View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Helsingin yliopiston digitaalinen arkisto

Date of Acceptance Grade

Supervisors:

Jürgen Münch

Tomi Männistö

Exploring Software Development as an Experiment System: An Interview-Based Qualitative Survey

Eveliina Lindgren

MSc thesis

Helsinki 03.02.2015

UNIVERSITY OF HELSINKI
Department of Computer Science

HELSINGIN YLIOPISTO – HELSINGFORS UNIVERSITET – UNIVERSITY OF HELSINKI

Tiedekunta – Fakultet –Faculty Faculty of Science	Laitos – Institution – Department Department of Computer Science

Tekijä – Författare – Author

Eveliina Lindgren

Työn nimi – Arbetets titel – Title

Exploring Software Development as an Experiment System: An Interview-Based Qualitative Survey

Oppiaine – Läroämne – Subject

Computer Science

Työn laji – Arbetets art – Level MSc thesis	Aika – Datum – Month and year 03.02.2015	Sivumäärä – Sidoantal – Number of pages 59 pages + 11 appendix pages

Tiivistelmä – Referat – Abstract

An experiment-driven approach to software product and service development is getting increasing attention as a way to channel limited resources to the efficient creation of customer value. In this approach, software functionalities are developed incrementally and validated in continuous experiments with stakeholders such as customers and users. The experiments provide factual feedback for guiding subsequent development.

Although case studies on experimentation conventions in the industry exist, an understanding of the state of the practice is incomplete. Furthermore, the obstacles and success factors of continuous experimentation have been little discussed. To these ends, an interview-based qualitative survey was conducted, exploring the experimentation experiences of ten software development companies.

The study found that although the principles of continuous experimentation resonated with industry practitioners, the state of the practice was not mature. In particular, experimentation was rarely systematic and continuous. Key challenges related to changing organizational culture, accelerating development cycle speed, measuring customer value and product success, and securing resources. Success factors included an empowered organizational culture and deep customer and domain knowledge. There was also a good availability of technical tools and competence to support experimentation.

ACM Computing Classification System (CCS):
Software and its engineering → Software creation and management → Software development process management → Software development methods

Software and its engineering → Software creation and management → Software verification and validation → Empirical software validation

Avainsanat – Nyckelord – Keywords
experimentation, customer feedback, experiment-driven software development, qualitative survey

Säilytyspaikka – Förvaringställe – Where deposited

Muita tietoja – Övriga uppgifter – Additional information

Contents

1 Introduction

New possibilities to observe customers and collect customer feedback allow software-centric companies to shorten learning cycles and improve their understanding of customer value. A potential approach is to build software products and services (henceforth, products) by continuously deploying new functionalities and features to customers. Instead of relying on pre-defined requirements or opinion-based assumptions, the customer value of the products is validated in their actual marketplace by conducting a constant series of experiments. This experiment-driven approach is currently most prevalent in the cloud computing environment, but it is beginning to affect the development of all Internet-connected software products [11].

During the last decades, agile software development methods have permeated the industry [63]. Agile development has changed the way software is developed for instance by advocating iterative and incremental development, embracing changing requirements, and highlighting the importance of customer feedback [66]. However, Holmström Olsson et al. [29] suggest that the application of agile methods within the research and development (R&D) organization is only one stage on the maturation path of companies' software engineering practices. The following stages are the continuous integration and deployment of R&D output, and finally, R&D as an experiment system. At this stage, development is based on continuous experiments that utilize instant customer feedback and product usage data to identify customer needs. This final stage is further systematized by Bosch [11]. He emphasizes constantly generating new ideas to test with customers, suggesting that the approach is best described as an innovation experiment system.

The lean startup methodology [50] provides a generic framework for systematic experimentation. The methodology focuses on creating valuable products by maximizing the learning about customers and their needs. Learning is gained from hypothesis-driven experiments where customer feedback on the product is collected either directly or implicitly by observing customers' use of the product. Creating a continuous Build-Measure-Learn customer feedback loop allows the company to make empirically validated decisions about whether to proceed with their current business and product strategy or to pivot to a new direction.

Fagerholm et al. [23] build upon the abovementioned ideas and propose a framework for *continuous experimentation*. Continuous experimentation refers to the constant testing of the value of products as an integral part of the development process in order to evolve the products towards high value creation. The development process is structured by consecutive iterations of the Build-Measure-Learn feedback loop, and supported by a technical infrastructure that facilitates rapid development and experimentation.

Despite the recent interest in experimentation as an essential part of software development, industrial experimentation experiences have not been studied widely. Most reports come from eminent web-facing companies such as Adobe [3], Google [59] and Microsoft [34]. There has also been relatively little discussion about the obstacles and enabling factors encountered by practitioners in relation to continuous experimentation.

This study aims at developing an understanding of the state of the practice of using an experiment system approach to software development. The key challenges and success factors related to the approach are also identified. An interview-based qualitative survey was conducted to gather multifaceted expert information on the subjects. Ten software companies operating in Finland contributed to the study, represented in the interviews by thirteen experienced industry practitioners. The interview data was analysed to develop a cross-case overview of the companies' experimentation experiences.

This thesis is organized into eight chapters. A summary of the background themes of continuous experimentation is given in Chapter 2. Chapter 3 defines the goals of the study along with the research questions it aims to address. Chapter 4 outlines the study design, while Chapter 5 describes the execution of the design. The findings of the study are presented in Chapter 6, followed by a discussion of the results and the study's limitations in Chapter 7. Finally, the study and its main contributions along with future research prospects are summarized in Chapter 8.

2 Evolution towards continuous experimentation

This chapter presents an overview of current industry practices and trends relating to continuous experimentation. The chapter builds on the “Stairway to Heaven” model presented by Holmström Olsson et al. [29]. The model outlines the evolutionary path of a software development company working towards building a system of continuous experimentation. Figure 1 depicts the five stages of the path.

A brief synopsis of stages 1–4 of the stairway is given in Section 2.1. Section 2.2 presents the conceptual background for reaching the last stage of the stairway. Section 2.3 discusses the final stage of running an experiment system in more detail. To conclude, a short overview of current experimentation practices in the industry is presented in Section 2.4.

2.1 Transitioning to agile development

The initial stage of the “Stairway to Heaven” (Figure 1) is characterized by a mode of development associated with the single-pass waterfall model: a slow, sequential process with gated steps and a big design up front approach [29]. Development team organization is often discipline-based (e.g. programming and quality assurance), thereby limiting interdisciplinary collaboration. Furthermore, customers are typically not involved in the development process and can therefore only provide feedback at the end of the process. The first systematization of the waterfall model is habitually attributed to Royce [53].

Many practitioners, including Royce, felt that the single-pass waterfall model was not functional in the often changing and unclear conditions that typified many software projects. Iterative and incremental development (IID) has provided an alternative approach to software development since before the 1970s [38]. Its popularity began to soar in the latter half of the 1990s with the emergence of agile software development methodologies such as Dynamic Systems Development Method (DSDM) [58], Extreme Programming (XP) [7], and Scrum [56]. Nowadays agile development is the de facto choice in software projects [63]. It represents the second stage of the “Stairway to Heaven” (Figure 1).

Transitioning to agile development brings about considerable changes. The agile principles, shared by numerous agile methods, include focusing on customer value creation, involving customers in the development process, shortening development and feedback cycles, and embracing changing requirements [6]. A human-centred approach is also central: development teams should be self-organized, cross-functional, and empowered with sufficient knowledge and authority to perform to the best of their ability. Frequent cooperation between different stakeholders, such as the R&D organization, product management, and customers, is encouraged.

Adopting agile software development is the first step on the path towards continuous experimentation. The implementation of continuous integration and delivery provides additional opportunities for agile teams. These practices are discussed in the following two subsections.

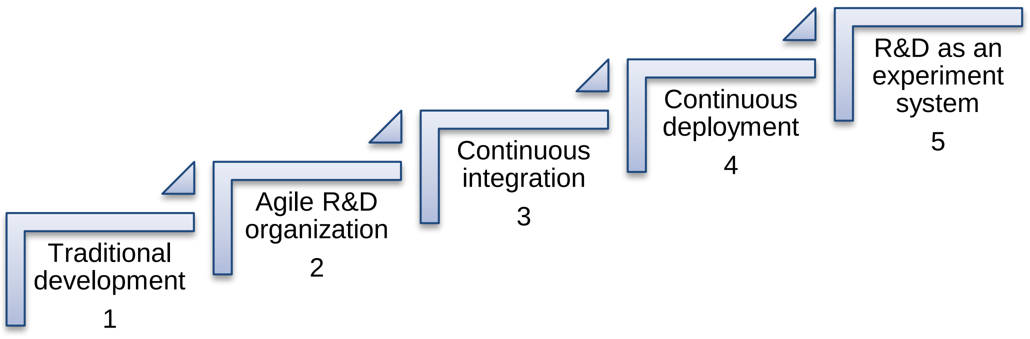

Figure 1. The maturation path of companies' software engineering practices. Adapted from [29].

2.1.1 Continuous integration

Continuous integration (CI) emerged as a practice associated with the agile software development methodology XP [7], although its use has since expanded elsewhere [63]. The central objective of CI is to ensure that a functioning version of the software product is available in the version control repository at all times [31]. This is achieved through the early and frequent merging of individual developers' work combined with the use of automated builds and comprehensive automated test suites. If a commit to version control introduces a problem, it can be immediately identified and fixed, thereby facilitating quality assurance and reducing project risks.

While automated builds and test suites typically form the core of a CI system, a host of other automated processes can also be applied [31]. These may relate, for instance, to configuration management and static testing. The utilization of CI represents stage three on the “Stairway to Heaven” (Figure 1). Reaching this stage signifies that agile practices have been extended from product development to quality assurance.

