the reality of constantly changing and emerging requirements (Sommerville, 2007).

Agile and lean methods emerged to replace the long lived waterfall model. While the waterfall model tried to avoid change of specification by all means, these new iterative models embrace change. Instead of planning the complete product up front, these new models only produce detailed plans for the immediate future. The work is done in short iterations, which typically last from one to four weeks. Iterations are planned, executed, and reviewed one after another. Ideally, there is a new functional version of the software available and deployed after each iteration. Relevant stakeholders participate in the planning and review of each increment and new knowledge gained from previous iterations is used when planning future work. Incremental development has two major advantages over traditional plan based development. Firstly, first versions of the product can be delivered to customers without needing to wait for the complete version. This allows customers to gain access to the most important new features early on and allows all stakeholders to validate or change initial requirements. Secondly, users get to use the system and provide feedback to the development team for each new increment of the system. Involving users early on helps with requirement validation and elicitation. Early users are committed to providing feedback in order to improve the product they are using (Sommerville, 2007).

Fitzgerald and Stol (2017) argue that while incremental development has made software development function iterative and continuous, other parts of the businesses are still functioning on a non-continuous, plan based, model. They propose a more holistic approach for developing software at an organizational level. They acknowledge that iterative agile methods are gaining ground from the long used plan based waterfall model. However, only applying the agile methods to the software development function is not enough. Business stakeholders are left out of the loop, and achieving flow throughout the organization should be considered instead limiting

the changes only to the software development and operations functions. Improving the overall communication and co-operation across the organization should enable the organization to be more sensitive to external signals as well as more capable to reacting to those signals.

Ideas from Beyond budgeting (Bourque and Fairley, 2014), Lean thinking (Poppendieck and Poppendieck, 2010), SaFe (Leffingwell, 2007), Lean Startup (Ries, 2011) can be used to convert the rest of the organization towards flow thinking instead of still operating in batched mode. Otherwise fine tuned software development and operations are limited by other parts of the organization operating on a much lower clock rate: typically a new release is produced at the end of a 2-4 weeks long sprint. While some practitioners can do multiple production releases each day, the consistency of the flow is more important than the absolute speed of the execution.

Fitzgerald and Stol (2017) acknowledge the many previously discovered continuous activities such as continuous integration (CI) (Kim et al., 2008) and later continuous deployment (CD) (Claps, Berntsson Svensson, and Aurum, 2015). They also identify and propose a number of other continuous activities, which they claim to be essential for activities throughout the organization: product development, automated customer feedback collection, business planning, and customer value creation.

Continuous experimentation and innovation are the base on top of which the continuous flow can be built on. Experimentation is proposed as the method for confirming the hypotheses behind innovation and improvement initiatives. The innovation and experimentation functions close the lean flow by feeding experimental data from operations into the business decision making process.

Fitzgerald and Stol (2017) introduce BizDev activities to close the disconnect between business, i.e., strategy and planning, and software development functions. This can be seen as analogous to DevOps (Ebert et al., 2016) that cares for the smooth co-operation of software development and operations stakeholders.

2.3 Continuous experimentation

Continuous experimentation is a process where an organisation conducts experiments in a cycle. As the organisation conducts experiments and collects experimental data, the organisation is expected to not only confirm or reject any old hypotheses but also build new hypotheses to be tested in the next cycles of experimentation (Auer et al., 2021).

2.3.1 Scientific experiment

Scientific experimentation is a basic tool of the scientific method. By running experiments, a researcher can study what kind of an effect a specific intervention has. For example, how adjusting the color of a link changes the likelihood of the user clicking the link (Hern, 2014). Experiments are good for providing causal descriptions, i.e., describing the consequences that can be linked to varying a treatment. While the consequence can be studied, an experiment is less useful when trying to explain what the underlying mechanisms causing the change are (Shadish, Cook, and Campbell,

2001).

There are multiple kinds of experimental setups available, each applicable for different kinds of studies. However, they all share the basic building blocks of an experiment. First, the researcher must have formed a hypothesis that they wish to validate or reject by running an experiment. The hypothesis must include the evaluation criteria for accepting or rejecting the hypothesis (Shadish, Cook, and Campbell, 2001).

Using the link color experiment as an example, the hypothesis could have been that the shade of blue used in the link has an effect on the user behaviour and it should be possible to find the best shade of blue in order to improve the chances of a user clicking the link.

Second, the researcher running the experiment should have as much control over the variables affecting the outcome as possible. The amount of control can change depending on the environment where the experiment takes place. Running experiments in a lab setup allows total control over variables. All experiments can't be conducted in a lab environment and the amount of control the researcher has over the variables can vary (Shadish, Cook, and Campbell, 2001).

The link color experiment setup was a split test setup where the researchers were able to only change the color of the link between different treatment groups (Hern, 2014). In this case, the researcher has some control over the service they own but they are still unable to control the environment where the user is interacting with the service.

Third, the researcher must have means of collecting data from the experiment. How measurements are collected varies on a case by case basis. Physical phenomena can be observed with sensors such as thermometers or microphones (Shadish, Cook, and Campbell, 2001). Use of software can be tracked e.g. by automatically logging user actions, observing the user when they interact with the software, and interviewing the users during or after they have used the service.

The link color experiment most likely used some sort of instrumentation build directly into the software service for automatic data collection. It would be expected that the generated data was likely quantitative in nature and saved in the form of user action logs.

Fourth, each experiment has some experimental units that are subjected to a treatment. The experimental units can be groups of people or chemical batches in a laboratory. Assigning treatments to test units and making notes of which unit receives which treatment allows the researcher to study the effects of applying, or not applying, one or more treatments to different test units (Shadish, Cook, and Campbell, 2001).

The link color experiment was described as a "1%" experiment where each shade of blue was assigned to a randomly selected 1% of their total user base. A total of 42 shades of blue was tested with the setup (Hern, 2014).

2.3.2 Timeline for CE research

Continuous experimentation is a product of several scientific advancements including the use of experimentation in product development (Thomke, 1998), the agile software development practices, and continuous software engineering practices such as continuous integration (Kim et al., 2008), continuous deployment (Claps, Berntsson Svensson, and Aurum, 2015). These advancements were behind the experimentation processes innovated by pioneering companies such as Microsoft (Kohavi, Henne, and Sommerfield, 2007), Netflix (Gomez-Uribe and Hunt, 2016; Amatriain, 2013; Steiber and Alänge, 2013). The research results, describing the benefits these large corporations had been able gain through their novel ideas, invited other researchers to study the underlying ideas and principles further.

Innovation Experiments System (Bosch, 2012) was one of the earliest studies attempting to describe the new paradigm with proposing a process with alternating development and data collection stages. Soon after, the Stairway to Heaven (Olsson, Alahyari, and Bosch, 2012) was published describing the evolution path from an organization using traditional development towards the new ideal form where the company as a whole works as an experiment system.

Two years later, Olsson and Bosch (2014) published the HYPEX model, which described a detailed experimentation process model spanning the complete product development cycle. The HYPEX model is depicted in Figure 1. The figure illustrates well how the experimental data is fed back into the strategic decision making to complete the flow.

The HYPEX model uses minimum viable features (MVF) as experimentation objects when implementing features from backlog. Using MVF approach, only a minimal implementation of a feature is built in order to validate it's applicability. The MVF is then implemented, data measured after deployment, and gap analysis used to determine the next course of action. Finally, the experimental data is fed back into the business strategy function. The HYPEX model does not include description of the experimentation infrastructure, and does not describe the roles of different stakeholders in detail (Olsson and Bosch, 2014).

The RIGHT model by Fagerholm et al. (2017) provides an even more detailed description of an experimentation framework. The model includes for the first time a detailed description of experimentation framework and the associated experimentation process.

The experimentation process block of the RIGHT model can be seen in Figure 2. The experimentation process is started by using previous learnings as the basis for hypothesis building. During the hypothesis formation, the company strategy and identified assumptions are combined with the data. Unless the company strategy is kept in mind during the hypothesis formation, there is a risk of doing experimentation that does not align well with company goals. After identification, the hypotheses are prioritized and sent to experiment implementation phase (Fagerholm et al., 2017).

During the implementation phase, the software development team builds a MVP version of the new feature, and adds any required instrumentation for experimental data collection. Simultaneously, an experiment for validating the hypothesis using

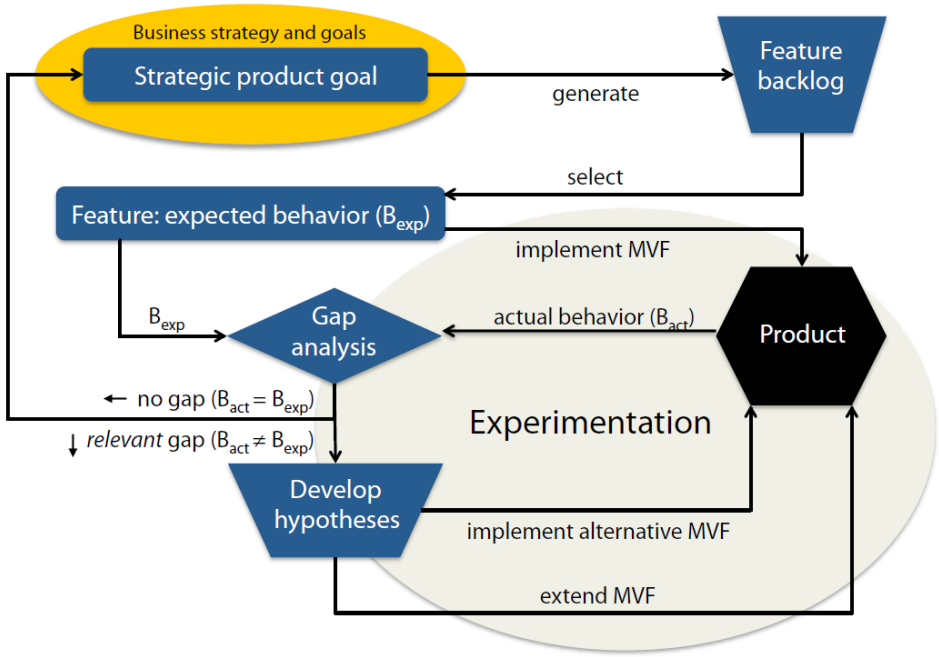

Figure 1: HYPEX model by Olsson and Bosch (2014).

the MVP implementation is designed. Then, the experiment moves into execution phase where the MVP is released to the target audience and usage data is collected based on the experiment design. The final phase of the experimentation is analyzing the data for decision making (Fagerholm et al., 2017).

The decision making is a strategy level choice that is made based on the experimental data. Much like the Lean Startup (Ries, 2011) process, if the hypothesis is validated, the company chooses to persevere on their current path, and proceed with full implementation of the planned feature. If the experiment does not appear to support the hypothesis, the company can either adjust the strategy, i.e. pivot, or choose to update their assumption based on the new information. After a decision has been made, the learnings are packaged for later use and a new round of experimentation can start by analysing any new and previous learnings (Fagerholm et al., 2017).

The experimentation infrastructure of the RIGHT model can be seen in Figure 3. The infrastructure describes the different roles, and associated tasks, needed to do continuous experimentation based on the RIGHT model.

Multiple roles and associated tasks are described in the RIGHT infrastructure model. The model is an ideal description of all the different responsibilities that need to be taken care when doing continuous experimentation. One individual can fulfill multiple roles as long as the tasks are taken care of. This can be seen in the case descriptions in Fagerholm et al. (2017). The same goes for technical infrastructure, in an ideal world, the organization would have all the described technical capabilities and services in place. However, especially when a company is just starting to use

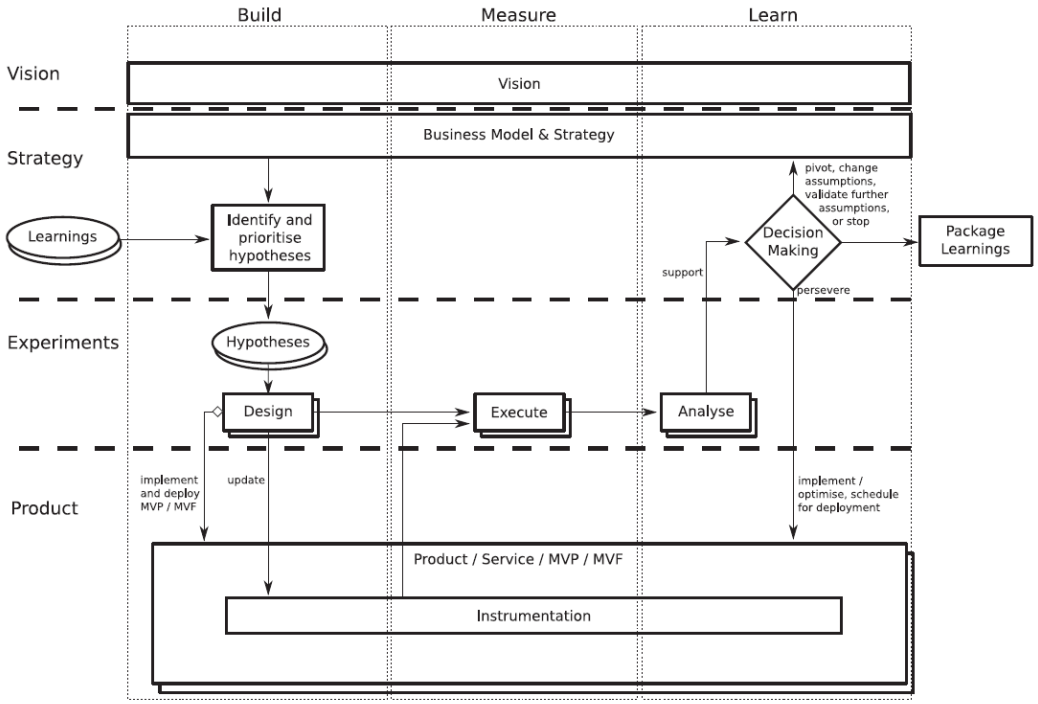

Figure 2: RIGHT process model by Fagerholm et al. (2017).

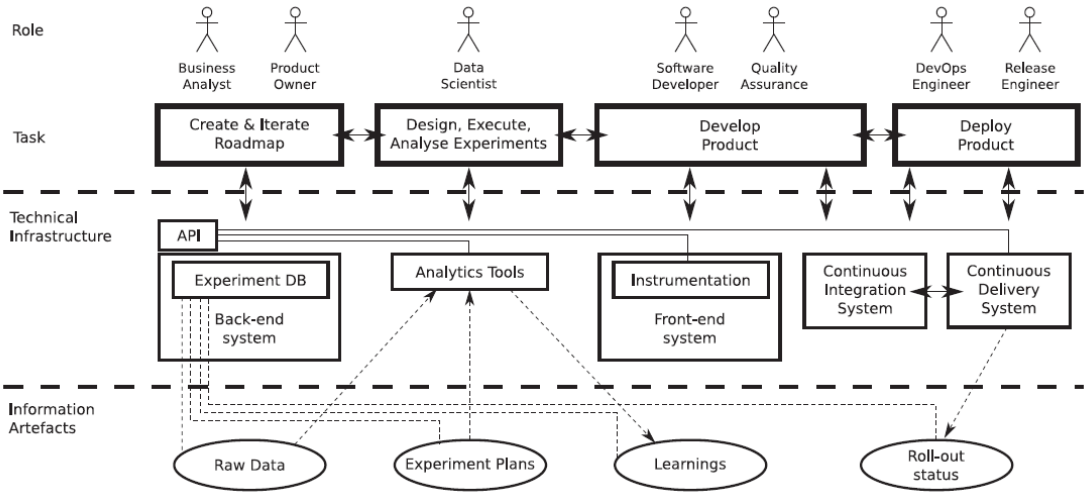

Figure 3: RIGHT infrastructure model by Fagerholm et al. (2017).

experimentation, it is possible to complete full experimentation cycles without setting up all described infrastructure. The full implementation of the infrastructure becomes more critical once an organization moves into truly continuous experimentation mode where multiple experiments are being executed simultaneously (Fagerholm et al., 2017).