2.1.2 Continuous delivery

The objective of the fourth stage of the “Stairway to Heaven” (Figure 1) is to extend agile practices to product management and the customers [29]. This requires a continuous feedback loop between the software company and its customers. Continuous, or at least frequent, deployments of software functionality to customers are essential for achieving this goal.

Some ambiguity exists regarding how the closely related practices of continuous deployment and continuous delivery are understood (see for instance [25, 31]). In this thesis, continuous deployment refers to a practice in which every committed change that successfully passes the CI system moves along a deployment pipeline and is automatically released to production. Continuous delivery (CD) is an otherwise similar practice, but the committed changes are not deployed to production (or other environments) automatically. The manual release decision allows for an enhanced degree of control over the deployment environments, and enables pacing deployments to suit business needs.

The DevOps movement is noteworthy when discussing the building of a smooth deployment pipeline. Humble [30] contends that adopting DevOps is a prerequisite for the successful implementation of CD. Several broad themes are discussed within the DevOps movement, but the integration of product development and IT operations teams remains a central topic. A high level of deployment automation is naturally useful in this respect. The close collaboration of development and operations teams also broadens the scope of learning about customer behaviour through product usage observation. This is a key factor in the fourth stage of the “Stairway to Heaven” [29].

There are differences in how software products lend themselves to continuous or frequent deployments. The issue is discussed by Bosch [11]. Hosted cloud-based products, such as software as a service (SaaS) solutions, are typically well suited to continuous deployment due to their business model and technical infrastructure.

Conversely, on-premises installations of licensed software are subject to scheduled version upgrades whose occurrence is often minimized to avoid disruptions to business. Customer-specific configurations may further complicate deployment. Finally, Bosch notes that embedded software solutions – especially those not connected to the Internet – also pose a challenge.

A suitable deployment pace is therefore dependent on the business case. It is also noteworthy that although both continuous integration and continuous delivery greatly facilitate the building of an experiment system, not all forms of experimentation are dependent on functioning software. The matter is discussed further in Section 2.3.

Section 2.1 has reviewed the first four stages of the “Stairway to Heaven” (Figure 1). While ascending the stairway, increasingly agile practices are implemented throughout the organization. Traditional slow, waterfall-like development gives way to agile R&D supported by continuous integration. By the time continuous delivery is adopted, customers are also engaged in an iterative cycle of rapid, agile product development. In order to reach the final stage of the stairway, an even broader perspective to business and product development is required. This is provided by lean thinking, discussed in the next section.

2.2 Lean thinking and the lean startup

In addition to agile principles, lean principles underlie continuous experimentation. This section provides a short synopsis of lean thinking and its application to software development. It then introduces the lean startup methodology. Its central practices, the minimum viable product and the Build-Measure-Learn customer feedback loop, are also the key elements of continuous experimentation.

2.2.1 Lean thinking

Lean manufacturing started to attract attention in the 1990s as the western world began to systematize [67] what had brought about the success of the Toyota Production System [44] in the automotive industry. Lean thinking has subsequently expanded outside the realms of manufacturing to become a widespread management philosophy with myriad manifestations. It covers a broad spectrum of concepts, principles, and practices whose description falls outside the scope of this thesis. However, some of its key principles are briefly outlined below.

The core of lean thinking is to create customer value by eliminating waste [67]. Anything that does not create customer value – something that the customer is willing to pay for – is considered wasteful. A just-in-time system known as kanban may be used to visualize and control the process of delivering the product to the customer. The objective is to ensure the continuous, unimpeded flow of the process. Limiting work in progress is a key factor in achieving this goal. Furthermore, the process should be based on customer demand (known as “pull”) rather than pushing to achieve previously set production targets. Finally, the company should strive to perfect the process through continuous improvement.

2.2.2 Lean software development

Lean thinking has also been applied to the field of software engineering. Lean software development was originally defined in [48] and has thereafter been revised for instance in [47]. Rather than a ready-made process blueprint, lean software development offers a collection of ideas and tools that companies may adopt to improve the identification and delivery of customer value. The approach is summarized in the following seven principles: eliminate waste, build quality in, create knowledge, defer commitment, deliver fast, respect people, and optimize the whole [47]. The concept of waste remains central: examples of waste in software development include defects; unnecessary features; and processes that result in waiting, partially done work, and lost knowledge.

The abovementioned principles are highly reminiscent of the agile principles outlined in Section 2.1. Indeed, lean and agile software development are inextricably linked, and their use is often combined by industry practitioners [52]. A specific example of this is the frequent use of kanban as a process management technique in agile software projects [63]. However, different interpretations exist concerning the relationship between lean and agile (see for instance [20, 65]). In the context of this thesis, lean thinking is regarded as a higher-level, end-to-end philosophy for business development and management, with agile methods typically used during actual software development. A topical methodology that combines these two approaches is the lean startup.

2.2.3 The lean startup

The lean startup methodology was first popularized by Ries [50], drawing on his experience in applying lean principles to software startups. Despite the fact that academic research on the merits of the methodology is scarce, it has expanded into a veritable movement with numerous advocates. Blank [8] provides a succinct summarization of the lean startup approach: “It favours experimentation over elaborate planning, customer feedback over intuition, and iterative design over traditional ‘big design up front’ development.”

Regardless of its name, the lean startup methodology is not restricted to actual startups. Ries ([50] p.27) defines a startup as “a human institution designed to create a new product or service under conditions of extreme uncertainty”. This broad definition does not restrict the maturity, size, or the industry sector of the organization. On a related note, it has been suggested that applying the methodology may also offer benefits to established organizations [8, 50].

The lean startup approach draws from the notion that entrepreneurs have an ultimate vision they set out to achieve [50]. A business strategy describes how to realize that vision. The strategy essentially consists of a set of initially uncertain assumptions related, for instance, to the company’s value proposition and product offering; its resources and finances; and its stakeholders such as business partners and customers. In lieu of a detailed business plan, these hypotheses may be succinctly visualized on a business model canvas [45], or its lean startup counterpart, a lean canvas [41]. The hypotheses then need to be empirically tested to provide a factual basis for decision making [50]. It is sensible to test the riskiest, most fundamental assumptions first.

In the lean startup methodology, entrepreneurs test hypotheses by “getting out of the building”: that is, by empirically eliciting early and frequent stakeholder feedback about a so-called minimum viable product [50]. This approach is based on the customer development methodology developed by Blank [9]. The entrepreneurs then use the collected stakeholder feedback to revise the hypotheses and make appropriate decisions, after which the learning cycle begins again [50]. The minimum viable product and the iterative learning cycle are discussed in more detail in the following subsections.

2.2.4 Minimum viable product

A minimum viable product (MVP) is required to test business and product hypotheses in experiments with stakeholders [50]. An MVP is a version of the product that holds just enough value to enable its presentation to stakeholders, thereby commencing a continuous learning process (Section 2.2.5). In the course of the learning process, the MVP is developed in an agile manner, iteratively and incrementally, until the product achieves satisfactory market success or is judged as non-viable and discarded.

An MVP needs to be suitably instrumented so that the stakeholders' reaction to it can be measured [50]. A simple example of an MVP is to provide a link to a planned new feature of a web application that in fact leads to a mock-up page. The instrumentation – the hit count of the link – provides data for estimating customer interest in the new feature. However, instrumentation is not necessarily equal to quantitative product metrics: various options exist for collecting stakeholder feedback (Section 2.3.3).

Others have proposed approaches similar to the MVP under different labels. For instance, Blank [9] uses the concept "minimum feature set", in addition to which the label "minimum viable feature" (MVF) is sometimes used in the context of extending existing products with new features (e.g. [23]). In this thesis, the term "minimum viable product" is hereafter used to cover all these variations.

2.2.5 Build-Measure-Learn

Once an MVP has been conceived, the Build-Measure-Learn loop can be entered [50]. Figure 2 portrays this three-step continuous, experiment-driven learning process. First, ideas are transformed into product functionalities (Build). Second, product development progress is evaluated by collecting empirical feedback from stakeholders during hypothesis-driven experiments (Measure). Third, the collected feedback is used to reflect upon initial hypotheses and product ideas (Learn). If the tested hypothesis is validated, product development may proceed to the next iteration along the chosen path, perhaps with some alterations or optimization. However, if the stakeholder feedback refutes the hypothesis, product development may need to pivot to another direction. The overall goal is to maximize the amount of learning before running out of resources. This is achieved by minimizing the total time through the loop.

The Build-Measure-Learn loop may be seen as a variant of the renowned Plan-Do-Check-Act (PDCA) cycle [19]. Both approaches have their roots in the scientific method, in which hypotheses are proposed (Plan), experiments are designed and run to test predictions derived from the hypotheses (Do), and experiment results are analysed (Check). To this foundation, both approaches add a step which focuses on continuous improvement: Act in PDCA [19], Learn in Build-Measure-Learn [50]. Moreover, although the Build-Measure-Learn loop does not include a separate step for planning, a degree of planning is implicit in the identification of hypotheses and the design of experiments to test them.