2.3.3 Controlled online experiment

Controlled online experiment (Fabijan, Dmitriev, Holmstrom Olsson, et al., 2020) is one of the most common experiment designs. These are often a A/B-test where two different test groups receive a different treatment. Often one of the groups receives the new version of a feature, and the other group receives the old version. The treatments are usually assigned randomly to the target groups and statistical analysis is used to determine which of the tested implementations produced better results.

2.3.4 Current state according to research

Being born from industry, large enterprises such as Google (Steiber and Alänge, 2013) and Microsoft (Kohavi, Henne, and Sommerfield, 2007) have been using continuous experimentation even before the research community had a name for the practice. When it comes to smaller companies, the research indicates that the practice is not widely adopted. Lindgren and Münch (2016) reported that experimentation in general was a common practice, but it was rare to see systematic experimentation. The startup data from Klotins, Unterkalmsteiner, Chatzipetrou, et al. (2021) shows that the startup companies were often not even using the prerequisite technologies of continuous experimentation such as continuous integration, automated testing, and continuous deployment.

A major obstacle in the adoption of continuous experimentation appears to be the advanced technological and organizational capabilities needed to adopt the practice. Auer et al. (2021) divides identified problems in to six different categories: 1. Cultural, organizational and managerial challenges, 2. Business challenges, 3. Technical challenges, 4. Statistical challenges, 5. Ethical challenges, and 6. Domain specific challenges. The sheer number of problem categories indicates that the continuous experimentation research community still has a lot of work to do.

The adoption of continuous experimentation is covered in earlier studies by e.g. Yaman (2019), Olsson, Alahyari, and Bosch (2012), and Lindgren and Münch (2016). All of the studies acknowledge that using continuous experimentation requires significant skill and coordination from the whole organization. No accessible easy to follow guide for adopting the practice has been published to date, and the general consensus appears to be that budding practitioners should start with small scale experimentation and scale out as they learn from previous experimentation rounds.

3 Methodology

This thesis uses a descriptive multiple case study approach (Yin, 2012, p. 49) to study the product development tools and practices used by the five case companies. The primary data collection method of the study is semi-structured interviews. Each company is treated as an individual case, and the the topics of interest is the tools and practices used, the use of experimentation in product development, and the factors affecting the selection of new tools and practices. The findings from different cases are compared against each other in order to find similarities, differences and common patterns.

3.1 Research questions

This study aims to answer the following research questions.

- **RQ1 What product development practices are the startup companies in the A Grid and Maria 01 communities using?**
- **RQ2 How do startup companies in the A Grid and Maria 01 communities use data in product development?**
- **RQ3 What factors affect the adoption of continuous experimentation in startup companies in the A Grid and Maria 01 communities?**

Answers to RQ1 and RQ2 are compared with the results of the previous studies by Lindgren and Münch (2015), Klotins, Unterkalmsteiner, and Gorschek (2019), and Klotins, Unterkalmsteiner, Chatzipetrou, et al. (2021). While the study is interested in all reported product development practices and use of data in product development, the focus is especially in experimentation related practices.

RQ3 attempts to provide an explanation for the data from RQ1 and RQ2. Finally, based on the answers to all three research questions, the readiness to practice continuous experimentation is evaluated. The RIGHT model by Fagerholm et al. (2017) is used as the reference model of continuous experimentation, as it is considered to be the current state of the art model of continuous experimentation by Auer et al. (2021).

3.2 Case study

Case study is a qualitative research approach where the unit of analysis has clear boundaries. This focus on a single case unit instead of the overall phenomena is what differentiates case study from other forms of qualitative research. Merriam and Tisdell (2015) define case study as ”an in-depth description and analysis of a bounded system”. A case study with a descriptive design is used to provide as systematic description of the bounded system. The description is expected to contain facts and characteristics of the system. In the context of this study, the objective is to describe the companies and the their practices in order to answer the research questions.

Similar to other qualitative research approaches, a case study can use different kinds of data sources to collect a rich set of data about the case subject. Three most commonly used methods for data collection are interviewing, observing, and data mining. Interviewing is often used as the single method of data collection within a case study, which is also the case in this thesis. Interviewing allows researchers to extract tacit knowledge about past events and experiences as recollected by the interviewee (Merriam and Tisdell, 2015). Interview as a research data collection method can be described as "a conversation that has a structure and a purpose" (Kvale and Brinkmann, 2015, p. 5).

3.3 Case companies

Case companies selected for the study are startup companies from Maria 01[1] and A Grid[2] startup communities. Maria 01 is located in Kamppi, Helsinki and has over 175 member startups residing in house. A Grid is located in the Otaniemi area in close vicinity to Aalto University campus in Espoo. There are some 150 startup members in the A Grid community.

Selection criteria for case companies was that they must be a member of either the Maria 01 or A Grid community and have a software based product or service. Selecting case companies from two different locations allowed a greater variety of companies to be interviewed and allowed the company location to be used as an extra data point. It was expected that company location could play a role in shaping the product development processes of companies as co-located companies may exchange information about their ways of working.

Convenience sampling was used to find the case companies for the study. Startup companies can be expected to utilise different kinds of product development practices. While numerous models and tools have been proposed to help companies find the product-market-fit, no two companies are likely to operate in an identical fashion. Furthermore, the company selection criteria allowed companies of any stage and companies with different kinds of products to all be included in the case pool. Overall, having quite relaxed selection criteria provides a heterogeneous set of cases and qualitative data with high variability. All case companies with their business models and description of their current stage are listed in Table 1.

3.3.1 Company A

The first case company was a mobile application company established in late 2017. First beta of their application was released 18 months ago and the 1.0 version of the application was released 3 weeks before the interview. The interviewee was one of the company founders and their job title was product owner. Initially, the interviewee had been working part time on the startup while still working full time on their previous job. They've been working full time at company A for 2 years at the time of the interview.

[1]Maria 01 startup community. URL: https://maria.io (visited on 12/18/2021)
[2]A Grid startup community. URL: https://agrid.fi/ (visited on 12/18/2021)

ID	Location	Business model	Distribution channel	Stage
A	Maria 01	Free app, sell ads	Mobile app	Version 1.0 launch in progress
B	Maria 01	B2B (provision)	API as a service	Released
C	Maria 01	B2C (subscription)	Side-loaded application	Early access program live
D	A Grid	B2B (provision)	SaaS	Live for over a year
E	A Grid	B2C (subscription)	Physical	Live for over a year

Table 1: List of case companies.

The business model of company A was to provide a free mobile application for users and sell ad space for third party companies. The mobile application had three key features. Firstly, the application contained a wealth of information about different aspects of climate change. Secondly, the users could find out ways to make more sustainable choices as well as log their actions for calculating the impact of their choices. Finally, users could receive deals for purchasing sustainable goods and services.

The mobile application was distributed to end users through official application stores, i.e., Apple's App Store[3] and Google's Play Store[4]. The in-app advertisements and partner provided content were sold by directly contacting potential customers. At the time of the interview, the company did not have an online service for their business customers.

The core team consists of a chief executive officer (CEO), chief communications officer (CCO), product owner (PO), a sales trainee, and two software developers. At the time of the interview, an advisor with previous experience of going to market with a new product was working more actively as a consultant to help out with the product launch. The team did not have an in-house designer and design work was bought from an outside design shop when needed.

3.3.2 Company B

The second case company was a financial services company established in 2017. The company had multiple products out in the market and sells their services to cryptocurrency trading platforms. The interviewee was the chief operations officer (COO) of the company who was in charge of the commercial side of the company as well as the watching over for the commercial aspect of the company's product business. Their involvement in the product business was explained by the fact that the company did not, at the time of the interview, have designated product managers on their payroll. At the time of the interview, the interviewee had worked for the company for 4 months. Before joining the company, they had been solving business problems for 10 years as a consultant.

The business model of company B was to build financial services and investment facilities and sell them with a software a service (SaaS) model. Their customers were trading platforms, institutional investors, and custody platforms. Trading and custody platforms integrated the services provided by the company B into their own services. Ultimately, allowing end users to access company B provided services through a familiar platform. The distributor model allowed the company to gain access to a large customer base as the business partners were able to sell company B products to their existing customers.

[3]Apple App Store. URL: https://www.apple.com/app-store/ (visited on 12/28/2021)
[4]Google Play. URL: https://play.google.com/store (visited on 12/28/2021)

3.3.3 Company C

The third company was a drone piloting software company established in March 2020. The company had built a pair of applications, one for Android devices and another for Microsoft Hololens. Together the applications provide a new, augmented reality based, way for drone pilots to fly their drones. The applications had been in development since March 2020 and had been launched a month ago as an early access program for around a dozen first stage customers. The interviewee was one of the company founders working as the chief technology officer. Before founding the company, they had been working as a software consultant. Their daily responsibilities included facilitating the work of the development team, maintaining the company roadmap together with the CEO, and trying to write code as many hours as possible.

The business model of the company was a subscription model where users pay monthly license fees for access to the software. The target audience for the product was expert drone pilots who fly drones professionally and would benefit from the advanced AR features the company is planning to bring into their product. The company expected their target audience to work professionally with drones and perform flights in order to e.g. inspect industrial facilities or large constructs such as bridges.

The team had a CEO, CTO, two in-house developers, and a data-analyst. The CEO and CTO shared the responsibilities of a missing product owner. The CEO had previous user interface design experience and acted as an in-house designer when needed. Both the CEO and the CTO were trying to write as much code as their other responsibilities allowed. Additionally, the company was using an external consultant who had been developing the company's product since before the company had hired its first developers.

3.3.4 Company D

The fourth company was a company building a commerce platform for selling access established in June 2018. The company provided a suite of software solutions for rental companies. On the commercial side, the company provided a white label online shop for rental businesses. Additionally, the company provided software for inventory management, order management, customer data management, and payment processing. The suite of products complemented each other and customers were able to select which parts of the company offering they want to take into use. The interviewee was one of the company founders working as the CEO and a member of the company board. They had been previously actively working on operations level when the company was at an earlier stage. However, at the time of the interview their work at the company had become more about leadership and management than hands-on operations.

The business model for the company was to provide their software solutions as a SaaS product and take a percentage commission of the sales done through their platform. There was a freemium version of their core product available for allowing new customers to try out the product before purchasing a more advanced feature set. The company customers come in all sizes from large international companies to

individual entrepreneurs.

The team was roughly 20 persons strong and had a total of three product development teams. A product development team was described as a cross functional unit that had a product designer and one or more solution engineers. A solution engineer's core skill was software development. However, they were also expected to understand how software was related to the commercial context of the businesses that buy the product.

3.3.5 Company E

The fifth company was a company providing care services for elderly persons and was established in 2018. The company's primary product was a physical service. The production of the physical service was supported by internal software services. At the time of the interview, the company had limited online services available for their customers and care reports were sent via email. The company was planning to enhance their service by building mobile applications where customers could initially browse care reports and images as well as manage their subscription. The interviewee was leading the company's internal software development team and was working under the title lead software engineer. In addition to software development, they were responsible for recruiting new developers as well as handling any operational tasks that affect the software development team.

The business model of the company was to provide the physical service on a subscription model basis. A typical customer was a next of kin who hired the company to visit their elderly parents on their behalf. The actual services provided during the visits varied from doing housework, going shopping, or just keeping company. Importantly, the paying customer was typically a different person than the one who receiving the service.

The company had multiple employees running the operational side of the business. However, the software development team consisted of two developers and a designer. The founders were described as not technical and having very little experience of software product development.

3.4 Data collection

The primary data collection method used in this thesis is expert interviews. A single individual from each case company are interviewed for approximately one hour during which following topics are discussed: the current stage of the company's product, the software development practices currently used within the company, releasing of new versions of the product, evolution of the development and release practices during the lifetime of the company, and the collection and use of user feedback and data.

Semi-structured interview was selected as the type of interviews as each startup is expected to conduct their business and product development in unique ways. For example, Klotins, Unterkalmsteiner, Chatzipetrou, et al. (2021) reported that startups are likely to use agile software engineering and product development practices selectively. Merriam and Tisdell (2015, p. 110) propose using semi-structured interviews

when the topic and the expected data is not well understood before starting the data collection. Using an interview structure with open-ended questions allows the interviewee to bring up topics and pieces of information that the interviewer could not have foreseen when sketching the interview guide. Furthermore, the semi-structured format allows the interviewer to ask follow-up questions regarding any unforeseeable piece of data that is deemed important enough during the interview.

An ideal interviewee has a broad understanding of both the technical as well as the business aspects of the case organisation. In a more established company, a chief technology officer, product manager or other high ranking employee would be a good interviewee candidate. However, such designated roles may not exist in all startups, and the interviewees were selected based on the judgement of the first contacted person of each case company. The job titles of interviewees for companies A-E were a product owner, COO, CTO, CEO, and lead software engineer respectively.

Interviews for companies A, B, and C were conducted face to face and interviews for companies D and E were conducted on a Google Meet call. Each interview was recorded in order to produce a transcription for later analysis. Two separate devices were used for redundancy in case one of the recorders should malfunction during the interviews. The data of companies A, B, and C were complemented with short additional interviews in order to fill gaps identified in the original interview data. For company A, a software developer answered additional questions about the software build pipeline and testing over instant messaging. For company B, the CTO of the company was interviewed briefly face to face regarding the software development processes and tools used. For company C, the original interviewee answered additional questions about the software build pipeline and testing using instant messaging.

The interview guide was based on the interview guide used in Lindgren and Münch (2015). Their interview guide covered the following three main topics: current software development practices, current practices of customer feedback elicitation and use, and future practices of customer feedback elicitation and use. The three topics were deemed suitable for collecting data about the current software development practices and the use of data in product development. The existing topics were extended with questions about how the companies had selected the tools and practices they were using. Also, continuous experimentation was added as a distinct topic that would be explored more deeply in case the interviewee appeared to know about the topic.

Example interview questions included ones such as “How do you make sure you are building the right product?”, “Are you familiar with the term continuous experimentation?”, “How have your software development practices changed during your time?”, and “Can you describe the process of releasing a new version to production?”. The interviewer would proceed deeper into the topic with followup questions such as “What kind of decision was made based on the results?” in case the interviewee had indicated that the company had conducted some form of an experiment.

The interview guide is provided as the Appendix A. While the interview guide was prepared in English, all interviews and followup questions were done in Finnish.

The rationale for using Finnish was that all of the interviewees were native Finnish speaking individuals and the semi-structured interview approach allows non-exact wording of questions. Also, the interview data consisted mostly of factual statements, which are easily translated from Finnish to English for analysis and presentation of results.

3.5 Data analysis

This section describes the different stages used in analyzing the interview data. The data analysis started with converting the interview data into textual representation. This was also the only linear part of the data analysis process. The later stages of coding and analyzing the interview data was performed multiple times as analyzing data from one case company would often provide a new perspective into the data of other companies as well. Multiple passes allowed the researcher to recognize and build higher level categories from topics that appeared repeatedly across case companies.

3.5.1 Transcribe

Data analysis started by producing an anonymised transcription of each interview. The transcribing process was fully manual as the author did not find a suitable tool that would allow converting Finnish language audio into text. VLC Media Player[5] was selected as the tool of choice for replaying the audio recordings. The player supports setting up global hotkeys for controlling the playback without moving focus away from the text editor. Overall, the quality of the recordings were satisfactory and all five recordings were successfully transcribed without losing any parts of the interview. All interview audio materials and the produced transcripts were in Finnish.

3.5.2 Coding

Once interviews were converted into textual representation, the transcriptions were loaded into ATLAS.ti[6] software for coding. Open coding approach was used to code the interview data. While there are some standard taxonomies such as SWEBOK (Bourque and Fairley, 2014) categories, using open coding allows the researcher to focus more on the data itself rather than trying to fit the data into a previously established frame (Merriam and Tisdell, 2015). Building codes from ground up was also supported by the premise that each of the case companies would be unique and the semi-structured interview format made it difficult to predict what kind of data the interviews would contain. Codes produced from the first passes of the data were found to be long and descriptive, and multiple coding passes were used to group similar codes under a single code. Codes were linked together by prefixing the code with the higher level code, separated by colon. The screenshot from ATLAS.ti project

[5]VLC media player. URL: https://www.videolan.org/vlc/ (visited on 12/28/2021)
[6]ATLAS.ti. URL: https://atlasti.com/ (visited on 12/28/2021)

in Figure 4 shows the technique in action. Notice that the coding is done in English even though the transcriptions are in Finnish.

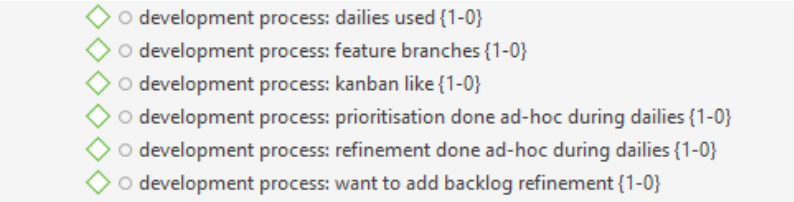

Figure 4: Higher level codes as prefixes.

3.5.3 Build categories

Higher level categories were built based on themes that appeared repeatedly during multiple interview transcriptions. Importantly, the analysis started without any predefined categories and not every coded piece of data is forced into a category. Quotes were added to categories by additionally coding the quote with a category specific code such as "C1 previous experience". The screenshot in Figure 5 shows how this method allows the quote to retain all previous coding information.

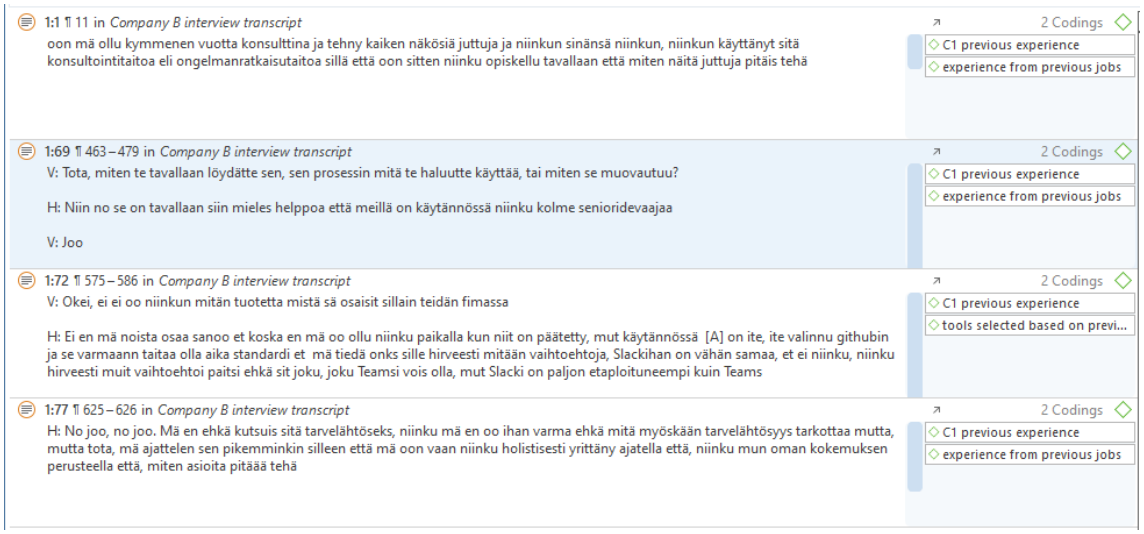

Figure 5: Quotes belonging to a single categor.

3.5.4 Analyze cases as individuals

After completing the coding and building of categories, each company was analysed as an individual case. Specifically, the software and product development tools and practices used by the company were identified from the coded interview data. Then, the practices used for collecting and using user data were extracted for closer inspection. Then, the flow of data within each company was converted into a

graphical representation, which allowed the researcher to build a complete picture of the organization and the interactions between different stakeholders.

The visual representation of the case organization was then compared with the idealized experimentation organization by Fagerholm et al. (2017). The comparison was the basis for evaluating how far from being capable to practice continuous experimentation.

The case company's position on the Stairway to Heaven (Olsson, Alahyari, and Bosch, 2012) was also determined based on the interview data. The evaluation was based on the identified organizational and technical capabilities. The software development capabilities covered the company's technical capabilities for making changes to their product and publishing new releases. The organizational capabilities included evaluating how well the organization is able to use experimental data from new releases in their product development.

Finally, all company specific problems related to product development, data collection, data use, and experimentation were analysed and compared against results from the prior studies. The reported problems were also studied as the potential factor behind the selection of tools and practices.

3.5.5 Analyze combined data

After completing individual case analysis, the combined data from all cases was analyzed. First, the tools and practices used by the companies were summarised into a summary table. The summarised data was then used to discover shared practices as well as to identify case specific outliers. The combined data was then compared against the previous knowledge from Lindgren and Münch (2016), Klotins, Unterkalmsteiner, and Gorschek (2019), and Klotins, Unterkalmsteiner, Chatzipetrou, et al. (2021).

Finally, the coded interview data from all five cases was coded again in order to common themes that would permeate the whole data set. At this phase, the analysis focused on uncovering the categories that could have been used to explain why the companies have selected the tools and practices they were using.

4 Results

In this section, our case companies are first described based on the interview data. The analysis of individual companies indicated that all of the companies were using agile or lean software engineering practices. Companies appeared to be reluctant to establish well-defined processes and systematic experimentation was found to be rare. Previous experience, limited resources, and the perceived value a tool or practice can provide in short term were found to be affecting the adoption of new tools and practices. Continuous experimentation appeared to have shortcomings with all three of the identified factors.

4.1 Case company descriptions

The following subsections describe the case companies based on the interview data. The descriptions do not embody the complete interview data and their primary purpose is to provide a wider context for the reader.

4.1.1 Company A

Company A's product was an educational mobile application that contained information packages related to a topical theme. The target audience was the large number of people passionate about the theme. The content provided through the application varied from educational information packages to actionable tasks the users could be inclined to complete. The application was free for download and the business model was to sell in-app advertisements and sponsored content spot. The value proposition was for the companies to be able to find the specific user segment, and the users to find the content they would care about.

Company A interviewee was one of the founders who worked as a product owner for the company's mobile application. The application had just moved out of beta and the 1.0 version was released just a few weeks before the interview. Company founders did not have prior experience of going to market with a new product. Therefore, one of the board members was working as an advisor for the team during the launch.

The software development team consisted of two in-house software developers who had joined the company after the development of the application had been started by an external consultant. The application had been in beta for a prolonged time as the consultant had not been available for Company A for more than a couple of days per week. The software development team nor the founders themselves had no prior experience of developing mobile applications, which had forced them to rely on the help of the previously used consultant when the team encountered a problem they couldn't solve themselves.

The product development team used two-week sprints to build the application incrementally. No specific incremental software development model or process was mentioned by name. The product owner prepared an initial plan for each sprint planning session and shared the plan with developers prior to actual sprint planning. This way the developers were able to raise any technical matters relevant to the plan

prior to the actual sprint planning. During the actual sprint planning session, new work items were added to the backlog. The team used Asana[7] work management software for managing their backlog. The team also used GitHub[8] for source code management. The interviewee explained the two services, Asana and GitHub, were integrated to automatically sync work items. Each sprint was concluded with a sprint demo session. There was no clear definition of done mentioned: some tasks could have required an explicit approval from the product owner before being included in the next release while other tasks were allowed to be included in the next release once developers were happy with the changes.

> It depends on the task because often we can push something into production before the end of the sprint...practically, if I have not requested a separate confirmation, our developers can push the changes in the next update.
> —Product owner, Company A

The new release process could have been initiated by the product owner or the development team. The product owner typically asked the developers to create a new build after a certain work item was completed. However, the developers could also propose a new release to be done when they felt like there are enough changes to justify a new release. While the developers could propose a new release, the product owner appeared to ultimately decide when a new release was made. The team used update branches for collecting release specific changes into a staging branch before an actual release. The features were developed in separate feature branches the feature branches were merged into update branches after code review. Merging into the main branch of the backend service triggered a CI and CD pipeline for automatic production releases. The mobile applications were built manually by the developers. Release candidates were tested manually by the product development team and the team did not have any automated test cases. Lack of automated testing was not seen as a problem as the team had not yet encountered serious quality issues. The team was also heavily prioritizing feature development and the interviewee mentioned that prioritization was necessary due to very limited funding. In case the release contains major changes to functionality, everybody working at the company was asked to have a look at the new release candidate before going live. Testflight[9] and Google Play[10] beta track were used to distribute release candidates to test devices. Once the team was confident that the build was good, the changes were pushed to production.

The company collected both implicit and explicit user data. The application was instrumented to log some user actions, and the team was actively seeking to improve their data collection capabilities. The lack of previous experience in mobile application development was mentioned as a major bottleneck when trying to acquire all the necessary data. The team initially had trouble understanding what data was possible to get, which tools were worth using, and what data was valuable to get.

[7] Asana. URL: https://asana.com/ (visited on 12/28/2021)
[8] GitHub. URL: https://github.com (visited on 12/28/2021)
[9] Testflight. URL: https://developer.apple.com/testflight/ (visited on 12/28/2021)
[10] Google Play. URL: https://play.google.com/store/ (visited on 12/28/2021)

Explicit user data was collected through feedback form, which was linked to from inside the application, company website, email newsletter, and company's Instagram page. Application store reviews were also mentioned as a regular channel for feedback. There was an in-app call to action for reviewing the application in-store. The product owner reviewed the collected data on a weekly basis and presented their findings during weekly meetings. The product owner would have wanted want to understand better what the user is doing in the real world as they use the application. The product owner would have also liked to have the capability to chat with individual users in order to gain better insight on how and why they used the application.

The interviewee was not familiar with the term continuous experimentation. However, use of experimentation was reported when developing the application's onboarding experience. The product owner had conducted user tests where they observed new users interacting with the application. The product owner was able to confirm an initial hypothesis that the users were overwhelmed by the amount of content in the application and kept scrolling the top-level content listing instead of navigating deeper into the actual content. Also, the poor visibility of the application menu was confirmed during user testing. The visibility of the menu was then improved and a new round of user testing was conducted with the new design. The data indicated a clear improvement over the old design. In this particular case, the team had planned to do a follow-up round of user testing. The interviewee reports that most of the development they do is some form of a test. However, it is was not clear if these tests would qualify as planned experiments.

> Well almost everything we do is maybe more like tests.
> —Product owner, Company A

It was not clear if hypotheses and measurable goals were set before starting the development of any particular task. The product owner did mention that the company had taken a feature out of the application after an initial version was developed. The company had assumed that their users would want to communicate with other users of the application and had therefore developed a messaging forum functionality into the application. However, the forum did not gain traction and users stopped providing content to the board. The forum was then removed from the application instead of being developed further.

4.1.2 Company B

Company B had built financial services and HTTP application programming interfaces (API) that they provide as white label products for financial platforms. First of the company's two main products had been launched 18 months prior to the interview and the second one had been available for 6 months at the time of the interview. The company offered the products as white label services for other businesses. The B2B model allowed other businesses to integrate Company B products into their own platforms where their existing users could access the Company B developed services through a familiar interface. While the Company B services hosted end-user data, the company did not have direct access to the end customers of their services. One

exception was a lending product targeted directly for a small number of institutional investors, which was not covered by the interview.

Company B interviewee was the COO of the company who had joined the company four months prior to the interview. His responsibilities were leading the commercial activities of the company as well as looking after the business side of product management. The interviewee had worked 10 years as a consultant prior to joining the company. They did not have any prior experience of working in product management and had been studying the relevant topics after joining the company.

The software development team was 5 developers strong after 3 developers were hired just days before the interview. 3 out of the 5 developers were considered to be senior. Additionally, the company had 2 employees working on commercial activities. Based on the company website, the founders and the first employees each have over 10 years of experience of building financial services and software.

The COO stated that the company maintained a high-level roadmap 6 to 12 months into the future. More detailed plans with implementation details had been done for "a month or two" (COO, Company B). The COO was personally responsible for maintaining the roadmap of one of the company's main products. When discussing roadmap management, the COO said that the main reason for adjusting future plans was that new data had come in and previously planned features were no longer seen as necessary: "It is more like continuous prioritization. At the beginning you have ideas... then little by little you learn... deadlines are closing in and you have to prioritize or adjust the scope". The COO explained that the product technology team had a biweekly session where the current plans, and any needs for adjustment of plans, were discussed. Overall, the company did not have an established or well-documented process for maintaining the roadmap. "Because we are such a small team, we do not need any heavy coordination processes" was the interviewee's justification for the way the roadmap was managed.

The development process used by the developers was set up by the developers themselves and the interviewee was unable to describe the process beyond stating that they did not use sprints. A quick follow up interview with the CTO was conducted in order to better understand how the company developed its software. The CTO explained that the team used Kanban (Poppendieck and Poppendieck, 2010) process to manage their work. The backlog and Kanban board were maintained virtually using the Trello[11] collaboration tool. Source code was pushed to GitHub code repositories. Code was developed on separate branches and GitHub pull requests were used to review the new code before merging to the main branch. New release candidates were built automatically by CI/CD pipeline with unit testing and automatic deployment to the integration environment. Finally, production releases required manual confirmation. The CTO claimed the team published new production releases on a daily basis.

The COO described weekly discussions with current and potential customers as the most used way to collect customer data. There were some limitations to the data the customer businesses are willing to share as well as actions they are willing to

[11] Trello. URL: https://trello.com/ (visited on 12/28/2021)

take in order to help Company B contact end-users. For example, the company had planned to do a survey in order to better understand their end-users. Even though the initial response from the partner businesses was positive, the survey was never made due to the lack of support from the partner business. The interviewee speculated that their partners may lack any real incentive to actively push the initiative on their end. The interviewee explained, "it has to be actively pushed through there if we want to get it done". They explicitly reported that not having direct access to end-users makes it difficult to contact end-users. The end users interacting through a third-party user interface prevented the company from collecting any user data outside the API calls made into their own systems. Effectively, the only user data the company was able to access is the data that is generated as a result of their business partners making calls to the company APIs.