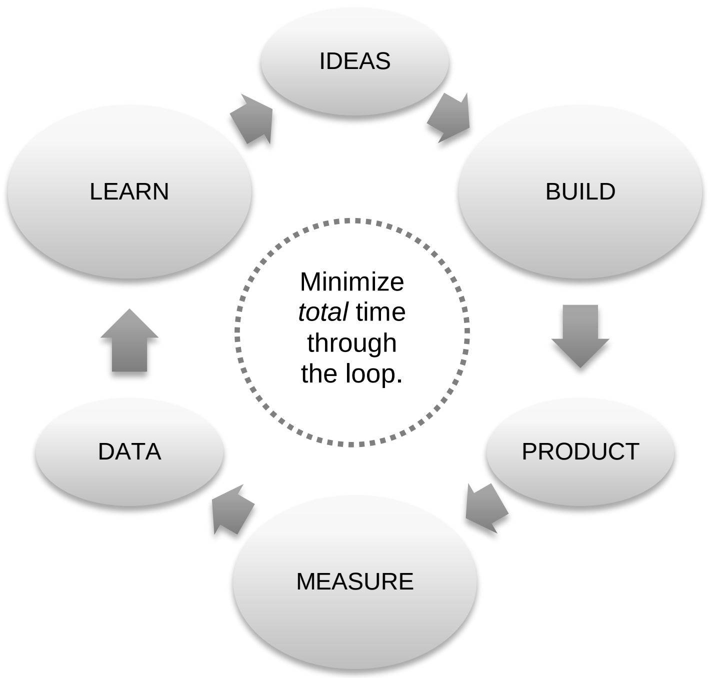

Figure 2. The Build-Measure-Learn feedback loop ([50] p. 75).

Further parallels with the Build-Measure-Learn loop can be found in the contributions of Cole [16] and Thomke [62]. Both authors focus on innovative product development in a fast-paced, turbulent environment, and propose iterative, experimental learning processes highly similar to the Build-Measure-Learn loop. The authors also discuss the use of prototypes in a manner reminiscent of MVPs in the lean startup approach. On a yet more general level, the importance of systematic experimentation in product and service development is well established: see for example [18]. Finally, Ries himself [50] mentions the Observe-Orient-Decide-Act (OODA) loop developed by John R. Boyd (and discussed for instance in [49]) as an important influence to the Build-Measure-Learn loop.

Section 2.2 has given a brief overview of lean thinking and its application to software product development. The close connection between lean and agile software development was shortly discussed. The lean startup methodology was then introduced as an example of applying lean principles to business and product development in a turbulent environment. Two central lean startup practices, the minimum viable product and the Build-Measure-Learn customer feedback loop, were discussed in more detail. The principles and practices presented in this section provide the foundation for reaching the final stage of the “Stairway to Heaven” (Figure 1). This stage of developing software products through continuous experimentation is discussed in the next section.

2.3 Implementing continuous experimentation

Based on agile and lean principles and the core practices of the lean startup approach, continuous experimentation can be implemented. This section first describes the origins and contents of the approach. It then provides viewpoints on experimentation in different phases of product development. Finally, the central concept of customer feedback is explored.

2.3.1 The experiment system approach

Holmström Olsson et al. [29] refer to the final stage of the “Stairway to Heaven” as *R&D as an experiment system*. The authors state that in such a system, a deployment is not regarded as a finalized delivery, but as a starting point for experimentation with customers to find out their needs. The data from the experiments is used to guide subsequent product development in an evolutionary manner. These overall goals of continuous learning and improvement are very similar to those discussed in Section 2.2.3 in relation to the lean startup approach.

Bosch [11] provides a more detailed analysis of the experiment system approach to software development. He enumerates its central characteristics as follows: 1) development is evolutionary and based on frequent deployments, 2) explicit and implicit customer feedback play a central role in the development process, and 3) development is focused on innovation and testing the value of R&D ideas through experiments with customers. Bosch terms this approach an *innovation experiment system*.

Bosch [11] proposes using 2–4 week R&D iterations followed by exposing the product to customers in order to collect feedback either directly or implicitly by observing product usage. The development approach recognizes three distinct phases of product development, namely pre-deployment, non-commercial deployment, and commercial deployment. Furthermore, the scope of experiment-driven development can vary from new products to new features and feature optimization.

In their descriptions of experiment systems in software product development, both Holmström Olsson et al. [29] and Bosch [11] focus particularly on the role of the R&D organization. However, technological product development, including experiments, should be aligned with the company's product vision and business strategy. A wider perspective to the matter is therefore in order.

2.3.2 A framework for continuous experimentation

Fagerholm et al. [23] combine earlier descriptions of the experiment system approach with key elements from the lean startup methodology and propose a framework for *continuous experimentation*. Consecutive iterations of the Build-Measure-Learn feedback loop structure the development process. Within each Build-Measure-Learn block, "assumptions for product and business development are derived from the business strategy, systematically tested, and the results used to inform further development of the strategy and product" [23]. Maintaining a link between the company's business goals, strategies, and the R&D aims at aligning the whole organization towards customer value creation. A similar approach, albeit with an emphasis on software measurement rather than experimentation, is explored in [5].

The experimentation process is supported by a technical infrastructure that 1) enables the lightweight releasing of minimum viable products (MVP), 2) provides means for advanced product instrumentation, and 3) supports the design, execution, and analysis of experiments [23]. A database for storing and retrieving information artefacts that are generated during the experiments is one such support function. The artefacts include experiment plans, results, and raw data collected via the product instrumentation.

Figure 3 illustrates the by now familiar activities occurring within each Build-Measure-Learn block. Fagerholm et al. [23] structure these activities further by describing the associated roles and tasks. A business analyst and a product owner collaborate with a data analyst to decide which strategic assumption to test. The assumption is formulated into a testable hypothesis. The data analyst then designs the experiment. Simultaneously, software developers and quality assurance personnel implement and deploy the MVP required for the experiment. The data analyst then oversees the execution of the experiment and analyses the resulting data. Finally, the business analyst and the product owner consider the experiment results while making the strategic decision of either proceeding with product development or pivoting to an alternative direction.

2.3.3 Experimentation throughout the product lifecycle

Consistent with the systematizations presented in [11, 23], this thesis takes a broad perspective to experimentation in software product development. Experimentation is possible throughout the product lifecycle, from the initial phases of problem definition and requirements engineering to the last stages of software evolution. The only prerequisite is the availability of an MVP or a more mature product version whose effect on stakeholders can be observed. Functioning software is not necessarily required.

A selection of experimentation techniques for different development scenarios is presented in [11, 27]. During the early phases of product development, techniques such as stakeholder interviews, mock-ups, and prototypes can be used. Alpha and beta testing are common examples of techniques used during software construction. Once an initial product version has been deployed to customers, controlled experiments such as A/B tests are applicable. Observing product performance and usage is another fruitful way to collect genuine feedback.

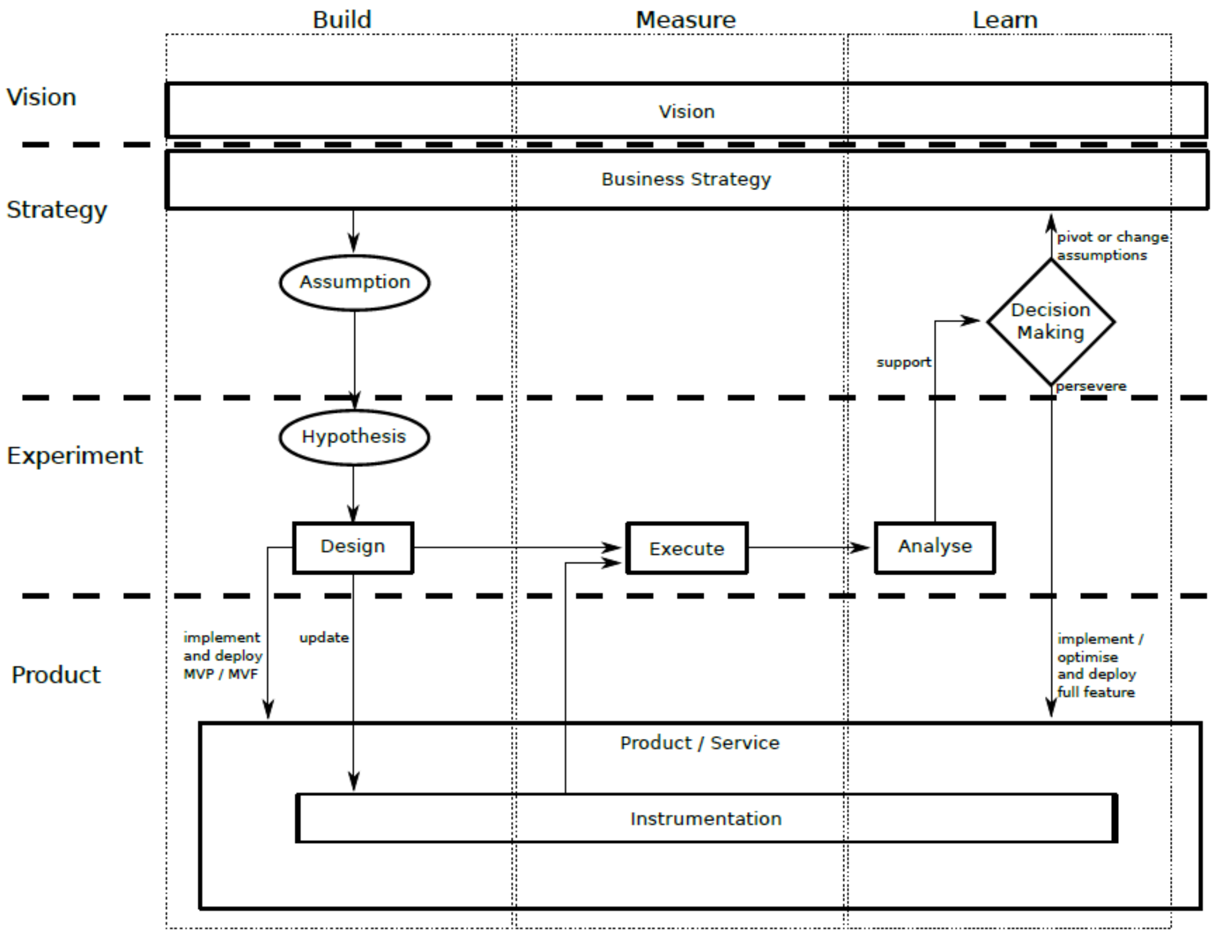

Figure 3. Activities within a Build-Measure-Learn block [23].