When discussing how the interviewee would like to enhance their data collection, they mentioned that hiring a business intelligence person to do competitor benchmarking was probably the first action they would take. They saw competitor benchmarking would allow them to better understand what kind of products customers actually want as well as what kind of pricing is competitive. Another area of interest was getting to know how much wealth customers held in order to understand how much room for growth there was in the present customer base. The interviewee had asked for access to the wealth data, but the request was denied by their business partners. While there were cases where business partners were not willing to help out with user surveys and were reluctant to share fine-grained data, the interviewee indicated that was not always the case. They had received higher-level data such as the aggregate amount of individual currency held on their business partners' customers. Additionally, the business partners had passed forward feature requests from their customers. Eventually, data indicating strong demand from the market had led to a new product being launched by Company B.

At the time of the interview, the COO was responsible for processing customer data from customer meetings and ensuring that technical personnel can understand customer needs. The COO filtered incoming feature requests and coordinated with the CTO before biweekly tech meetings. Sometimes highly technical issues were discussed, which required the COO to consult a technical expert in order to understand the technical implications of specific requests before deciding the best course of action. They reported that "On the other hand, having a small organization allows you to always know who is the best expert in the subject". In addition to the biweekly tech meetings, the company had weekly commercial meetings where commercial issues and e.g. data from customer relationships management (CRM) systems were reviewed. Overall, the company did not have well-established and documented processes for processing data or making decisions. The interviewee described the way of working by saying "I have not worked in a startup before, but I would imagine this is a quite typical early-stage state, a kind of creative chaos". While that was the case at the time of the interview, the COO was actively working to convert the implicit processes into more tangible documentation.

The interview data did not indicate frequent experimentation taking place in company B. There was as single rate adjustment experiment being conducted at the

time of the interview. The interviewee explained that changing the rate was the only parameter they can adjust, and they were looking forward to seeing what kind of effect the rate adjustment would have in the withdrawals and deposits. The rate adjustment experiment did not have multiple test units, and all customers received the same treatment. Changing the rate had required coordination with the business partners and the nature of the market had made it economically unfeasible to offer different prices on different platforms. The company felt they would have been able to help their business partners with building better customer experiences. However, it appeared the business partners were not willing to let a third party company touch their service.

4.1.3 Company C

Company C's product was an augmented reality application for professional drone pilots. Their product allowed drone operators to observe the drone, real-world objects, and any additional information rendered into their field of vision. The business model was to license the application with a monthly subscription fee, which the interviewee claimed to be the de facto standard way of selling applications in the drone market. At the time of the interview, the company had started to offer the application to first customers through an early access program (EAP).

Company C interviewee was the CTO, and a co-founder, of the company. They had been working for the company since it was founded in March 2020. They were responsible for setting up development practices, selecting the appropriate technologies, and making sure that the development team is able to do work that improves the company's product. The team appears to be somewhat experienced in building applications. The interviewee claimed that the team being familiar with Unity development was the reason for choosing to build their product using the Unity engine. Furthermore, the CEO had been working as a user interface designer before founding the company.

The software development team consisted of the CTO and CEO, an external consultant, and two or more newly hired developers who had been recruited after the previous fundraising round. The CTO was responsible for coordinating and facilitating the work of the development team. The CTO and CEO had many other responsibilities at the company but were still writing code whenever possible. Working with new hardware, the team had had to perform some initial research in order to successfully stream live video from the flying drone to the AR headset via the companion Android application. Even though working with partly unfamiliar technology, the team had been able to successfully build an initial version of their product.

The interviewee mentioned that the company did maintain a roadmap. The content of the roadmap were split into two categories. First category was the bare essential features required for flight, which where being validated during the EAP program. The second category was all the experimental features the team was planning to build after validating the basic functionality. The current development work has been focused mostly on resolving fundamental issues regarding the technical

viability of the product. At the beginning, the team worked closely together with business partners and drone pilots in order to understand what drone hardware the company should initially target. In addition to the hardware selection, the business partners and drone pilots were also consulted when the team was planning their initial requirements.

The software development process the team uses to manage their work was described to be a Kanban (Poppendieck and Poppendieck, 2010) process. The Kanban board, as well as the company roadmap, is maintained virtually using the Trello[12] collaboration tool. The interviewee mentioned GitHub[13] actions being used to implement a CI/CD pipeline, which also implied that the source code was hosted on GitHub. The team used code reviews but there were no automated tests at the time of the interview. The build process was described to be mostly automated, having a manual triggering for publishing new releases. The development team was the one to choose when a release was to be published. The product was deployed as a side-loadable application, and releasing a new version of the software updated a link and a version number on the product website. Side-loading an application means that the application package is not installed through an official store such as Google Play[14]. The team was planning to eventually publish the application through official channels. At the time of the interview, the team was more focused on building the product itself and considered side-loading to be a good enough approach for their early access program.

The team had been collecting data from multiple sources from the very beginning. Early on, the company had been working closely with its potential customers and industry experts. Later on, they had been working with three expert drone pilots to collect feedback about their development versions. The team completed multiple iterations of building a new version, collecting data from expert users using the product, and planning the next version based on the feedback. At the time of the interview, the team had just released their first public version of their product to EAP participants and was expecting to receive feedback from the program participants. There were no predefined metrics for what kind of feedback the company would expect to receive from the users. Instead, they were waiting to see what the users think of the product. One potential problem the interviewee mentioned was that there are very few AR products on the market, which makes comparing the product to other similar products very difficult. Furthermore, they expected their product to be the first AR product their users would experience, which could make it difficult to know if user feedback was about the AR experience itself or the actual product experience. Before the EAP launch, the main channel of communicating with users was discussing with users. The email was the channel used for EAP feedback collection and the interviewee expected the discussions to remain as the main channel for user collection for quite some time. They did recognize that a better process would be needed in the future as the current scheme would not be able to scale for more than a handful of users. The company was not automatically collecting user event data from their

[12]Trello. URL: https://trello.com (visited on 12/28/2021)
[13]GitHub. URL: https://github.com (visited on 12/28/2021)
[14]Google Play. URL: https://play.google.com/store (visited 12/28/2021)

applications as they expected their customers would not want to share all the details of their drone flying operations. Instead, the application had an option to share data with the developers by pushing a button. The interviewee explained that the feature was primarily meant for debugging errors and allowed the team to see what data the user had seen during a flight.

4.1.4 Company D

Fourth company was a company building a commerce platform for selling access. The company provided a suite of software solutions for rental companies. On the commercial side, the company provided a white label online shop for rental businesses. Additionally, the company provided software for inventory management, order management, CRM, and payment processing. The suite of products complemented each other and customers were able to select which parts of the company offering they want to take into use. Their first products had been launched 30 months prior to the interview. They reported having customers from all around the globe.

The interviewee was one of the company founders working as the CEO and a member of the company board. They had been previously actively working on operations level during the earlier stages of the company. However, at the time of the interview, their work at the company had become more about leadership and management than hands-on operations. The company had 3 product development teams with a designer and one or more software engineers each.

The interviewee explained that the company maintained a roadmap with over 600 items. They recognized that they would be able to complete maybe 50 items in a year and were therefore heavily prioritizing their work. The company was using a framework called DIBB (Kniberg, 2016) to prioritize their development work. The name is short from data, insight, belief, and bet. These four words describe how data is first converted into insight, then the insights turned into beliefs, and finally the team can bet on their constructed beliefs. The bets of the model were formalised with objective key result objects (OKR) that described high level company goals in measurable form. Importantly, the OKR items were used to ensure the alignment of planned product development work, company strategy, and meeting the investor set goals. The OKR items were planned quarterly and development teams were tasked with implementing them. The teams were free to choose how they would reach the goals defined in the OKR items, as long as they able to meet the goals by the set deadline.

The company did not have a shared software development practice. Instead, each team was able to choose the tools and processes they wanted to use. While there were no enforced practices, the interviewee explained that the teams are certainly using agile and lean methods in some form. Generally, there were automated CI/CD pipelines with automated unit and integration tests. The development teams were also doing code reviews. Releases were highly automated and the interviewee stated that creating a release is mostly limited by finding someone to review the changes. The target release interval was one to two weeks. Anything longer was seen as a failure in the planning of work. The interviewee explained that the the OKRs were

purposefully scoped to be small so that the company would be able to collect user data from the changes without needing to wait for too long. Once released, all services were under continuous run-time monitoring and automated notifications were sent if the services were not running nominally.

The company was collecting as much data as they could. They had instrumented their software for event level data collection. There was an always on chat embedded into their software allowing real-time communication with the users. They were observing every change in their databases, and even combining their accounting data with everything else they collect. There was a dedicated data ops team whose sole purpose appeared to be to analyze, refine, and maintain the data the company was collecting. The company had built a Google Data Studio page for accessing the data, and new data views were built whenever someone from the organization came up with a new piece of information they would want to see. As mentioned before, the data was used extensively in the planning of new OKR items. The data was used for detecting problems and rooms of improvement in the current product. Interestingly, the interviewee explained that the team also tried to find a hidden signal from the noise in order to predict where the market is going to develop in the future. They claimed that some 10% of their new tasks were a result of predicting future needs based on their data. Also the the product development teams were reported to frequently use company data when implementing the OKRs. The teams would use the data for understanding customer needs during the design phase of new features.

Experimentation was used frequently in the company. Building medium-fidelity prototypes was a regular activity among the product development team. The practice was to build a great many prototypes and test them out internally. Occasionally, the prototypes were tested with customers or regular people outside the company. The prototypes were typically built before writing a single line of code. The company also mentioned doing AB-testing on their website for apparently marketing purposes. However, the practice was not done with their actual products due to "not having resources to do it". When asked about any major product development failures, the interviewee claimed that there weren't really any. There were still plenty of really basic features that would obviously improve the user experience, and the product, without any need for experimentation.

4.1.5 Company E

Company E provided a subscription-based service where the company employees visit the homes of the customer or e.g. a relative of the customer. During the visits, the employees provided services that varied from helping out with daily tasks to socializing and going for a walk together. Even though the service was provided by human employees, the company had developed internal software for managing employees, managing subscriptions, sending invoices, communicating with customers, and building reports for customers. The company was also planning to build customer-facing mobile applications by the end of the year. The interviewee estimated that the service had been available for customers for 24 to 30 months at the time of the interview.

Company E interviewee was the lead software engineer who had worked for the company for 5 months at the time of the interview. They were responsible for leading the software development as well as any recruitment and operative work that concerned the tech team. They had a few years of experience working as a software engineer consultant before joining Company E. They noted that most of the team of Company E, outside the tech team, has very limited understanding of how software is developed.

The interviewee stated that the company had a roadmap "pretty much until the end of the year". The major items on the roadmap were customer-facing mobile applications, an internal mobile application, and automated invoicing. There was no discussion about potential other, non-technical, items on the overall company roadmap. However, the interviewee explained that the length and contents of the roadmap were based on the goals set by the investors. The work of the software development team was planned during weekly meetings where the founders and tech team members discuss the current status. During these meetings, the founders could also prompt the tech team to start working on a new item from the roadmap. The interviewee explained, that the roadmap items are often described on a very high level, usually lacking a definition of done, and the first order of business would be for the tech team to divide the new epic into manageable pieces. There was no dedicated product owner and the development team appeared to be fully independent to choose how to implement and prioritize the work after the task had been moved from the roadmap into the implementation phase. An additional source of software development work items was feedback from company staff using the applications. The interviewee explained that they maintain a separate space where these work items enter without much filtering. The items were then later to be reviewed during dailies and promoted to the backlog at the discretion of the tech team.

The software development team consisted of two developers and a part-time designer. The interviewee explained that they were using agile methods, and the model they were using at the time was inspired by kanban (Poppendieck and Poppendieck, 2010). They admitted that the process was still taking shape, and they were trying to add a proper backlog refinement activity into their way of working in near future. At the time of the interview, ticket refinement was done during the tech team's daily meetings as needed. The interviewee hoped that establishing a separate refinement meeting would allow founders and other employees outside the tech team to provide more input when maintaining the tech team's backlog. There was no discussion about the tools used to maintain the kanban board, roadmap, or individual tickets. The source code was pushed to GitHub code repositories and new work was developed in separate branches. GitHub pull requests were used for code reviews and merging to the main branch triggered CI/CD pipeline to automatically deploy into a test environment. The CI/CD pipeline lacked any automated testing, and the team tested their releases manually before allowing the automation to deploy the changes to production. The interviewee admitted that regression bugs were fairly common and manual testing had proved to be challenging as the software grew more complex. They were looking into building automated test suites.

The new release process was initiated by the tech team whenever either one of

the two developers felt like a new release would be in order. The decision for pushing a new release had to be unanimous. It was left unclear, whether the decision was made solely by the discretion of the tech team or if there were other stakeholders, e.g. the founders, involved. There was no target for a release cadence and the interviewee described that "...it varies a lot. Some weeks we may have no production releases. In another week, we can have ten production releases. It varies a lot.".

The primary channel of customer data collection was the unsolicited feedback received through e-mail. The interviewee explained that the customers would reply to visit reports written by the company field workers after each visit. The company also had a feedback form available. Customer service agents collated the feedback and relevant stakeholders were notified. The feedback data was not generally available internally and the tech team was kept in the dark. Even though good feedback was occasionally shared on a company Slack channel, the interviewee hoped for better access to data while recognizing that some members of the company may not have been accustomed to a transparent work environment. They speculated that further discussions would be needed before everybody would see the benefits of more open access to all data. The primary channels for collecting feedback about the internal applications were observing colleagues using the application. The interviewee felt that any problems with the applications were quickly reported either face to face or through per-application Slack channels. The company had done a survey for their field workers in order to understand how they feel about the application they use to check their visiting schedules and write post-visit reports. There was no instrumentation in any of the internal applications and it appears that the company collected only qualitative data.

4.2 Findings

Results for each individual case company are presented in the next subsections. Research questions 1 and 2 are answered for each company. However, research question 3 requires synthesis of data from all five cases and the proposed answer is presented in the last subsection.

4.2.1 Company A

The company operated on a basic Scrum model planning their work in two week sprints and finishing each sprint with a demo session. Figure 6 describes the process. The upper part of the graph contains the flow for designing of new features and introducing the work items into backlog.

The software development team used GitHub for source code management and development work was done in separate branches. Feature branches were merged into update branches after being reviewed. Each update branch contained a set of changes that would be released at the same time. They reported that up to two update branches were typically built during a two week sprint, which would suggest an average release cadence of one week. Update branches would be merged to master branch once there was enough changes for a new release.

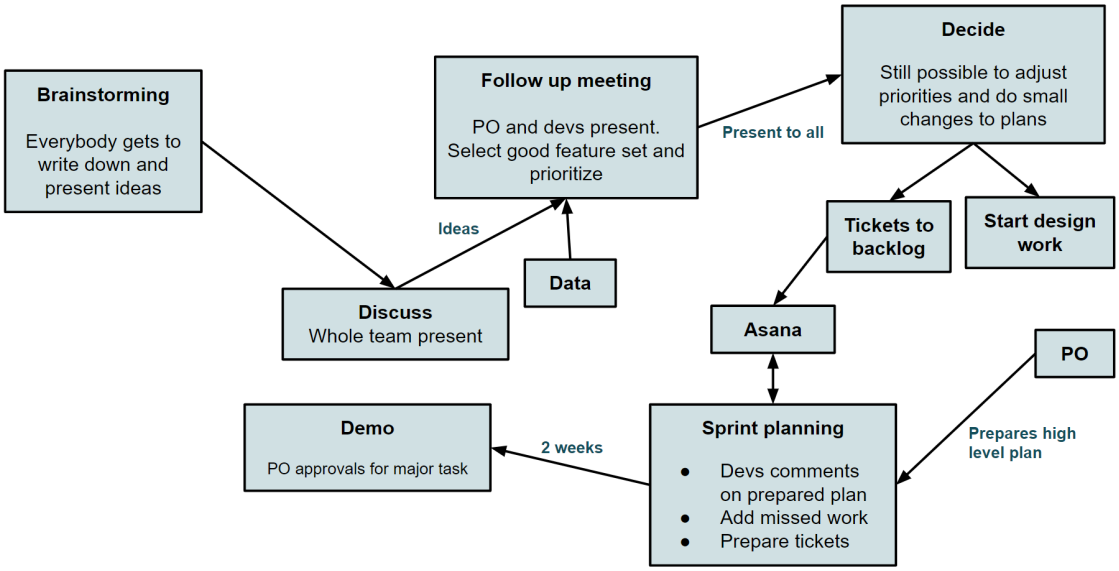

Figure 6: The product development process of company A.

The company had set up a CI/CD pipeline to automatically create a new releases after master branch was updated. Backend releases had previously been pushed automatically to a staging environment pending for manual approval before being pushed to production. After moving to a new cloud environment, changes to the master branch were automatically pushed into production. The mobile applications were built manually using Xcode IDE and Android build tools. The application packages were normally first published as internal beta releases and made public after the team had verified that the new versions work as intended.

The company had instrumented their application for event level data collection, and also collected qualitative data through multiple channels including feedback form, application store reviews, social media, and a feedback form. The data was analyzed by the product owner and insights were shared with the rest of the team during weekly meetings. There were two identfied cases, where user data had been used to improve the product. First, a poorly performing feature was removed from the application after the initial version of the feature had failed to gain traction. Secondly, the main menu appeared to go unnoticed by many users, which prompted further investigation. The team conducted user testing, which verified their hypothesis that the menu icon was not noticed by the users. An improved version of the UI was developed and another set of user testing was used to validate the new design, with great results. Similarly, user testing was being used to improve the onboarding flow of new users. At the time of the interview, the second version of the flow was being tested and the team was waiting for the data to come in.

The team appeared to be using some form of experimentation as a part of their product development work. There were examples of the team analyzing the data and forming hypotheses for new feature development as well as improving poorly performing parts of the application. However, the interview data suggested that the experimentation was used very selectively, and most of the time the team was not

trying out multiple implementations.

> And then you need to decide, if your runway is for example six months, then you need to choose what we want to do still, or try during these six months. And because we have such a small team it is difficult to do or try two completely different things during this period.
> —PO, Company A, on prioritization

The team appeared to be still searching for a product-market fit, and implementing new features from the roadmap was prioritized over experimentation or building automated test suites. It was apparent that limited funding and resources forced the team to only concentrate on the tasks that appeared to add most value from day 1.

4.2.2 Company B

The company did not have any established and well defined product development processes. The process as interpreted from the interview data is described in Figure 7.

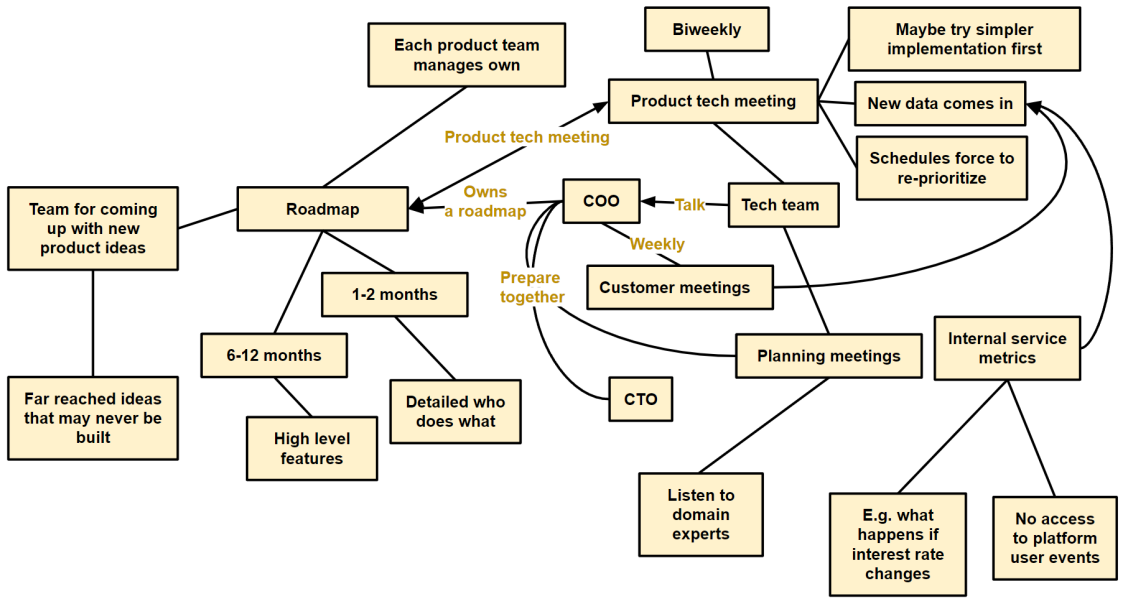

Figure 7: The product development process of company B.

The central role of the COO is not due to the fact that he is the COO but due to his role as being responsible for leading two products. The main model of operating appeared to be weekly or biweekly internal meetings where different configurations of business and tech persons discussed based on new data and adjusted their plans accordingly.

> We do not have that product management and product development process, so for example some sprint agile thing or something. We do not have that at the moment, so we need to develop something like that.
> —COO, Company B

The interviewee argued that the small team was better off without establishing rigid processes, which was seen as introducing unnecessary overhead into the work of the team.

> There is no set in stone formal process that 'this is the roadmap and this is how we execute it' because we are such as small team after all, so we do not need any heavy coordination processes.
> —COO, Company B

While the tools and processes outside software development functions were less organized, the software development team was using Kanban model and building their software incrementally. They used GitHub for source code management and development was done in separate branches. Branches were merged into master branch after passing an automated test suite and code review. CI/CD pipeline automatically deployed new changes into an integration environment and the team was pushing new releases into production on a daily basis.

The company had very limited access to user data as their service was used through a third party UI. The business partners owning the UI were not willing to share detailed user data, and the company had to rely on the limited data they had available from their own systems. Even though the business partners were unwilling to share data, the company worked closely with current and potential customers for requirement elicitation. Data from weekly customer meetings was used extensively from early on and the products were engineered to meet partner defined functional and non-functional requirements. The company also planned to use competitor benchmarking to come up with new product ideas. Overall, the company appeared to base their product development decisions on the data received through their business partners and less on the scarcely available user data.

Even though the company would likely be technologically capable of running even continuous experimentation, the company's B2B business model appears to make any user data involving experimentation very difficult. Firstly, the company did not control the UI used to access their service and was therefore unable to do any UI related experimentation. Secondly, their business partners were unwilling to share detailed user data. Finally, the only parameter the company claimed to be adjustable was the pricing of their products. Adjusting the pricing was also considered to be difficult as any changes would need to be coordinated with, and approved by, their business partners. These obstacles of experimentation for companies operating in the B2B domain are well recognized in earlier studies (Auer et al., 2021).

> We do not have resources at the moment to do anything repeatedly or as a process...this interest rate is the only parameter we could control.
> —COO, Company B, on A/B-testing

The limited access to user data and lack of modifiable parameters in their current products appeared to be major blockers for systematic experimentation on Company B.

4.2.3 Company C

Company C was in a transition phase where they had just launched their application to first customers through an early access program (EAP). The product development process used before moving to the EAP model is described in the Figure 8.

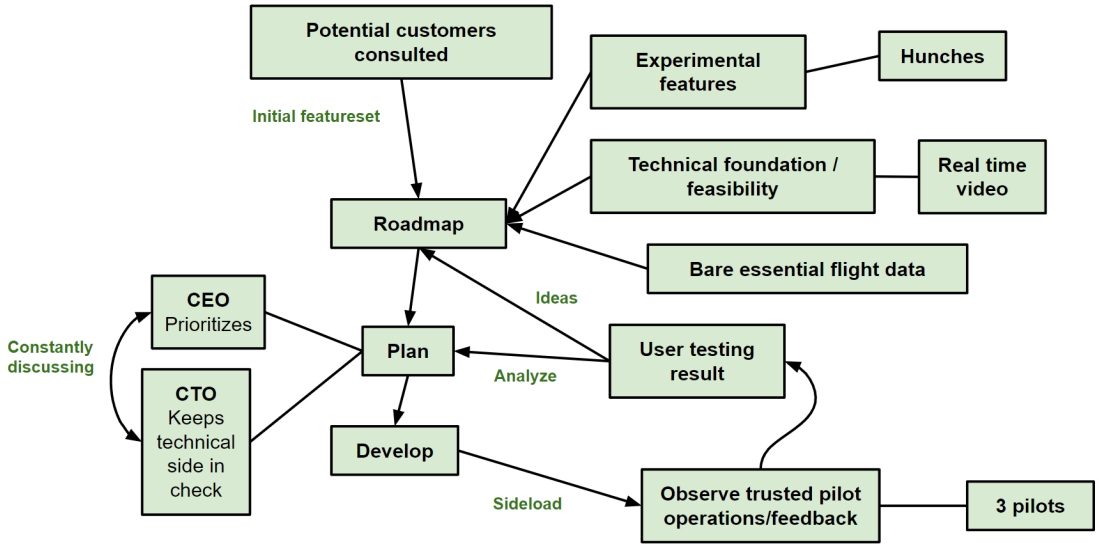

Figure 8: The product development process of company C.

When developing this MVP version of their application, they had worked closely with potential customers and industry experts in order to understand what kind of requirements their future customers would have for the product. After establishing the initial set of requirements the company executed numerous build-measure-learn (Ries, 2011) iterations where each new version of the product was subjected to an user test by an expert user. During each test, the team collected user data by observing the user as well as interviewing the user for feedback. The user data was then converted into software development tasks. In the absence of a product owner role, the product development prioritization was done by the CEO who, as a founder, had invented the original product idea. The CTO's role in the planning was then to make sure that all the technical concerns were taken into account during the planning. Overall, there was very little structure in the planning process as described by the CTO.

> Practically we discuss these matters all the time, so we do not have a formal process other than that. We need to get into the timeline the tasks we are going to do next.
> —CTO, Company C

> It has been this like, very tight develop and test type of work where we develop something and then go fly.
> —CTO, Company C, on the development process

The software was developed incrementally using the Kanban (Poppendieck and Poppendieck, 2010) process. They used Github[15] for source code management and development was done in separate branches. Branches were merged into master branch after a code review and CI/CD pipeline automatically generated release candidate builds from new changes. Each release candidate was tested manually and there was no automated testing at the time of the interview. Production releases required manual confirmation before an automated process pushed new application packages into a distribution site. Notably, the application had to be manually installed instead of being available through an official application store such as Google Play. This was deemed a good enough solution for now, and reflects the overall team philosophy of deferring setup of processes and infrastructure until there was a clear need for change.

The company was mainly focused on collecting qualitative data from the expert users and EAP members. The company had used the data from expert users for validating their initial set of requirements for the first public version of their application. Moving to EAP, the interviewee explained that there were no plans for how the data would be used at the time. They did not have any structured feedback form prepared for the EAP customers to fill in and all feedback was collected by email. They were hoping that users would provide general feedback about what is good and what is bad in the first version as well as comment on the pricing of the product. The company was worried that, being the first of its kind AR application, the users would not be able to compare the product with any other products they would have previously used.

> There is a lot of people who have never worn a Hololens...when they try our application, just seeing holograms flying in your field of vision and doing something useful, that alone can provide a wow effect.
> —CTO, Company C, on users experiencing their application

The company had included instrumentation into their application and stored detailed logs of application use for debugging purposes. However, due to user privacy expectations as well as to ensure offline capabilities, the data was not automatically sent to the company. At the time of the interview, the company was expecting that most of the feedback would be received through email and discussing with the users. This was seen as good enough for now and they recognized that some other way of handling user feedback would need to be implemented once their customer base grows.

The company was doing some form of repetitive experimentation when building their EAP version. However, there was no sign of predefined metrics or evaluation criteria which would indicate a systematic approach for experimentation. For example, when discussing the success criteria for an UI improvement, the interviewee explained that "so far it is like, well this is a user interface so, if it looks good, that's the criteria". The company had thought about doing AB-testing sometime in the future and the interviewee explained that they had recently implemented feature flags, mainly for

[15]GitHub. URL: https://github.com (visited on 12/28/2021)

licensing purposes, that would provide them technical capability to offer modified versions to different user groups.

> As a theoretical model because right now we do not have too many customers, so we are not quite there yet, but then at some later time...we could feature flag.
> —CTO, Company C, on AB-testing

The company was effectively validating their MVP feature-set with their EAP and as such felt that implementing the basic features adequately, which were really the bare minimum requirements for flying a drone, was the most valuable work to do at the moment. These basic features appeared to be understood well enough to not require an experimental approach. The company also felt that they would need to know their users better and improve their distribution method before they would be ready for AB-testing.

> Well, maybe we should first understand our users better overall, and then the application [distribution] model as it is now is maybe weak because users have to install it themselves.
> —CTO, Company C, on AB-testing

4.2.4 Company D

Company D was using a fairly well defined process for planning their product development. The process model used by the company D is depicted in figure 9.

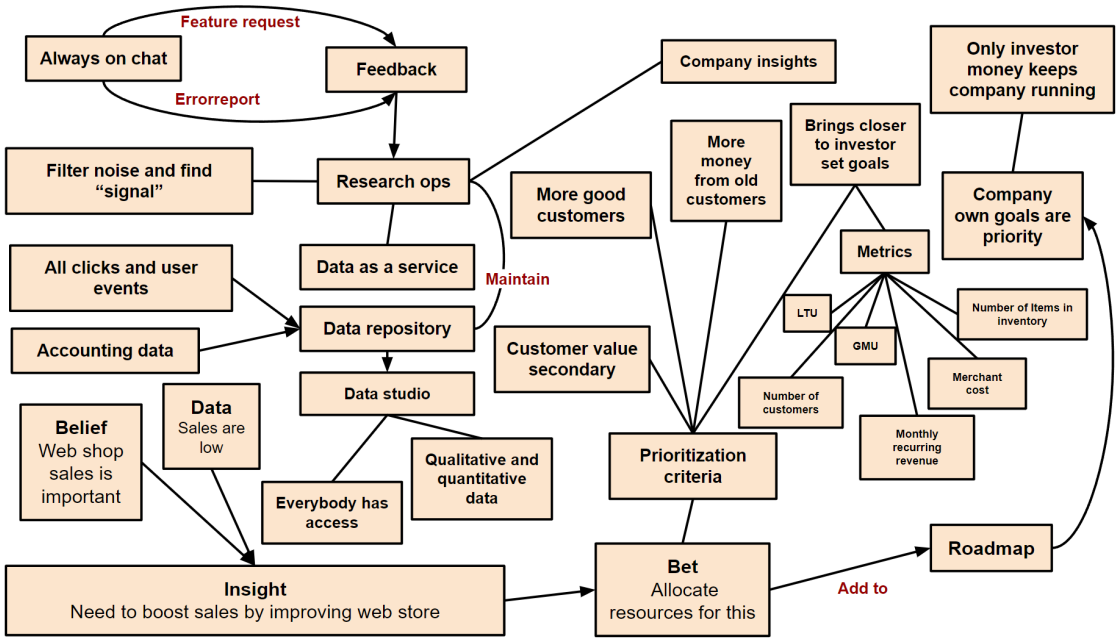

Figure 9: The product development process of company D.