However, specific technical and organizational capabilities are required to enable the systematic collection and utilization of product usage data. To this end, Holmström Olsson and Bosch [28] have proposed a framework for companies hoping to improve their product data usage. The framework is portrayed in Figure 4. It is based on five levels that represent increasingly sophisticated purposes for using the data. These range from 1) basic performance monitoring to 2) supporting troubleshooting through diagnostic data, 3) understanding the usage of individual features, 4) improving existing features, and 5) developing new features experimentally. The right-hand side text boxes in Figure 4 summarize the actions required to reach each level. The peak may be seen as analogous to a fully functioning experiment system (stage 5 in Figure 1).

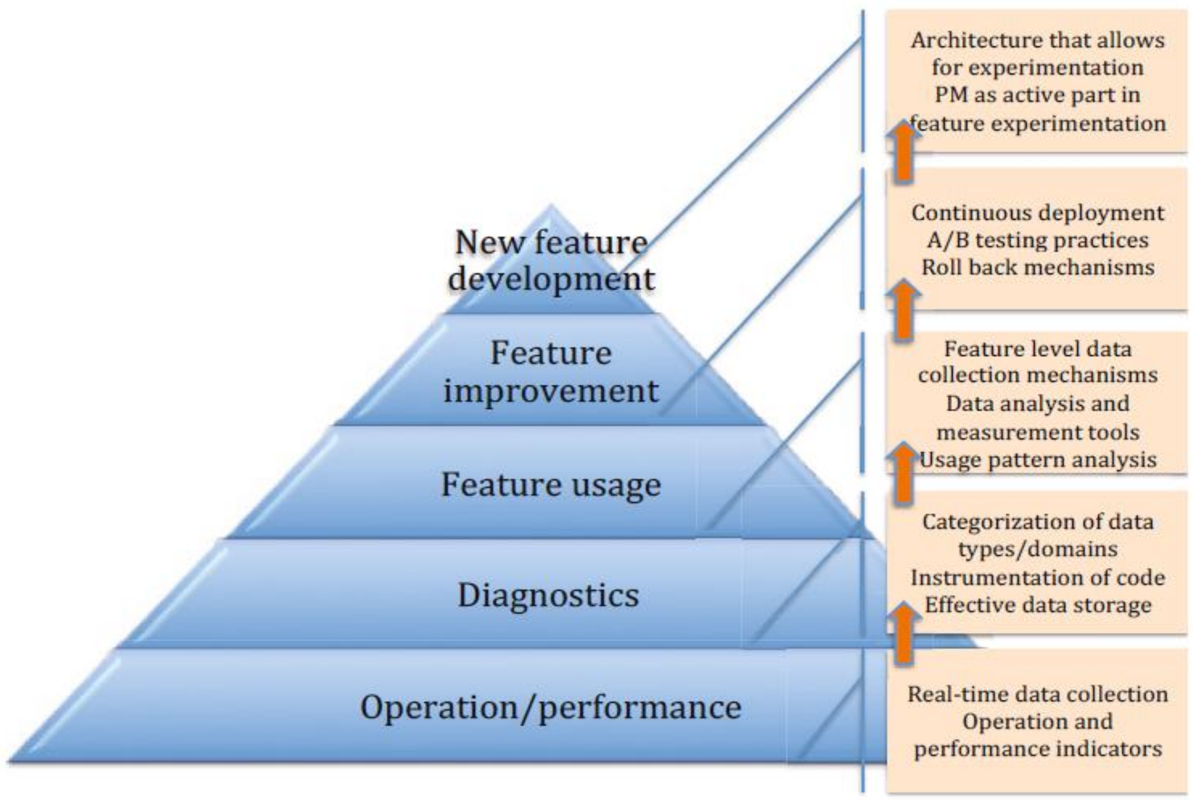
New feature development
Feature improvement
Feature usage
Diagnostics
Operation/performance
Architecture that allows for experimentation
PM as active part in feature experimentation
Continuous deployment
A/B testing practices
Roll back mechanisms
Feature level data collection mechanisms
Data analysis and measurement tools
Usage pattern analysis
Categorization of data types/domains
Instrumentation of code
Effective data storage
Real-time data collection
Operation and performance indicators

Figure 4. Framework for improving product data usage [28].

2.3.4 Customer feedback in continuous experimentation

In continuous experimentation, various business stakeholders may provide feedback on the product [23]. However, this thesis focuses solely on customer feedback since it is the most crucial factor in terms of evaluating created customer value. Building on the lean definition of value (Section 2.2.1), a customer is defined as someone who pays for the product. In some cases, especially in business-to-business software, this is a separate role from that of the end user of the product. Since product success necessitates taking both of these roles into account, the term “customer” is hereafter used in this thesis to encompass both of these roles. In cases where the distinction between customers and end users is relevant, it is explicitly stated.

For the purposes of this thesis, two forms of customer feedback are identified: 1) explicit, actively offered input, and 2) implicit, passively generated data. Examples of explicit customer feedback include data gathered from interviews, usability studies, and surveys. Depending on the method of feedback collection, explicit customer feedback is either qualitative (e.g. in-depth interviews) or quantitative (e.g. statistical surveys). Implicit customer feedback is generated when the customer uses the product, and is logged in the form of product usage and performance data. It therefore provides quantitative feedback on customer behaviour. The abovementioned definitions are founded on those used in [11]. Henceforth in this thesis the term “customer feedback” is used to include both explicit and implicit forms of feedback unless otherwise stated.

Section 2.3 has provided an overview of building software products through continuous experimentation. This lean startup-based approach seeks to ensure the identification and delivery of customer value through frequent experiments in which customer feedback on the product is collected. Variations of the approach can be used throughout the product lifecycle, and the level of sophistication of the experiment system can vary according to needs. Although academic systematizations of the experiment system approach are relatively recent, elements of it are already utilized in the industry. Examples of these industry practices are shortly outlined in the next section.

2.4 Experiment systems in the industry

Microsoft’s experiences with systematic online controlled experiments are recounted in numerous reports, for instance [34-36]. The reports provide thorough descriptions of designing, executing, and analysing rigorous controlled experiments, especially at a large scale of up to hundreds of concurrent experiments. They also consider the requirements that systematic experimentation places on organizational culture, engineering practices, as well as analytic and statistical abilities.

Google purports to experimentally evaluate almost every change that has the potential to affect user experience [60]. The authors describe the infrastructure requirements of running such a large-scale experiment system, as well as the tools and educational processes associated with its use. Supporting and fostering innovation is a key element of the Google experiment system. A detailed analysis of the company’s approach to continuous innovation is provided in [59].

Netflix has labelled their approach to experimentation as “consumer data science” [4]. It is based on a two-step process: experiments are first conducted offline, and if they validate the hypothesis, an online customer experiment is designed and executed to provide definitive validation.

Adobe has adapted the principles of agile software development and the lean startup methodology to fit the needs of a multinational corporation [3]. The resulting “Pipeline” innovation process is consistent with the continuous experimentation approach. It attempts to maximize the learning about a given problem through rapid prototyping and frequent customer evaluation.

eBay is another heavy user of experimentation. Their experimental process is outlined in [18]. It includes a variety of techniques besides online controlled experiments, such as usability testing, focus groups, and diary studies.

The development practices of Intuit are discussed in [11, 12]. The reports relate and analyse the experimental techniques used at Intuit during different phases of product development. They include A/B tests, in-product surveys, and “solution jams” in which solutions to customer pain points are developed rapidly and collaboratively.

Amazon has long utilized a system of continuous online controlled experiments and web analytics to test the value of their R&D ideas. An anecdotal summary of Amazon’s experiences is given in [37]. A similar anecdotal presentation, but from the perspective of a smaller company, is provided by Etsy [42].

In addition to the web-facing domains, the collection and use of product usage data within the embedded systems domain is explored by Holmström Olsson and Bosch [27, 28]. The authors conclude that while such data is habitually collected, an experimental, improvement-oriented approach is often lacking. Finally, examples of successful experimentation experiences in academia-industry collaborations are described in [23, 43].

The abovementioned studies portray different approaches to experimentation. In the context of this thesis, the following criteria are used as requirements for *systematic experimentation*: 1) the business-driven definition of explicit assumptions, 2) the design and conducting of experiments to test those assumptions, 3) the analysis of experiment data, and 4) the use of experiment results as input for decision making and follow-up action. Continuous experimentation is achieved if these steps are a permanent part of the development process.

3 Research questions

This chapter specifies the goals and research questions of the study.

3.1 Goals

As demonstrated by the overview in Chapter 2, there is increasing interest in experimentation as an integral part of software product development. Reflecting this interest, several case studies on companies' experimentation experiences have recently been published. However, the majority of these reports portray the practices of eminent, typically web-facing corporations. A broader understanding of the state of the practice is therefore lacking. There has also been relatively little discussion about the challenges and enabling factors that practitioners associate with continuous experimentation.

This study therefore attempts to achieve two main objectives:

Develop an understanding of the state of the practice of continuous experimentation

Identify the key challenges and success factors
with respect to continuous experimentation

3.2 Research questions

Based on the study goals, two principal research questions are defined. One of the questions is divided into two more precise subquestions. The research questions are:

RQ1: How is continuous experimentation applied in software development companies?

RQ1.1: How is customer feedback concerning the software product collected?

RQ1.2: How is the collected customer feedback used in the software product development process?

RQ2: What challenges and success factors are associated with continuous experimentation?

4 Study design

This chapter discusses the design of the study. It begins by outlining the research strategy and then goes on to describe the research method and the sampling strategy. The specifics of data collection are discussed next. The chapter concludes with an overview of the data analysis method.

4.1 Overall research strategy

The purpose of this study was to explore and portray the experiences and views of industry practitioners with relation to continuous experimentation. The overall design of the study followed the qualitative paradigm. This approach was chosen because qualitative research aims at developing a multifaceted understanding of complex phenomena in their natural context [51]. An understanding of the research topic is built inductively, based on the learning gained from research participants. Moreover, the qualitative research process is typically emergent, taking shape based on initial findings.

The philosophical framework underlying a piece of research influences its design and therefore merits a brief discussion. Robson [51] describes the most common philosophical stances. First, positivism is founded on the perception of verifiable, objective truth. This approach is typical of the natural sciences. Second, constructivism, also known as interpretivism, is based on the notion that truth is a socio-cultural construction and thus context-dependent and subjective. Constructivism characterizes much of qualitative research. However, since the subject matter of this study was more practical than philosophical, a middle ground approach of pragmatism was considered most appropriate. According to Robson, pragmatism approaches truth in terms of what works while acknowledging that truth is relative to the observer.

4.2 Research method

A qualitative survey approach was chosen for this study. Fink [24] identifies several occasions when a qualitative survey design is appropriate, including the following four ones relevant to this study: first, the study is focused on exploring the knowledge and opinions of experts in a particular field. Second, the study intends to collect information in the participants’ own words rather than use predefined response choices. Third, there is not enough prior information of the study subject to enable either the use of standardized measures or the construction of a formal questionnaire. Fourth, sample size is limited due to access or resource constraints.

Methodologically, qualitative surveys resemble the widely used qualitative research methods of multiple case studies [54] and grounded theory [26]. However, both case studies and grounded theory typically rely on multiple methods of data collection, which are used repeatedly during the study. Qualitative surveys do not necessitate this [24, 32], making them a suitable choice for smaller-scale investigations such as the present study. Furthermore, case studies aim at producing an in-depth analysis of particular cases [54], while grounded theory aims at generating an explanatory theory [26]. The focus of qualitative surveys is less specific and theoretical, and more concerned with providing a multifaceted, diverse view of the topic of interest [24, 32]. This made it a suitable choice for the current exploratory study.

The use of the label “qualitative survey” is currently not prevalent, although Jansen [32] argues that many small-scale qualitative studies might well be typified as such. Jansen contends that following the principles of qualitative surveys helps to establish a credible methodological basis to this type of research. Despite the relative novelty of the label, studies using variations of the qualitative survey method can be found even in the field of software engineering, for instance [22, 46, 55].

Finally, qualitative surveys are not to be confused with quantitatively oriented statistical surveys. As Fink ([24] p. 68) summarizes, qualitative surveys forgo statistical representativeness and generalizability to “provide depth and individual meaning to the questions of interest”.

4.3 Sampling strategy

Purposive, nonprobability sampling was used to select study participants, as is common in qualitative surveys [24, 32]. Purposive sampling involves the discretion of the researcher in selecting a sample that is suited to the study’s purposes [51]. Several specific techniques may be used depending on the research goals. Since qualitative surveys are concerned with forming a multifaceted view of the research topic [24, 32], a purposive diversity sample was selected.

The target population of this study consisted of the representatives of software development companies. To achieve a diverse set of participants, companies of various sizes, domains of operation, and stages of life cycle were selected. Companies with proprietary software product development were sought in order to gain an understanding of their particular situation, as opposed to companies specializing exclusively in consultation or customer projects. Furthermore, company representatives from different roles and with solid experience in the software industry were sought to ensure knowledgeable views on the study subjects.

Determining the size of the sample beforehand is often difficult in qualitative studies due to the emergent nature of the research process [51]. Saturation, or the point when the addition of new data provides no substantive new insights with respect to the phenomenon being studied, is sometimes used as an indicator of when to stop collecting data (see for instance [26]). Since formally determining the point of saturation is challenging, a more subjective approach of constantly evaluating the quality and comprehensiveness of the collected data may be used instead [51]. The latter approach was used in this study.

4.4 Data collection method

Interviewing was used for data collection since it is a highly versatile method which can provide illuminating data about the interviewees' opinions and experiences [51]. Interviewing is also highly characteristic of qualitative surveys [24, 32]. A commonly used classification of interview types is based on their level of formality and standardization and distinguishes between structured, semi-structured, and unstructured interviews [51]. The semi-structured approach was chosen for this study because it enabled focusing on predefined research topics while also being highly flexible to allow for unforeseen information. Flexibility in conducting the interviews was necessary due to the exploratory nature of the study. Audiotaping the interviews allows the interviewer to concentrate on the unfolding discussion rather than on note-taking [51]. It also facilitates data analysis by providing a complete record of the interview.

There are certain key disadvantages to using interviews to collect data [51]. First, the elicited information is filtered through the interviewee, and is therefore indirect. It may also be inaccurate or biased. This risk cannot be wholly eliminated, but can be mitigated for instance by following up on responses, using probes and prompts, and emphasizing the non-evaluative nature of the interview. A second intrinsic disadvantage is the resource intensiveness of interviews for both researchers and interviewees.

Individual interviews were selected in lieu of the focus group approach in the present study. Focus groups can provide a rich account of the study subject since the interaction between participants may reveal insights that would not surface in individual interviews [51]. However, emerging group processes can bias the results. On a practical note, focus groups are even more resource intensive than individual interviews. Due to these reasons, focus groups were not employed in this study.

Besides interviews, direct observation and document analysis were viable data collection options for the purposes of this study. Both of these methods are applicable in qualitative surveys [24, 32], and have the potential for deep insights that realistically portray the state of the practice [51]. However, direct observation requires access to the site of interest [51]. It can also be a particularly resource-intensive form of data collection. Another issue in direct observation is reactivity, or the effect of the observer on the situation being observed. These drawbacks resulted in direct observation being rejected as the data collection method of this study.

As regards document analysis, it may unearth detailed information that would not come up in interviews [51]. Its weaknesses include the possibility of restricted access and the difficulty of locating relevant documentation. In this study, publicly available online information was used to gather basic data about the participating companies before the interviews. Analysis of companies' internal documentation was not conducted due to access and resource restrictions.

Semi-structured interviews offered several benefits that made them a suitable data collection method for this study. Nevertheless, as with the use of any other individual method, limitations are inevitable. In order to mitigate these limitations and enhance validity through triangulation, additional data collection methods should be used in possible future instantiations of the study.

4.5 Interview structure

An interview guide containing key topics, questions, and prompts was developed to provide guidelines for the interviews. The interview guide is presented in Appendix 4. The first version of the guide was designed before data collection began and tested in a pilot interview. It was then improved iteratively during subsequent data collection, although the alterations were rather small.

Characteristic of the semi-structured approach, the guide was intended as a thematic backbone of the interviews rather than as a questionnaire-type form [51]. The sequencing and wording of the questions was not fixed, and the time allocated to each topic varied between interviews. Questions were omitted if they did not suit the context of the interview. Furthermore, clarifying questions and other unscheduled questions and observations were permissible for both interviewer and interviewee. This may allow for deeper insights and reduced misunderstandings [51].

Open-ended questions were used throughout the interviews to gain expressive answers in the interviewees' own words [24]. Basic information about the interviewee and his or her company was gathered in the beginning of the interview (Appendix 4, questions 1–3). These questions were meant to "warm up" the interviewee for the main body of the interview [51]. Similarly, there were "cool off" questions at the end to start bringing the interview to a close (questions 21–22). The main body of the interview was centred on three pre-defined themes that were derived from the research questions:

1. Current software development practices (questions 4–6)
   - ➢ Provided background information for both research questions
2. Current practices of customer feedback elicitation and use (questions 7–17)
   - ➢ Sought answers to research question 1 concerning the application of continuous experimentation in software development companies
3. Future practices of customer feedback elicitation and use (questions 18–20)
   - ➢ Sought answers to research question 2 concerning the challenges and success factors of continuous experimentation

4.6 Data analysis method

The data from the semi-structured interviews was examined through thematic (coding) analysis [15, 51]. Thematic analysis was selected since it offers an accessible, flexible, and epistemologically independent approach to the analysis of qualitative data. The analysis was based on an inductive, iterative coding process, during which themes, or patterns, were identified within the data.

As with any other method, there are drawbacks to using thematic analysis. Most importantly, its flexibility requires the researcher to maintain a clear sense of focus and rigor during analysis [51]. On the other hand, this flexibility allows the analytic style to be adapted to suit study purposes.

The guidelines of thematic analysis are similar to those of many other qualitative data analysis methods, such as grounded theory [26]. Thematic analysis is in fact sometimes portrayed as a tool that is used within other data analysis methods to derive insights from qualitative information [14]. However, in this study thematic analysis was used as a stand-alone method as described in [15, 51]. It consists of five phases (Figure 5) which are performed partially in parallel with data collection. The process is therefore evolutionary and iterative: preliminary analysis affects subsequent data collection, and vice versa.

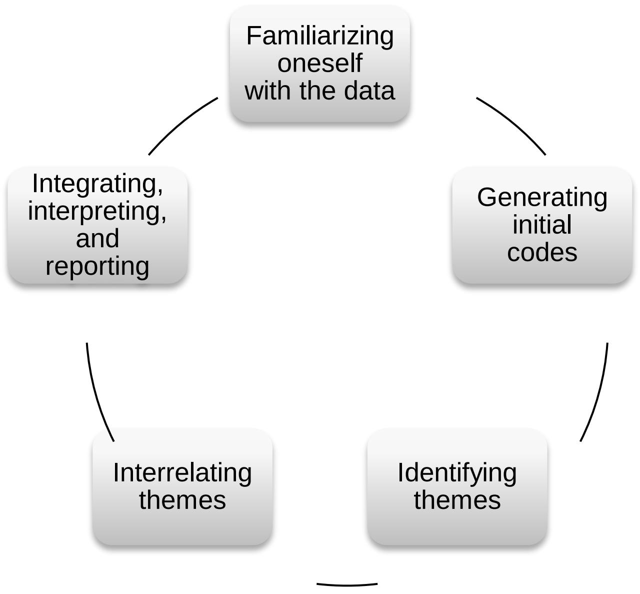
Familiarizing oneself with the data
Generating initial codes
Identifying themes
Interrelating themes
Integrating, interpreting, and reporting

Figure 5. The five phases of thematic analysis. Adapted from [15, 51].