The leadership would define a set of objective key results (OKR) for the next quarter for the product development teams to implement. Each OKR would contain

a high level goal and the metrics that would be used to evaluate whether or not the goal was reached at the end of the quarter. An OKR item could also have a stretch goal that would be nice to reach, but should not be prioritized over meeting the deadline. When planning the OKRs, the company made sure the OKRs align with the company strategy and would bring the company closer to meeting investor defined goals. Specifically, meeting investor defined goals was seen as more important than fulfilling customer needs. Sometimes, new data could render a planned OKR obsolete, and the company could discard an OKR before it was fully implemented. Another identified tool used by the company was the DIBB argumentation framework (Kniberg, 2016) the team was using to come up with the OKRs. The interviewee explained that they had come up with the approach naturally and had only later learned that what they were doing was essentially implementing the DIBB framework.

The OKR items were implemented by product development teams who were free to choose the best way to reach the defined goals. The product development teams were free to choose their tools and processes. The CEO explained that "nobody says they use Scrum, nobody says they use Kanban, but the related agile practices and the related mantras, are certainly followed". Generally, there would be a quarterly kickoff where schedules were planned as well as weekly and daily meetings, but the teams were able to bring good practices from their previous jobs into the current one. An example of such practice was an employee who had worked previously in a software consultancy and had helped setting up retrospective meetings after joining the company.

The development teams regularly built prototypes of designed features before a single line of code is written. Quantity of prototypes was preferred over quality and one feature had 40-50 different prototype flows built during the development. The prototyping was mostly used internally, but occasionally the prototypes were subjected to end users or customer representatives. An expert board reviewed the experimental data from prototyping and selected the best implementation based on qualitative feedback.

While teams were able to choose their practices freely, there were common software development tools and practices used. Source code was managed in GitHub and code reviews were used for new changes. There was an automated CI/CD pipeline with automated unit and integration tests. Live systems were monitored automatically and notifications would be sent if any problems were detected.

There were numerous user data collection channels employed by the company. Quantitative data was collected on the event level on different parts of the system: database, backend services, and frontend. Qualitative data was collected through an always-on chat embedded in the service, which allowed customers to easily communicate any feature requests or problems to the company staff. The product related data was then combined with e.g. company financial data for a more holistic view. Research ops was a dedicated business unit tasked to analyze and refine all available data. Their responsibility was to make sure that the data is made available to all members of the organization should they need to access it. The access interface to all data was a Google Data Studio page maintained by the research ops.

Data was used extensively in all stages of planning and product development.

The DIBB framework used for planning future work built insights and beliefs based on the data and allowed the leadership to bet on these beliefs in the form of OKR items. The product development teams would access the data, and generate more data through prototyping, to choose the best way to implement the goals defined in the OKR. Finally, the successful implementation of the OKR item was determined by comparing the OKR defined metrics to the data from the period after the changes were deployed to production.

It would appear that the use of data in company D was continuous as the company was not only reacting to bad data, but actively trying to learn from the data and even predicting future needs based on historical data. Not only the use of data in decision making, but also the collection and refinement of data was also continuously done by the research ops team. The continuous nature of data use can be seen in Diagram D.

> 10 percent of the features are ones where we had combined all this feedback and looked at it, and predicted what kind of things there would be coming. So we can say that we actually need to build this kind of feature because now we see they [customers] are going towards this kind of world
> —CEO, Company C, on use of data

The company was also doing experimentation by building prototype versions of new features. However, the data suggests that experimentation was primarily done within the product development teams. The data from experiments may have been used in planning of the OKRs, but the nature of OKRs appeared to be more about setting goals and allocating resources, rather than representing an experiment. The OKR embodied only the leadership's will to bet on some belief formed after gaining insight by analyzing available data. It could be argued that the insight and belief are a form of a hypothesis, and the goals are the defined metrics. However, it was still up to the product development team to decide if they want to take an experimental approach or not.

4.2.5 Company E

Company E product was a physical service company, and software was developed primarily to support internal processes. The company maintained a roadmap spanning until the end of the year, which was also the current funding period. The product development process used for developing their internal software is depicted in figure 10. The figure illustrates how little the software development team was involved with the rest of the company.

On a high level, the software development work was planned in weekly meetings where founders and development team discussed current status and founders had an opportunity to move epics from company roadmap into development team backlog. These epics only contained high level descriptions of company goals such as building a mobile application for customers.

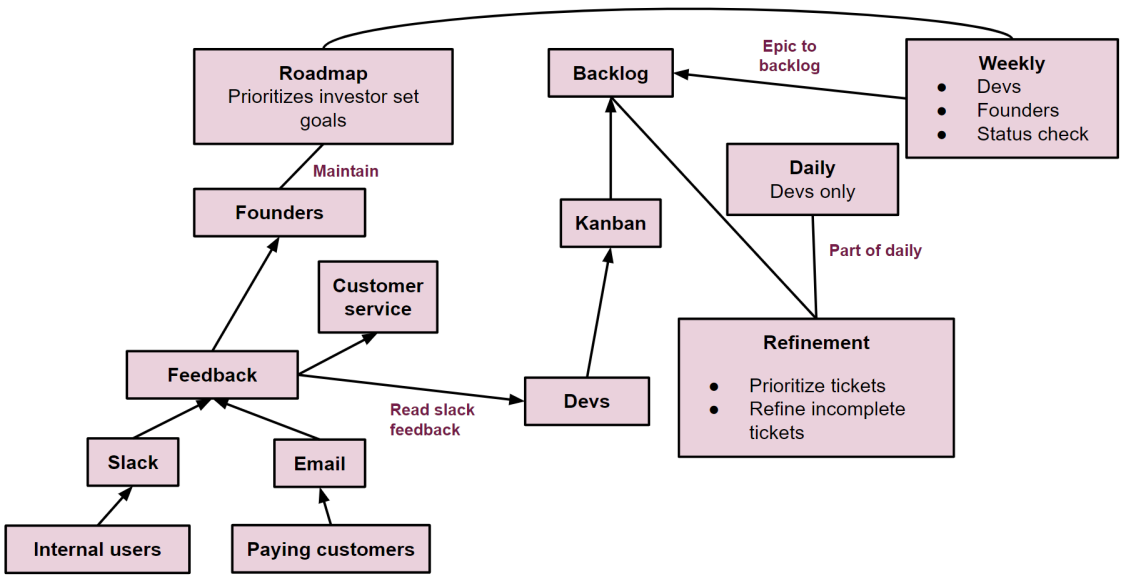

Figure 10: The product development process of company E.

> We do not have any definition of done at the moment. Often the epic is described at a really high level. It can be usually something big such as the client application.
> —Lead software engineer, Company E, on roadmap items

It was the task of the development team to turn the high level epic into manageable work items that would eventually end up on the team's backlog. The interviewee described the working process of the development team as Kanban (Poppendieck and Poppendieck, 2010) inspired. It appeared that daily meetings were the only meetings the team used to coordinate their work. The interviewee felt that there should be a separate backlog refinement meeting that would be joined by the founders and other employees who were not participating in the daily meetings. There was little understanding of how software is developed outside the development team, which could explain the apparent disconnection of the development team and the rest of the organization.

In addition to the Kanban process, the team used GitHub for code versioning and code reviewing. There was an automated CI/CD pipeline that deployed new changes automatically into a test environment. There were no automated testing and all changes were tested manually in the test environment. Manual confirmation was required to trigger the production release pipeline. The team had encountered frequent regressions bugs and was hoping to build an automated test suite at some point.

> If we had more resources, I would probably want to have better automated testing and monitoring, alarms and such. Certain infrastructure level work would need to be done if we had more resources.
> —Lead developer, Company E, on QA

The internal software had no instrumentation for qualitative data collection as the team did not appear to be interested in collecting quantitative data from internal software. The two main methods of collecting qualitative user data for the internal software was observing office staff and reading feedback from internal application specific chat channels. This internal feedback loop was tied directly back to the development team's Kanban board where any new input was entered into a processing column. The team would look at the processing column items during daily meetings and decide by themselves what to do with the items. A survey was used to collect feedback from company field workers who used an internal service to check their work schedule and write post-visit reports. It was not clear how this data was then analyzed and what kind of actions were taken based on it. Diagram E depicts the high level flow of data within the organization.

> We have sent our field workers these quick surveys just to see what is the general opinion about this app and such. We did not have any specific metrics like "rate this from 0 to 10" or anything.
> —Lead developer, Company E, on user surveys

Customer feedback was collected through a feedback form and email. This feedback was handled by customer service agents and was generally not available for everybody. The interview data contained little information about how customer feedback is used in the development of the company's physical service product. Nevertheless, the company had decided to build customer facing mobile applications based on the received customer feedback. Overall, the company did not appear to be systematically using data at least in their internal software product development. The data was perhaps seen only as a way to find out if something is not well, instead of building a better understanding of the users' needs.

> ...ideal situation would also be that there is no feedback at all as quite often not receiving any feedback about anything indicates that things are probably quite ok as there is nothing to complain about.
> —Lead developer, Company E, on negative feedback

When it comes to experimentation, the team had envisioned doing AB-testing once they have released customer facing applications. However, based on the interview data, the organization at its current state would not have the capabilities to effectively use experimentation as a part of their product development. Notably, the disconnection of the development team and the rest of the organization makes it difficult to see how the organization as a whole could come together to effectively run experiments.

4.3 Combining the pieces

Combining data from all case companies shows that there are some common tools and practices used by most or all case companies. Four out of five companies are operating in a less organized way and one company appears to be somewhat more advanced on multiple levels. Table 2 contains the product development and data usage practices the case companies were found to use based on the interview data.

Table 2: Summary of identified practices in all case companies.

	A	B	C	D	E
Process	Scrum	Kanban	Kanban	Agile or lean	Kanban like
Continuous Integration	Yes	Yes	Yes	Yes	Yes
Continuous Delivery	Yes	Yes	Yes	Yes	Yes
Continuous Deployment	Backend yes, Application no	Yes	Yes	Yes	Yes
Continuous Monitoring	Unknown	Unknown	Unknown	Yes	No
Code Versioning	Yes	Yes	Yes	Yes	Yes
Automatic testing	No	Yes	No	Yes	No
A/B-testing	Some	No	No	Some	No
Experimentation	Yes	Limited	Yes	Yes	No
Prototyping	Yes	Unknown	Yes	Yes	No
Roadmap	Yes	Yes	Yes	Yes	Yes
Event level data	Yes	Yes	Opt-in	Yes	No
Data channels	Email, In-app instrumentation, Application store reviews, Social media, Feedback form	Weekly meetings with customers, Meetings with potential customers, Competitor benchmarking, API events	Email, Frequent user testing with expert users, meetings with industry experts	In-app chat, In-app instrumentation, Database instrumentation, Financial data	Observe users, Internal chat channel, Email, Feedback form

Continuation of Table 2

	A	B	C	D	E
Data analysis	Weekly by PO	Weekly by COO	CTO and CEO together	DIBB, OKR planning, Dedicated data operations team	User data by customer service agents and founders, Internal service feedback by the development team
Data based decision making	Yes	Yes	Yes	Yes	Yes
Access to user data	Yes	No	Yes	Yes	Yes

4.3.1 Product development practices

All case companies were building their software incrementally and using lean or agile practices. Continuous integration, delivery, and deployment were used by all case companies. Continuous deployment was done automatically to some test environment, and a manual confirmation was required to deploy the changes to production. Company A application was built manually and company C had the application release candidates built automatically. Based on their respective technical infrastructure, all of the case companies would be on the fourth step, Continuous deployment, of the Stairway to Heaven by Olsson, Alahyari, and Bosch (2012). However, the Stairway to Heaven also lists other factors such as the involvement of product management and customers in the product development cycle. Only the companies B and D appear to satisfy all of the requirements for being on the fourth step of the model. Companies A and C fall short with their non-continuous deployment of the application and Company E has an R & D department operating mostly separately from the product management. Furthermore, only company D had well defined processes that displayed systematic use of data in all stages of their product development process from planning, to execution and even data refinement.

The product management and work planning practices on reported by companies A, B, C, and E were quite light, consisting mostly of weekly or biweekly meetings between stakeholders. Small team size appears to allow handling matters with ad-hoc meetings and companies seemed reluctant to establish heavy coordination processes.

> Because we are such a small team, we do not need any heavy coordination

processes.
—COO, Company B

Only the Company D, with around 20 employees, demonstrated thought out structure in their ways of working. They were doing quarterly planning of higher level company goals in the form of OKRs. Importantly the OKRs were also tied to company strategy and investor defined goals. On a lower level, their implementation teams were allowed to work and implement the OKRs in any way the team saw fit as long as the deadlines and goals set by the management were met.

4.3.2 Experimentation

Experimentation was used in some form in four of the five companies in different forms. There was user testing with different UI implementations and a MVP feature implementation on company A, a price change experiment going on in company B, repetitive prototyping and expert user testing in a continuous cycle on company C, extensive prototyping and user testing in company D. Company C appeared to be using prototyping in a systematic way. However, their data collection and data analysis practices did not appear to be systematic. Based on the interview, they were mostly giving the product to their expert test user without planning ahead what kind of data and metrics they were going to collect. Company D appeared to have very advanced data collection, storage, and refinement capabilities. They also used the data systematically, and would most likely be capable of continuous experimentation.

4.3.3 Previous experience matters

The combined data contains three recurring code categories that could explain why the case companies are using the tools and practices they use, appear to be neglecting some known best practices such as automated testing, and choose to operate without establishing well defined product development processes.

The first category is using previous experience as the basis for selecting the tools and practices used in the new company. It appeared that companies had set up the tools and practices based on the previous experience of the founders or employees. There were single pieces of experimentation practices such as user testing, prototyping, and even occasional AB-testing used in the companies, but that appears to be a result of previous good experiences from these techniques instead of some active effort of attempting to apply some industry wide best practices.

> This is still a small company so practically somebody was there first and set something up.
> —COO, Company B

> I have holistically tried to think about it, based on my own experience, how these things should be done. And tried to implement them in prioritized order.
> —COO, Company B, on establishing processes

> Q: How do you find the process you want to use, or how does it take shape?
> A: Well, it is kind of simple as we have three senior developers...So practically three senior developers and they get to work it out themselves.
> —COO, Company B

> New ways of doing things, yes, they come often through recruitment and people's hobbies and that open culture.
> ——CEO, Company D, on introducing new tools and practices

4.3.4 Limited resources

The second recurring category is limited resources. Allocating resources in the best possible way to execute the company roadmap in the most efficient way was central in the planning of work in all of the case companies. Company B was the only one not talking directly about the funding of the company, but was nevertheless forced to prioritize tasks and adjust scopes in order to meet deadlines. The common theme was that the company roadmap contained far more work than the company would ever be able to implement within the runway they had available and the resources they had available for development.