The first phase involves preparing the data for analysis and developing an overall understanding of the data [51]. Transcribing the audio recordings of the interviews helps to achieve both of these objectives. In the present study, transcription guidelines were created to support the study's analytic goals. Transcription was verbatim apart from consecutive word repetitions and non-verbal filler utterances, which were omitted. In addition, conspicuous emphases and emotional expressions such as laughter were indicated, as were other observations considered relevant by the transcriber. Detailed notes on social interaction or the use of language were not included since their analysis was not relevant to the study goals.

In the second phase of thematic analysis, initial codes are assigned to analytically relevant data segments [51]. The codes provide a way to label, organize, and synthesize the contents of the data. An inductive approach, in which the codes were generated based on the data instead of using a predefined coding frame, was considered suitable for the exploratory purposes of the present study. The coding process involved a constant comparison between existing codes and the processing of new data to ensure uniformity. Reflecting this, the study's codebook, presented in Appendix 5, was developed iteratively and incrementally.

Codes may be descriptive, analytic, or category markers depending on the research questions the analysis attempts to answer [51]. In this study, the first research question regarding the use of continuous experimentation generated a large number of descriptive codes (codebook categories "Background information", "Techniques of customer feedback collection", and "Use of collected feedback"). Fewer, more analytic codes were created to answer the second research question regarding the challenges and success factors of continuous experimentation (codebook categories "Challenges" and "Strengths"). Category markers, whose role is to structure the code list, are identified in the codebook by uppercase headings.

Once initial codes have been developed, they are grouped into themes in the third phase of thematic analysis [51]. According to Robson ([51] p.474), a theme "captures something of interest or importance in relation to […] [the] research question(s)". In this study's codebook, the themes are mostly organized hierarchically into top- and sub-level categories. Examples of deriving hierarchical themes from the data are presented in Table 1. The category “Strengths” is not organized hierarchically in the codebook because it contains so few codes.

In the fourth phase of thematic analysis, the relationships between the themes are examined [51]. Finally, in the fifth phase the data is explored within and across themes to summarize the results. When reporting the results, extracts from the data are provided to substantiate findings.

Table 1. Examples of deriving themes from the data.

Original quotation	Code	Sub-category	Top category
“I would say we have good connection to customers and it’s… We know our customers.”[12:29]	Customer and domain knowledge		Strengths
“[T]his helps the customer to prioritize what they should do first and what next. And sometimes even they are able to realize that they might have some development ideas that they realize that they should actually forget about them.”[15:18]	Identifying customer value	Relevance of feedback	Use of collected feedback
“[P]ipelines are pretty much full. So this means if you want to get something new out, it’s going to take at least six months or even more.”[21:3]	Release cycle speed	Product management	Challenges
“[I]t’s more like we pull the feedback from them, not that they push it to us. So people are very reluctant, is that the right word, I don’t know… To give feedback.”[16:33]	Customer organization culture	B2B domain specific	Challenges

5 Study execution

The purpose of this chapter is to specify how the study design, described in Chapter 4, was instantiated in this particular study. The specifics of recruiting participants, conducting interviews and analysing data are presented. Finally, the ethical considerations of the study are discussed.

5.1 Participant recruitment

Study participants were recruited via two channels: among the affiliates of the Need for Speed research program [1] and through the professional contacts of the author of this thesis. Due to practical constraints, only companies operating in Finland were considered. None of the study participants were known to the author beforehand.

Participant recruitment was performed either by the author or by the study’s supervisors. Gatekeepers were contacted at each company, who either participated in the study themselves or suggested a suitable interviewee. A covering letter outlining the purpose and procedures of the study (Appendix 1) was sent via email to the gatekeeper except in the case of the pilot interview.

Once an interview had been agreed, a more detailed study information sheet (Appendix 2) along with an informed consent form (Appendix 3) was sent to the interviewee. In the pilot case, these documents were presented at the beginning of the interview. The purpose of sending the documents in advance was to allow the participants to orientate themselves to the study’s subject area and procedures, and to give them time to voice any queries about the study.

In one case, the recruitment process resulted in four interviewees from a particular company, compared with one interviewee from the other companies. It was decided that the opportunity to gain additional insights should be utilized and all four persons were therefore interviewed. This decision was in accordance with both the study’s exploratory objectives and its purposive sampling strategy.

The recruitment resulted in the interviewing of thirteen representatives of ten software companies. Additional participants were not recruited as no significant new themes had emerged from the latest interviews. This suggested that the existing sample was sufficiently comprehensive for the purposes of this study.

5.2 Study procedures

Thirteen semi-structured individual interviews were conducted in Finland between February and April 2014. A pilot interview was conducted first to test the interview guide (Appendix 4) and to gain an understanding of the duration of the interview. Since the pilot interview revealed that the interview guide was functional and only required minor changes, the pilot interview data was also included in the study results.

The average length of the interviews was 48 minutes, with the range spanning between 36 and 64 minutes. All interviews were conducted in English and recorded with an audio recording device. Eleven interviews were performed face to face on interviewees' company premises, one via video conferencing, and one as a VoIP call. Eleven interviews were conducted by the author of this thesis, while in the remaining two cases the study's primary supervisor was also present to provide expert guidance. One of these two cases was the pilot interview.

The interview recordings were transcribed by the author shortly after the completion of each interview. Clarifications were requested afterwards from the interviewees via email when necessary. The transcripts were coded and analysed using ATLAS.ti [2], a computer-assisted qualitative data analysis software (CAQDAS) tool. Specific analytical tools used in ATLAS.ti included document and code families, code frequency counts, the code co-occurrence explorer, network views, and the memo functionality. The analytic codebook (Appendix 5) was also reviewed by the study's primary supervisor.

5.3 Ethical considerations

Guidelines formulated by Vinson and Singer [64] were followed in ensuring the study's ethicality. The purpose and procedures of the study were shared with the participants via an information sheet (Appendix 2), in addition to which they were asked to give voluntary informed consent to partake in the study (Appendix 3). Permission to record the interviews was requested from the interviewees. This request was stated in the information sheet (Appendix 2) and reiterated verbally in the beginning of each interview. None of the interviewees had objections to the use of audio recording. To promote confidentiality, only persons involved in the execution of this study had access to the audio recordings or their transcripts. Moreover, identifying information was removed from the transcripts before their use in data analysis.

The participants were informed that direct, anonymous quotations from the interviews might be used in this thesis and possible subsequent publications (Appendix 2). Quotation marks and an italicized font mark the interview excerpts. An anonymous quotation number generated by the data analysis tool, ATLAS.ti [2], is used to identify the excerpts. Square brackets within the quotations denote additions and modifications that have been done to improve legibility. Grammar has not been corrected. Ellipses within square brackets ([…]) indicate an omission of a word, a sentence, or a longer text fragment.

6 Results

The objectives of this study were to 1) explore the state of the practice of continuous experimentation, and to 2) identify the challenges and success factors which industry practitioners associate with continuous experimentation. To this end, thirteen semi-structured interviews were conducted with software company representatives. A characterization of the participants is given in Section 6.1.

The main findings from the interviews are presented next. A short synopsis of the participating companies' software development practices is given in Section 6.2, followed by an overview of their customer feedback collection techniques in Section 6.3. A discussion on how customer feedback is used to guide development follows in Section 6.4. Finally, the various challenges and enabling factors associated with continuous experimentation are presented in Sections 6.5 and 6.6, respectively.

6.1 Overview of participants

Ten ICT companies operating in Finland participated in the study. The focus was on their software product development functions. Table 2 gives a characterization of the companies by size, domain, and product orientation (business-to-consumer (B2C) or business-to-business (B2B)). Three of the companies can be described as startups. More details about the companies are not disclosed due to confidentiality reasons.

Most interviewees held either senior management (31%) or middle management (54%) positions in their companies. Consultant and senior software architect roles were also represented (15%). The interviewees' length of employment in their current company varied between 1 and 26 years, with the average being 7.7 years. The interviews were grounded on each interviewee's role within the company. Consequently, the study results do not necessarily represent a comprehensive account of the opinions and ways of working of the participating companies.

Unlike the other companies who only had one representative, company C (Table 2) was represented by four interviewees. Their answers were merged together to form an overall impression of the company. This merging also applies to the analytic code frequency counts in Chapter 6: all code counts are given by company, not by interviewee. As regards company E (Table 2), their practices of software development were not discussed during the interview since the interviewee was not actively involved in this part of the company's operations. Input from company E is therefore only included in Sections 6.5–6.6.

Table 2. Participating companies (size classification: small < 50, medium ≤ 250, large > 250 employees)

Company	Company size by no. of employees	Company domain	Product orientation
A	Small	Gaming	B2C
B	Small	ICT services	B2B
C	Large	ICT services	B2B
D	Small	Sports	B2B, B2C
E	Medium	ICT services	B2B
F	Small	Software development tools	B2B
G	Medium	Software development tools	B2B, B2C
H	Large	Security	B2B, B2C
I	Large	Telecom	B2B
J	Small	Multimedia	B2B

6.2 *Software development practices and principles*

The development practices and principles of the participating companies are summarized in Table 3. The findings are based on the interviewees' informal descriptions of their development approach rather than a formal questionnaire or definition provided by the researchers. Overall, the findings were similar to a recent international survey [63] and a slightly older national survey [52], although the prevalence of lean-inspired practices and continuous integration (CI) appeared to be higher. Consistent with previous research [57], there was variability in how CI was interpreted and implemented in the companies. Details of the companies' CI systems were not examined in this study.

Release cycles were mostly short (Table 3 – the figures concern new product versions, not urgent bug fixes). Furthermore, interviewees often made remarks on constantly having a deployable product version available, working in a production-like environment to simplify deployments, and pursuing a DevOps mode of operation. The overall impression from the interviews was that deployments were quite lightweight and flexible, except for on-premises installations in B2B environments. However, the particularities of the companies' deployment pipelines were not investigated further in this study.

Table 3. Software development practices and principles in participating companies.

Development practice or principle	**No. of companies employing practice**	**Percentage of companies employing practice**
Agile software development	9	100%
Continuous integration	9	100%
Kanban	4	44%
Minimum viable product	4	44%
Scrum	4	44%
Continuous deployment	2	22%
Lean	2	22%
Release cycle ≤ 1 month (incl. continuous deployment)	5	56%
Release cycle ≤ 3 months	3	33%
Release cycle > 3 months	1	11%

6.3 *Customer feedback collection techniques*

The companies used a wide array of techniques to learn about customer needs and evaluate created customer value (Table 4). Most techniques were based on eliciting direct customer feedback through familiar means such as stakeholder interviews and surveys, prototypes, usability and user experience testing, and other forms of user testing. Bug reports and feature voting were also used as a way to guide development. It appeared that customers were generally not very closely involved with the everyday activities of development, for instance in the role of a resident product owner. Overall, the present findings were similar to the techniques mentioned in previous research [11, 27]. As in Section 6.2, these findings are based on the interviewees' informal descriptions.

Table 4. Techniques for customer feedback collection in participating companies. Table structure adapted from [27]. Techniques also mentioned in [11, 27] are italicized to provide a point of comparison.

Development phase	**Development activity**	**Feedback collection technique with no. of companies employing it**	
Pre-development	Exploration and problem definition Requirements engineering	*Customer representative*	1
		Internal brainstorming	2
		Proofs of concept	2
		Prototyping	3
		Stakeholder interviews	5
		Use cases	2
During development	Evaluation and validation	Acceptance testing	1
		Alpha and beta testing	3
		Informal end user tests	2
		Internal experiments for consumer products	2
		Labs website	1
		Pilot customers	4
		Usability or user experience testing	4
Post-deployment	Evolution and maintenance Improvement and innovation	*A/B or multivariate testing*	4
		Bug report analysis	3
		Feature voting	3
		In-product surveys	1
		Product usage data analysis including performance data analysis	5
Continual		*Customer surveys*	6
		Direct customer feedback via email, meetings, phone etc.	7
		Market research	5

Implicit customer feedback in the form of product usage data was collected by a slight majority of the companies (Table 4). In many cases the product instrumentation only covered overall performance data and basic user demographics. However, some companies also had more sophisticated, feature-level instrumentation. Seven companies (78%) had plans either to begin collecting product usage data or to improve current practices in the future. The key motivation behind these plans was the possibility to assess customer value and enable data-driven decision making. Product usage data was considered *"an* *excellent tool […] to see in which features to invest [and] how to improve them […]. And also for […] directly guid[ing] our development efforts." [29:15]* It was also recognized that the enhanced capacity to identify customer value *"ha[s] a positive impact to our business as well, eventually." [28:23]*

However, quantitative product usage data was not regarded as a panacea for the extensive knowledge requirements of software development. Purely quantitative data was considered somewhat one-sided in the sense that *"the data can only tell us we have a problem somewhere, but it cannot tell us what the problem is, or how to fix it." [27:34]* Rigorous data analysis was seen as a necessity for *"figur[ing] out root causes. So it's […] brainwork, […] that's the difficult bit." [21:45]* Qualitative customer feedback can facilitate analysis by illuminating the reasoning behind customer behaviour: *"I think both are […] needed […]. What the users say or think and […] what they actually do, might be also a little bit different in some cases." [22:43]* Overall, interviewees often remarked on the difficulty of measuring created customer value.

Despite the wealth of techniques used to collect customer feedback, their use in systematic, continuous experimentation with customers was rare. Experimentation based on explicit, business-driven assumptions only appeared to be an integral development practice in one (startup) company. Some companies utilized A/B or multivariate testing (Table 4), but most only used it occasionally and not necessarily in a systematic way. Additionally, three companies (33%) had plans to begin using A/B testing or to improve current practices in the future. The unsystematic approach to experimentation was also acknowledged by some of the interviewees:

> *"Whether we are systematic and very good, I have some doubts. It's a little bit ad hoc. So 'Let's have a tagline like this, and maybe like that. Okay, let's put it up there [to production] and let's see'. […] So […] it is not very thorough and not very scientific."[21:30]*

Finally, there was uncertainty regarding the application of A/B testing: some interviewees associated it only with the optimization of existing features, whereas others thought that minor changes do not merit experiments. Moreover, some interviewees thought that A/B testing may be hard to justify to stakeholders if it causes additional R&D expenses.

6.4 Customer feedback in guiding development

The collected customer feedback provides a basis for guiding product development to better meet customer needs. Findings on how the feedback was integrated into the product development process are described in the first section. The following section considers the connection between customer feedback and business strategy.

6.4.1 Integration into product development

Based on the conducted interviews it appeared that customer feedback was integrated into the development process in fairly traditional ways. The feedback was analysed to extract work items which were then organized into a prioritized product backlog. In a business-to-business environment these phases were typically underpinned by an ongoing dialogue with the customer organization(s).

There was some variation in how the interviewees described their approach to customer feedback processing. Particularly the startup representatives emphasized an innovation-oriented approach: *"If we only follow the explicit customer requirements, we don't actually do anything innovative, we just fill the need that they currently have." [16:36]* Exploring the feedback beyond face value in order to generate new ideas was an essential factor in this approach: *"The interesting thing is their complaint, not the solution that they are providing." [14:37]*

The level of involvement of different stakeholders in analysing customer feedback varied: in some cases, product or project managers (or equivalent roles) and the development team were all heavily involved with analysing the feedback and the responsibility was shared. In other cases, management roles had the principal responsibility for the process but all the feedback was reviewed together with the development team. Finally, particularly in the larger companies, the process was management-led and the development team mainly based their work on a ready-made product backlog. Some interviewees considered this problematic: *"[T]here is still a lot for improvement in that area [sharing customer information with the development team]." [20:12]* The problems arose from a possibility for lost insights: *"[I]t's very, very important [...] to spread all this [customer] information even to the developers, because they have typically great ideas." [20:13]*

6.4.2 Connection to business strategy

Two divergent approaches emerged regarding the influence of customer feedback on business strategy and goals. First, some company representatives thought the strategy was continuously being revised based on the feedback. This approach was predominant among the startup companies. As one interviewee said: *"Our strategy is to experiment. [...] At the moment [...] we are making it up as we go along and we see, 'Hey, that's getting traction, let's do more of that'. Rather than a big strategy deck." [27:54]*

It is noteworthy that employing a flexible strategy did not imply an unclear product vision. Several interviewees remarked that new task candidates were habitually evaluated against the product vision: *"[I]f it's something that is not in the roadmap, then we evaluate [...] does this fit into our [...] current vision of the product." [16:56]*

In the second approach, business strategy and goals were considered more stable and not directly influenced by the customer feedback: *"[O]f course everything is connected, but it [the customer feedback] doesn't direct our business goals." [12:42]* In this approach the emphasis was on the strategy guiding the development activities rather than the other way around: *"[T]here is a direct link [to] what we are doing from our strategy." [28:59]* This approach appeared to be more typical to established companies.

6.5 Challenges with respect to continuous experimentation

Gaining an understanding of the obstacles that practitioners associate with experimentation was a central objective of this study. This section begins by presenting the key domain-independent challenges, classified into four broad topic areas that emerged from the interview data. Figure 6 gives an overview of these challenges. Additionally, challenges specific to the business-to-business domain were identified. The section concludes with a review of these factors.

6.5.1 Organizational culture

For the purposes of this study, organizational culture was broadly defined as "the way we do things around here" ([13] p. 22). A range of issues was categorized under the concept. These included issues related to overall beliefs and perceptions, ways of working, and roles and responsibilities.