> We have a roadmap, probably describes best that we have 600 feature request of which we know we can implement during the next six months about 50.
> —CEO, Company D

> Q: Is this lack of developer resources a bottleneck that limits you?
> A: Yes it is, even our CEO always tries to code when possible...because there is always more work to be done than hands available.
> —CTO, Company C

> Almost paradoxically, techniques commonly used to free developer time, such as automated testing, appear to be known to the interviewees and are not taken into use.

> And we have more important tasks...those tests are good to be there later, if we have otherwise functioning product with actual users.
> —PO, Company A, about automated testing

> Unity is not really nice for automated testing. UI iterations are done fast, so the amount of work that would be needed to build tests for potentially discarded features has felt too big in comparison to the benefits.
> —CTO, Company C

> If we had more resources, I would probably want to have better automated testing and monitoring, alarms and such. Certain infrastructure level work would need to be done if we had more resources.
> —Lead developer, Company E, on QA

The same appeal to limited resources was also used to explain why experimenting with multiple designs is often not seen feasible.

> And then you need to decide, if your runway is for example six months, then you need to choose what we want to do still, or try during these six months. And because we have such a small team it is difficult to do or try two completely different things during this period.
> —PO, Company A, on prioritization

> ...based on the fact that we just don't have the resources to do it at the moment.
> —CEO, Company D, on AB-testing

This chronic lack of resources appeared to force the companies to prioritize exactly the kind of work that would bring them closer to the next funding round. This could even mean saying no to work that would otherwise produce real value to the customers. Reasoning for this was given by the company D CEO in the following quote.

> Often our product roadmap serves at the moment not only our customers but the goals of the company, and the stage we are as a company is, that we are very much being funded...and then a trivialization is formed that you have some runway, say 12 months money for this group. And investors and the corporate governance have often quite explicit milestones to reach in order to get more funding ... so it does not matter how good of a job we do...we are not going to get the money needed to fund this development from our customers.
> —CEO, Company D, on why meeting investor needs is prioritized

Based on these observations, it can be argued that these companies are very careful to not waste any of the precious funding and time available to meet the investor defined milestones for the next round of funding.

4.3.5 Introducing change

The companies appear to not actively seek change. In fact, they appear to actively avoid setting up complicated processes until something is seen as really needed.

> Because we are such a small team, we do not need any heavy coordination processes.
> —COO, Company B

> As crude as it can be to get it shipped. Our plan generally, our model has been to take small steps and improve as needed.
> —CTO, Company C, on setting up distribution channel

They may even experience some problems with the way they are operating, but that alone is not enough to start looking for better ways. Even if the team would know a better way of doing their work, the effort needed to set up a new system and the opportunity cost of not being able to work on other, more important, tasks can stop any change from taking place.

> And we have more important tasks...those tests are good to be there later, if we have otherwise functioning product with actual users.
> —PO, Company A, about automated testing

Even if companies do adopt some new practices, they appear to be ready to return to old ways unless the expected benefits are actually realized.

> Some practices we have started have also been killed after noticing that...they only consume calendar time and gives me nothing.
> ——CEO, Company D

Change can be seen as waste of resources unless there is perceived tangible benefit in sight. One that materializes within the available runway of the company as described by the Company D CEO in the following quote.

> Someone checks that we can get an impact from this within our runway and accepts. If it is something that we can see in two years, then we most likely say that there is some other company where these things can be done, but our mission in this company right now is something else.
> —CEO, Company D, on adopting new practices

4.3.6 The appeal of Continuous Experimentation

Combining the findings from these three categories allows us to understand some of the reasons why continuous experimentation is not used in the companies. It would appear that for these case companies to adopt a tool or practice, three conditions need to be satisfied. First, following from the observation that previous experience influences the adoption of a tool or practice in the company. Someone in the company must have prior knowledge of the tool. Secondly, companies running short on resources, taking the new tool or practice into use must not require a significant amount of resources, i.e. time, money or man hours. Thirdly, for a tool to be adopted, the company must perceive that the tool or practice solves a serious enough problem the company is facing, and provides perceived value within a reasonable time.

Looking at how continuous experimentation relates to the three conditions. The interview data shows that only the interviewee from company D had heard about continuous experimentation. They were not able to describe continuous experimentation in more detail, which would suggest that they had no prior experience of continuous experimentation. Therefore, the data would suggest that none of the case companies had previous experience of using continuous experimentation.

The second condition, regarding the amount of resources to adopt continuous experimentation, is not so clear. The companies have been doing experimentation in

some form previously, and making the experimentation more systematic could be a matter of educating the company employees. However, establishing an experimentation process and the required technical infrastructure would require some work. Additionally, depending on the kind of experiments, building multiple implementations of the same functionality would require more resources than building a single one.

The third condition of a tool solving a concrete problem and providing perceived value is more complex than the previous two. There is no doubt that at least companies A and C where had not yet established a decent user base. They were effectively looking for a product market fit and trying to validate their assumptions with the first versions of their application. The core idea of continuous experimentation is to help companies to validate their assumptions through experimentation. However, these two companies appeared to be less inclined to do systematic experimentation than they were to develop features from the roadmap and see what users would say about the next version. It could be that continuous experimentation would be the right tool for validating their products, but these companies would see it more as a problem of not building the product fast enough. Therefore, the companies would not be able to perceive continuous experimentation as the right tool for their problem and would also fail to perceive the value of using the tool.

5 Discussion

This study investigated the software development practices of five startup companies from Maria 01[16] and A Grid[17] communities. The research questions focused on mapping the software development and product management practices used in the case companies, how case companies collected and used data in their product development practices, and what factors affect the adoption of continuous experimentation.

5.1 The tools and practices used by the companies

Concerning RQ1, all five case companies were building their products incrementally using agile or lean software development practices. The companies were not only using iterative development, but had also set up advanced CI/CD pipelines with code versioning and automatic builds. Only the mobile application builds of the company A were built manually by developers instead of using automated pipelines. Surprisingly, the software development practices used by all five companies resembled each other significantly. The only major difference being automated testing, which was only used by two out of the five companies. Comparing the data to Klotins, Unterkalmsteiner, Chatzipetrou, et al. (2021) study of 84 startups, we can see that our case companies have significantly higher adoption rate of key agile practices such as continuous integration, continuous deployment, backlog, version control, iterative development, and incremental development. For comparison, the use of version control and backlog were the only practices with higher than 35% adoption rate in Klotins' data-set. Looking at a smaller data-set of Finnish companies by Lindgren and Münch (2015), the CI adoption rate was reportedly adopted by all companies in both studies.

Looking at other agile practices, three of the companies had adopted agile practices rather selectively. All five companies had found to not follow any particular practices to the letter. For example, company A, C, and E were not using automated testing. Company A had also not set up automated builds for their mobile applications. Curiously, the company A was also the only company doing server software releases into production without a manual confirmation step. Company E were doing backlog refinement during dailies instead of a proper backlog refinement meeting. Similar data was also found by Klotins, Unterkalmsteiner, Chatzipetrou, et al. (2021) who also warn that selectively picking only some agile practices, and failing to adopt supporting practices, may lead to adverse effects.

Four out of the five companies appeared to be doing their product planning more or less on the level of sprint iteration or similar timescale. All companies mentioned using a backlog and a roadmap, but only the data regarding company D indicated well defined practices for maintaining the roadmap and backlog. Klotins, Unterkalmsteiner, and Gorschek (2019) also reported similar results where most common scope of planning was the current iteration or the next release. Research by Gralha et al. (2018) also found that especially early stage companies tend to see

[16]Maria 01 community. URL: https://maria.io (visited on 12/28/2021)
[17]A Grid community. URL: https://agrid.fi (visited on 12/28/2021)

up front planning and generation of documentation as waste. They also found that companies tend to become more structured, planned, and documentation-oriented as they mature. This view would align well with the fact that the company D, being older than the other case companies, appeared to be working in fairly structured manner.

Experimentation was used frequently by companies A, C, and D. Company B reported only one instance of experimentation, and company E data did not indicate any experimentation activities. The company D was severely limited due to the nature of their product, and the B2B distribution model. Companies C and D were using experimentation frequently in their product development processes. While frequent, the company C's approach to experimentation did not appear to be too structured when it comes to data collection and planning of experiments. On the other hand, the company D had set up good prototyping practices and there appeared to be a plan for the analysis, usage, and refinement of experimental data. The observation of only one company doing somewhat systematic experimentation is in line with the findings of Lindgren and Münch (2015). They also reported that non-systematic experimentation was common among their 10 case companies. Likewise, their data indicated that systematic experimentation among startup companies was rare. The company B appeared to be willing to experiment more than they were able, but faced many similar B2B related problems as reported by Rissanen and Münch (2015).

5.2 Use of data

RQ2 concerned the use of data in the case companies. Four out of the five companies had good access to user data. All four companies collected qualitative user feedback. The most common channels for qualitative feedback was user interviews, feedback forms, and email. Company D had an embedded service chat in their service. Company A reported receiving feedback through social media and application stores. Company E was using internal chat channel for collecting feedback about their internal services. Overall, the four companies had established solid ways for collecting qualitative data.

Company B was operating in a B2B model, which practically prevented from accessing qualitative end user data. They had to make do with the data their business partners were willing to share. Similar findings in B2B context was also reported by Rissanen and Münch (2015).

Four out of the five companies had built instrumentation for collecting event level quantitative data. However, only companies A and D had good automated access to UI level events. Company B was limited to backend data, and company C had chosen to implement an opt-in data collection mechanism primarily for debugging purposes. Company E had chosen to not instrument their internal services. The companies A and D had demonstrated product improvement as a result of their using event level data.

The company D had the most advanced data use practices of the five case companies. The other four companies appeared to analyze the data on a weekly basis and generate work items based on new findings. However, there was no clear

indication of systematic data refinement, storage, and use. Company D on the other hand had set up a central data repository and even had a dedicated team taking care of the company data. Company D was the only case company that appeared to be using data in a continuous fashion. They were using data to predict how their customer needs would evolve in the future and there was also a link between the company strategy, product planning, and investor defined goals. Their use of DIBB Kniberg (2016) framework and formulation of beliefs and bets show many similarities to continuous experimentation.

Lindgren and Münch (2016) also report on that companies generally employ varying data collection techniques and are generally good at collecting data. However, they also found that the stakeholders involved in the data collection varied between companies. Sometimes, especially on smaller companies, the development team was heavily involved. In larger companies, the data analysis was done by the management and the development team was mostly executing the product backlog. Similar dichotomy can be seen with the five case companies of the present study. Company D, being the largest with 3 development teams, was using data to define OKR items for the development teams to execute. The other case companies typically had the development team involved when planning the work and going through any new findings.

5.3 Adoption of continuous experimentation

RQ3 asked about the factors affecting the adoption of continuous experimentation in the case companies. Looking at results so far, companies appear to be extremely selective when choosing their tools and practices. Furthermore, three factors would appear to be linked to adoption of tools and practices. First, the data indicates that previous experience of using some tool of practice repeatedly mentioned as the reason for selecting a certain tool or practice.

Second, all case companies reported lack of resources and need for prioritization. Four out of the five companies also mentioned limited funding and securing more funding to guide their priorities. The fear of running out of time and money may prevent the companies from taking any unnecessary risks with trying out new tools and practices. Especially, if adopting new tools or practices requires large investments in the form of time and resources.

Third, the companies appear to want to have tangible benefits soon after adopting new tools and practices. This makes perfect sense when one takes into account the fact that many of these startups may not be around in six months, unless they manage to secure their next round of funding. With their limited resources, it does not make sense to waste time and effort trying to adopt something that is not believed to provide tangible value before the current estimated lifetime of the organization.

Given the three above mentioned factors, it could be argued that companies are not likely to adopt continuous experimentation easily. First, only one of the five companies had heard the term continuous experimentation prior to the interview. Second, starting systematic experimentation most likely requires expertise not readily available in a startup. Third, the companies may not be able recognize the value

of systematic experimentation. Specifically, a company may believe they are better off building good enough products without spending extra resources on building multiple implementations of their features.

The adoption of continuous experimentation is covered in earlier studies by e.g. Yaman (2019), Olsson, Alahyari, and Bosch (2012), and Lindgren and Münch (2016). All of the studies acknowledge that using continuous experimentation requires significant skill and coordination from the whole organization. Therefore, the adoption of continuous experimentation is seen as a journey that should be started gradually. Lindgren and Münch (2016) and Yaman (2019) propose starting with small scale experimentation and building the technical and organization capabilities gradually on each subsequent round of experimentation. The gradual approach is seen as a low cost way for adopting the practice. Also Fagerholm et al. (2017) suggest their experimentation framework to be implemented gradually based on the organizational capabilities.

Yaman (2019) also proposes having a continuous experimentation champion to push the change inside the organization. The idea of a champion aligns well with the identified need for prior experience, as well as the startup companies suggested inability to see the value of systematic experimentation. The champion would certainly have knowledge of continuous experimentation, have the required expertise to bootstrap the experimentation practice without spending too much time and resources, and be able to make the rest of the organization see the value of doing systematic experimentation.

Increasing the awareness of continuous experimentation and training these champions e.g. by universities or during entrepreneurship courses, similar to Camuffo et al. (2020), could boost the adoption rate of continuous experimentation. Based on the interview data, it does not look likely that the companies would start using continuous experimentation unless at least one of their employees had previous experience of the practice.

When it comes to the value proposition of continuous experimentation, it is easy to be inspired by reports such as the Google's $200 million revenue increase from finding the perfect shade of blue Hern (2014). Arguably, this kind of success stories are not applicable for small startup companies with five figure revenue numbers. The potential upside of improving a click rate by e.g. 5 percent is most likely in the order of thousands of dollars rather than millions. Furthermore, a well resourced company can afford dedicating a small crew for this kind of experimentation. The rest of the development work would not be greatly affected by not having a few hands on deck for a week or two. The equation looks wildly different for a startup with two developers. The opportunity cost of running an experiment effectively is to not being able to do any other development at all. With relatively immense opportunity cost and a relatively small potential upside, it is easy to understand why a startup company would prefer to simply continue executing their roadmap. This would suggest that the benefits of continuous experimentation may appear different for organizations in different stages of their life-cycle. This is a topic the previous research would appear to have failed to address.

5.4 Limitations of the study

The findings of the study are generally in line with the previous studies. The agile tools and practices reported by the companies are similar to what have been reported in previous studies. Also, the practices used for product development planning and data analysis appear to be in line with previous studies. The way how startups choose to selectively pick their tools and not follow generally acknowledged best practices has been reported previously.

While the results appear to be in line with previous research, the results are only the researchers interpretation of the collected interview data. The interview data from only one interviewees per company provides only a limited view into the case companies. This was apparent during the data collection as technical matters had to be discussed with secondary interviewees on two occasions. Furthermore, the data set only contains 5 case companies, which does not warrant wider generalisation of the results. However, seeing the results aligning with the results from previous studies would seem to increase the credibility of the study.

6 Conclusions

The thesis was set to study the use of continuous experimentation in the Finnish startups. The research attempted to describe what product development tools and practices the companies were using and what were the factors leading to the choices the companies had made. Furthermore, the study aimed to answer the question of how these companies collected user data and how the collected data was used in product development.

Based on the results, all companies were using agile or lean software development practices as well as continuous software development practices such as continuous integration Kim et al. (2008) and continuous deployment Claps, Berntsson Svensson, and Aurum (2015).

Based on the results, the product development planning practices used by the companies consisted mostly on weekly or biweekly meetings which suggested that the typical planning period was aligned with sprints or release cycles.

Regarding the collection of data, most companies were able to collect qualitative user data and two of the companies were also doing automatic collection of user interface event level data from their services. The companies used the collected data mostly for validating that their services did not contain errors. Only one company appeared to be systematically using the data for predicting future user requirements. Regarding continuous experimentation, the practice is not commonly known as only one of the five interviewees had heard about the practice when asked. Furthermore, the one person was not able to describe the practice in detail. Additionally, previous experience and expected short term value appear to be significant factors when companies are selecting tools and practices. All five companies struggled with limited resources. Resource limitations forced the companies to carefully prioritize the work. Limited resources were also given as the reason for not implementing heavy coordination processes or not using engineering best practices. Together, these findings appear to be significant when considering the possible actions that could make continuous experimentation more widely adopted.

The study found evidence that making software developers and entrepreneurs more aware of continuous experimentation could improve the adoption rate of the practice. Future work would be needed to investigate the best ways to raise this awareness. Potential approaches could be teaching the practice in university curricula or preparing an entrepreneurship training programs with continuous experimentation related material. The latter approach by Camuffo et al. (2020) have proved to be somewhat promising.

References

Amatriain, Xavier (2013). "Beyond data: from user information to business value through personalized recommendations and consumer science". en. In: *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management - CIKM '13*. San Francisco, California, USA: ACM Press, pp. 2201–2208. DOI: 10.1145/2505515.2514701.

Auer, Florian et al. (June 2021). "Controlled experimentation in continuous experimentation: Knowledge and challenges". en. In: *Information and Software Technology* 134, p. 106551. ISSN: 09505849. DOI: 10.1016/j.infsof.2021.106551.

Bosch, Jan (2012). "Building Products as Innovation Experiment Systems". en. In: *Software Business*. Ed. by Wil van der Aalst et al. Vol. 114. Series Title: Lecture Notes in Business Information Processing. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 27–39. DOI: 10.1007/978-3-642-30746-1_3.

Bourque, P. and R.E Fairley (2014). *Guide to the Software Engineering Body of Knowledge, Version 3.0*. online. URL: www.swebok.org.

Camuffo, Arnaldo et al. (Feb. 2020). "A Scientific Approach to Entrepreneurial Decision Making: Evidence from a Randomized Control Trial". en. In: *Management Science* 66.2, pp. 564–586. DOI: 10.1287/mnsc.2018.3249.

Cerdeira, Nicolás and Kyril Kotashev (2021). *Startup Failure Rate: Ultimate Report + Infographic [2021]*. en. URL: https://www.failory.com/blog/startup-failure-rate (visited on 12/20/2021).

Claps, Gerry Gerard, Richard Berntsson Svensson, and Aybüke Aurum (Jan. 2015). "On the journey to continuous deployment: Technical and social challenges along the way". en. In: *Information and Software Technology* 57, pp. 21–31. DOI: 10.1016/j.infsof.2014.07.009.

Ebert, Christof et al. (May 2016). "DevOps". en. In: *IEEE Software* 33.3, pp. 94–100. DOI: 10.1109/MS.2016.68.

Fabijan, Aleksander, Pavel Dmitriev, Helena Holmstrom Olsson, et al. (Mar. 2020). "The Online Controlled Experiment Lifecycle". en. In: *IEEE Software* 37.2, pp. 60–67. DOI: 10.1109/MS.2018.2875842.

Fabijan, Aleksander, Pavel Dmitriev, Helena Holmstrom Olsson, et al. (Aug. 2017). "The Benefits of Controlled Experimentation at Scale". en. In: *2017 43rd Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*. Vienna, Austria: IEEE, pp. 18–26. DOI: 10.1109/SEAA.2017.47.

Fagerholm, Fabian et al. (Jan. 2017). "The RIGHT model for Continuous Experimentation". en. In: *Journal of Systems and Software* 123, pp. 292–305. DOI: 10.1016/j.jss.2016.03.034.

Fitzgerald, Brian and Klaas-Jan Stol (Jan. 2017). "Continuous software engineering: A roadmap and agenda". en. In: *Journal of Systems and Software* 123, pp. 176–189. DOI: 10.1016/j.jss.2015.06.063.

Gomez-Uribe, Carlos A. and Neil Hunt (Jan. 2016). "The Netflix Recommender System: Algorithms, Business Value, and Innovation". en. In: *ACM Transactions on Management Information Systems* 6.4, pp. 1–19. DOI: 10.1145/2843948.

Gralha, Catarina et al. (May 2018). "The evolution of requirements practices in software startups". en. In: *Proceedings of the 40th International Conference on Software Engineering*. Gothenburg Sweden: ACM, pp. 823–833. DOI: 10.1145/3180155.3180158.

Hern, Alex (Feb. 2014). "Why Google has 200m reasons to put engineers over designers". In: *The Guardian*. ISSN: 0261-3077. URL: https://www.theguardian.com/technology/2014/feb/05/why-google-engineers-designers (visited on 12/18/2021).

Kim, Seojin et al. (Sept. 2008). "Automated Continuous Integration of Component-Based Software: An Industrial Experience". en. In: *2008 23rd IEEE/ACM International Conference on Automated Software Engineering*. L'Aquila, Italy: IEEE, pp. 423–426. DOI: 10.1109/ASE.2008.64.

Klotins, Eriks, Michael Unterkalmsteiner, Panagiota Chatzipetrou, et al. (2021). "SIoT Framework: Towards an Approach for Early Identification of Security Requirements for Internet-of-things Applications". en. In: *e-Informatica Software Engineering Journal* 15.1. DOI: 10.37190/e-Inf210103.

Klotins, Eriks, Michael Unterkalmsteiner, and Tony Gorschek (Feb. 2019). "Software engineering in start-up companies: An analysis of 88 experience reports". en. In: *Empirical Software Engineering* 24.1, pp. 68–102. DOI: 10.1007/s10664-018-9620-y.

Kniberg, Henrik (June 2016). *Spotify Rhythm - how we get aligned (slides from my talk at Agile Sverige)*. URL: https://blog.crisp.se/2016/06/08/henrikkniberg/spotify-rhythm (visited on 12/16/2021).

Kohavi, Ron, Randal M. Henne, and Dan Sommerfield (2007). "Practical guide to controlled experiments on the web: listen to your customers not to the hippo". en. In: *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '07*. San Jose, California, USA: ACM Press, p. 959. DOI: 10.1145/1281192.1281295.

Kvale, Steinar and Svend Brinkmann (2015). *InterViews: learning the craft of qualitative research interviewing*. Third edition. Los Angeles: Sage Publications. ISBN: 978-1-4522-7572-7.

Leffingwell, Dean (2007). *Scaling software agility: best practices for large enterprises*. The Agile software development series. Upper Saddle River, NJ: Addison-Wesley. ISBN: 978-0-321-45819-3.

Lindgren, Eveliina and Jürgen Münch (2015). "Software Development as an Experiment System: A Qualitative Survey on the State of the Practice". In: *Agile Processes in Software Engineering and Extreme Programming*. Ed. by Casper Lassenius, Torgeir Dingsøyr, and Maria Paasivaara. Vol. 212. Series Title: Lecture Notes in Business Information Processing. Cham: Springer International Publishing, pp. 117–128. DOI: 10.1007/978-3-319-18612-2_10.

— (Sept. 2016). "Raising the odds of success: the current state of experimentation in product development". In: *Information and Software Technology* 77, pp. 80–91. DOI: 10.1016/j.infsof.2016.04.008.

Merriam, Sharan B. and Elizabeth J. Tisdell (2015). *Qualitative research: a guide to design and implementation*. Fourth edition. The Jossey-Bass higher and adult education series. San Francisco, CA: John Wiley & Sons.

Olsson, Helena Holmstrom, Hiva Alahyari, and Jan Bosch (Sept. 2012). "Climbing the "Stairway to Heaven" – A Mulitiple-Case Study Exploring Barriers in the Transition from Agile Development towards Continuous Deployment of Software". In: *2012 38th Euromicro Conference on Software Engineering and Advanced Applications*. Cesme, Izmir, Turkey: IEEE, pp. 392–399. DOI: 10.1109/SEAA.2012.54.

Olsson, Helena Holmström and Jan Bosch (2014). "The HYPEX Model: From Opinions to Data-Driven Software Development". en. In: *Continuous Software Engineering*. Ed. by Jan Bosch. Cham: Springer International Publishing, pp. 155–164. DOI: 10.1007/978-3-319-11283-1_13.

Poppendieck, Mary and Tom Poppendieck (2010). *Lean software development: an agile toolkit*. Nachdr. The agile software development series. Boston: Addison-Wesley. ISBN: 978-0-321-15078-3.

Ries, Eric (2011). *The lean startup: how today's entrepreneurs use continuous innovation to create radically successful businesses*. 1st ed. New York: Crown Business. ISBN: 978-0-307-88789-4.

Rissanen, Olli and Jurgen Münch (May 2015). "Continuous Experimentation in the B2B Domain: A Case Study". In: *2015 IEEE/ACM 2nd International Workshop on Rapid Continuous Software Engineering*. Florence: IEEE, pp. 12–18. DOI: 10.1109/RCoSE.2015.10.

Shadish, William R., Thomas D. Cook, and Donald T. Campbell (2001). *Experimental and quasi-experimental designs for generalized causal inference*. Boston: Houghton Mifflin. ISBN: 978-0-395-61556-0.

Sommerville, Ian (2007). *Software engineering*. 8th ed. International computer science series. Harlow, England ; New York: Addison-Wesley. ISBN: 978-0-321-31379-9.

Steiber, Annika and Sverker Alänge (2013). "A corporate system for continuous innovation: the case of Google Inc." In: *European Journal of Innovation Management* 16.2, pp. 243–264. DOI: 10.1108/14601061311324566.

Thomke, Stefan H. (June 1998). "Managing Experimentation in the Design of New Products". In: *Management Science* 44.6, pp. 743–762. DOI: 10.1287/mnsc.44.6.743.

Yaman, Sezin Gizem (Oct. 2019). "Initiating the Transition towards Continuous Experimentation: Empirical Studies with Software Development Teams and Practitioners". en. ISBN: 978-951-51-5543-6. Doctoral thesis. Helsinki, Finland: University of Helsinki.

Yin, Robert K. (2012). *Applications of case study research*. 3rd ed. Thousand Oaks, Calif: SAGE. ISBN: 978-1-4129-8916-9.

A Interview guide

Key areas on interest

- Background information
    - The product the company is building
    - How long has the product been in development
    - How long has the product been available
- Software development practices
    - Use of agile methods
    - Level of automation
    - Deployment process
    - How these practices have evolved
        - How is a need for change recognized
- User data collection, analysis, and usage
    - How these practices have evolved
        - How is a need for change recognized

About the interview

Interview is for my masters thesis only. The interview is recorded in order to produce a transcript for further analysis. The transcript or the original recording is not published or shared with third parties. Thesis supervisor and advisor may get to read an anonymised version of the transcript.

Companies and interviewees are not mentioned by name. Short direct quotations may be used in the thesis if that is ok with the interviewee.

Background information

Interviewee

- What is your current position in the company? (LM)
- How many years have you worked in the company? (LM)

Company

- Can you describe the product or service your company is building?
    - Business model?
    - Physical device? SaaS service? Installable application? Something else?
- How long has the company developed the product(s) you are currently working with?
- How long has the product been available for customers?
- What are the next steps for the product
    - Still looking for market fit?
    - Building a rodmap?
    - Executing a roadmap?

Software development practices

- What kind of software development process do you use? (LM)
- Do you use continuous integration? (LM)
    - How about continuous delivery or continuous deployment
- How often do you deploy new versions to production? (LM)
- Can you describe the process of releasing a new version to production?
    - Who
    - How much manual work is involved in the release process?
    - Who were involved in the process?
      (e.g. Business Analyst, Product Owner, Data Scientist, SD, QA, DevOps, Release Engineer)

Software development practice evolution

- How have your software development practices changed during your time?
- Can you describe a time when a new practice/tool/process was taken into use?
    - What triggered the need for new practice/tool/process?
    - How were replacement options evaluated?
    - Who were involved in the process?
      (e.g. Business Analyst, Product Owner, Data Scientist, SD, QA, DevOps, Release Engineer)

Customer data collection and analysis

- How do you make sure that you are building the right product? (LM)

- How do you collect customer feedback? (LM)
  a. Before development (LM)

b. During development (LM)
b. After deployment (LM)
(Possible prompts: informal channels, interviews, surveys, support systems, prototyping, development demos, usability tests, alpha / beta tests, A/B tests etc. (LM))

- How often are the aforementioned customer feedback collection methods used? (LM)
- Do you collect data about customer behavior, for example in the form of product usage data? (LM)
- How do you use the collected customer feedback and other data? Is there a link to:
  a. Product development (LM)
  b. Further innovation (LM)
  c. Business goals and strategy (LM)
- Can you describe how you review the collected customer feedback and other data?
  - Who is involved in reviewing the collected customer feedback and other data? (LM)
- Do you think your current practices of customer feedback collection and customer involvement are adequate? (LM)
  a. If not already ideal: How should they ideally be performed in the future? (LM)
- Are there any obstacles to collecting and using deeper customer insights? (Possible prompts: technical issues, lack of resources, personnel skills, company culture etc.) (LM)
- What are the factors that support collecting and using customer insights in your company?
  (Possible prompts: technical know-how, ample resources, personnel skills, company culture etc.)

Experimentation

If experimentation has not come up before

- Have you run experiments to test how users would react to some changes? (RQ1)
- Are you familiar with the term continuous experimentation?
  - If yes: Ask for description

    - If yes: Would you say you practice continuous experimentation?
    - If yes: Are you planning to use continuous experimentation?

- Can you describe how your experiment was carried out? (If no experiment available, ask for description of something from methods mentioned in customer data collection and analysis) (RQ1)

    - What was experimented?
    - Why was experimentation chosen as the method this time?
    - How were metrics collected
    - How were metrics analyzed
    - What kind of decision was made based on the results?

- How is the result of experiment used in decision-making? (RQ2)

- How do you decide if something needs to be experimented? (RQ1)

- Can you describe how your use of experiments has changed over the time? (RQ3)

- Are there any obstacles to experimenting as much as you would like to? (RQ1)

Closing questions

- What would you want to improve in your product development if you weren't limited by your current resources or schedules?
    - Ask for another thing to improve
- Is there something you believe could be relevant to the topics discussed that I should have asked about but didn't?

(LM): From Lindgren, E., Münch, J., 2016. Raising the odds of success: the current state of experimentation in product development. Information and Software Technology 77, 80–91. https://doi.org/10.1016/j.infsof.2016.04.008