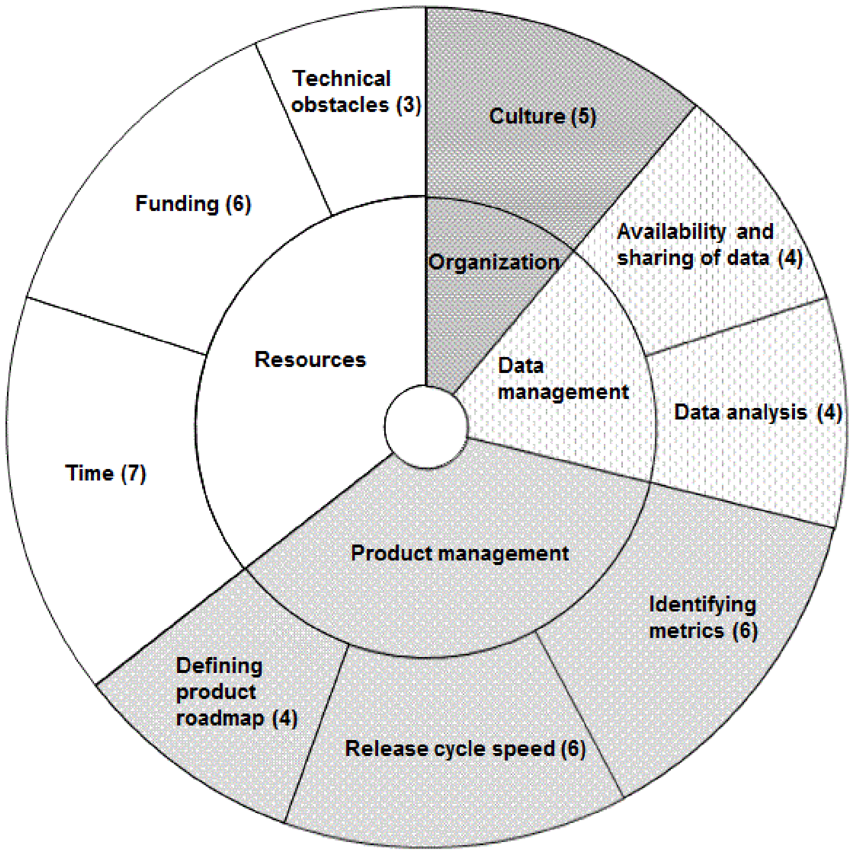

Figure 6. The key domain-independent challenges with frequency of occurrence by participating company (outer circle) sorted by topic areas (inner circle).

Half of the company representatives considered **organizational culture** a major obstacle to moving towards an experimental mode of operation (Figure 6). The overarching issues with relation to organizational culture included a perceived lack of agility, proactivity, and transparency – either within the company or in relation to the company's customers. It was noted that *"the technical things are not [...] even close to the weight of the cultures' obstacles." [29:22]* Another interviewee agreed that trouble in embracing experimentation *"has nothing to do with technology" [18:17]* and that moving towards an experimentation culture was the principal challenge.

Transforming the culture of established companies with a long history in traditional, big design up front software development was considered challenging:

> *"I think the biggest obstacle [...] is that the middle management of every companies is fine-tuned to do industrial production. But now we're working with brains, not anymore [...] [with] the pieces of a factory." [18:17]*

Accepting a degree of initial uncertainty was considered essential when shifting from heavy upfront planning to an experimental mode of operation, *"[a]nd this thinking is very hard, and this has to do with the culture of the companies. That they think [about the] business potential for [...] the unknown things, and they test it out." [18:13]*

In addition to the aforementioned higher-level challenges, some more specific issues regarding roles and responsibilities were identified. First, some interviewees observed that the development team was not always sufficiently involved in customer interaction and customer feedback analysis, and valuable insights may therefore be lost. The interviewees hoped to be able to better utilize team input in the future. Second, there were a few comments on different roles having divergent viewpoints to product development: *"[T]echnological people [...] love to tinker with details. Or with something that doesn't sell anything." [14:23]* Third, the importance of allocating responsibilities and sufficient resources for managing them was noted:

> *"[T]here should be at least one person within the organization whose main concern is [...] to be the one [...] who understands what our customers need. [...] [U]sually [...] it's nobody's responsibility, and nobody has time for that." [15:27]*

The overall impression from the interviews was that cherishing and improving customer understanding was a central value to all of the participating companies. However, one interviewee questioned the depth of commitment to customer value in the sense that companies may *"do customer listening [because] everybody has to do it, but then, in the end, the message really never comes through." [15:15]* For instance, there may not be enough resources for executing change or improvement initiatives in a short-term-oriented atmosphere where "*saving money now seems to be the main issue, and not really how to attract the customers more." [15:24]*

Cultural challenges were remarked upon by the representatives of both established and startup companies. However, the general impression was that there was some variation in the magnitude of the experienced challenges: the startup representatives appeared to identify individual, specific issues, while the more fundamental issues were typically brought up by the representatives of established companies.

6.5.2 Product management

Concern over slow **release cycles** was one of the central themes in terms of product management challenges (Figure 6). Despite the already relatively short release cycles of the companies (Table 3), the wish to further accelerate development was evident in the interviewees' comments. Some companies considered this their principal target for development: *"[W]e are experiencing a [...] slowing down of the development process, and that is one of the biggest challenges that we are experiencing today. I would say that the biggest challenge." [18:31]*

Some company representatives recognized clear reasons for the suboptimal functioning of the development cycle. A surplus of work compared to R&D capacity was a significant factor: *"[P]ipelines are pretty much full [...] [I]t takes so long to get [a new requirement] through the system." [21:3]* Bottlenecks in the development process were another reason: *"[T]he customer requirements, they quite often lie too long in that [product] backlog. Nothing is done." [12:38]*

Focusing on products and features that create most customer value was seen as a central way to speed up development. As one interviewee summarized: *"I don't think you can accelerate anything. What you can do is do less. [...] So pushing back on the need for more would be the way to speed up things." [21:32]*

Identifying the metrics to evaluate created customer value and product success was a major challenge both in relation to dedicated experiments and to the general observation of product usage (Figure 6). In the words of one interviewee:

> *"To measure the right thing is the hard thing, to know that what is relevant. I think you can easily measure such a lot of things that you [...] lose sight of the forest for all the trees. And then you just optimize irrelevant things."[27:22]*

This sentiment was shared by another interviewee: *"We have [a] huge amount of data, but we need to use it wisely. It's no sense to make metrics without rationality." [13:22]* Similarly, the challenge of defining metrics to suit the characteristics of different products was noted: *"We measure the wrong things for some of [the] products." [21:44]*

A particularly interesting challenge from the point of view of experimentation related to which metrics and techniques of customer feedback collection to use when scaling up a product. After a certain point in growth the company could not *“rely on individual customer comments too much, we really need to get the much bigger sample from the [...] [customers] that what is important and what is not important.” [28:45]* However, finding a feedback collection technique to suit both the nature of the product and its customer base was considered demanding:

> *“You can’t throw big data analytics on this [product] with a few thousand people, but you can’t really [...] interview each [...] one of them [...] either. And this is exactly the spot where you yearn to move upwards, and we don’t know what’s [...] the obstacle to move into the hundred thousand or million downloads.” [21:52]*

A further set of issues was related to **defining the product roadmap** (Figure 6). Identifying a minimum viable product (MVP) was considered *“very easy to say, very hard to do. And the main focus shouldn’t be minimum [...], but what is viable, that’s the real thing.” [21:34]* Another interviewee contemplated the challenges of the MVP approach with respect to scheduling experiments:

> *“One big thing is always how far to take the product before we measure it. [...] [W]hat is the minimum viable product? If I get crap metrics, does that mean that my product idea is crap or just that my implementation is crap?” [27:13]*

As regards established products, one interviewee described formulating a product backlog as *“black magic” [21:61]* as it could be so difficult to organize and prioritize varying customer needs. Another closely related challenge was the alignment of customer needs with the company’s internal product vision and strategy.

6.5.3 Data management

Interviewees frequently expressed concern over **deficiencies in the analysis** of collected customer feedback and other data (Figure 6). Learning potential was lost because the data was not rigorously analysed and integrated into the product development process: *“There’s too little analysis of available data, we should actually utilize [...] the existing data more in our decision making so that the element of gut feeling or some kind of intuition would be minimized.” [28:27]*

Lack of time emerged as a key reason for inadequate data analysis. In addition, insufficient analytic expertise was mentioned as a contributing factor. The extract below describes these challenges:

> *"[T]here are not so many people who really are skilled enough to understand the numbers [produced e.g. by web analytics], or they don't have enough time to really go deeper into the numbers and figure out what is this all about and what does this mean to us." [15:19]*

Learning potential was also reduced by obstacles encountered in the **availability and sharing of data** with relevant stakeholders (Figure 6). In some cases the data was available in principle, but it was hard to find or decipher (especially in the case of quantitative data). The challenge of sharing and managing tacit knowledge was also remarked upon. One interviewee described their challenges with sharing data as follows:

> *"The data is scattered all over the place [...]. If you need specific data that you don't have, you would need to know the guy who has the data and kindly ask him. [...] [W]e are quite far from providing [a] really convenient, broad spectrum of data to all of the employees." [21:15]*

6.5.4 Resources

Lack of time and funding were among the most often mentioned challenges to experimentation (Figure 6): *"[T]he limiting thing is the money. [...] Money and time." [12:28]* Investing in building an experiment system had to be considered in relation to other improvement initiatives. On the other hand, some interviewees emphasized the potential long-term benefits of investing in experimentation: *"[T]here's no strong enough belief on the fact that actually from customer satisfaction you would get better results in a long term." [15:14]*

Technical obstacles to experimentation barely featured in the interviewees' commentaries; only three cases emerged in which technical concerns restricted experimentation or had done so in the past (Figure 6). Moreover, these concerns appeared to be primarily linked to the resource demands of experimentation rather than insurmountable technical problems. In the words of one interviewee: *"[O]f course, there are tools [to support experimentation], the implementation is then [...] maybe the [...] harder thing. [...][I]t always requires that you do some implementation to your product." [16:50]*

6.5.5 Business-to-business-specific challenges

Figure 7 presents the key business-to-business-specific challenges that emerged from the interviews. The supplier company's experimentation approach must take into consideration the cultures, processes, and resources of its customer organizations. In many cases, aspects of the **customer organizations' culture** presented a challenge with relation to experimentation. For instance, despite the prevalent use of agile methodologies, customers were not always able to participate in the development process or in dedicated experiments. The following excerpt describes how this may limit the potential for learning about customers and also affect the mood of the development team:

> *"[U]sually they [the customers] are not interested [in] how the development is going at all. We tried to invite them to the development sprint demos, but no one came. So they are buying the service, not development. And it was disappointing for the developers, because they was asking [for] the customer feedback and then they realized that 'Okay, they are not caring about us at all.'" [13:27]*

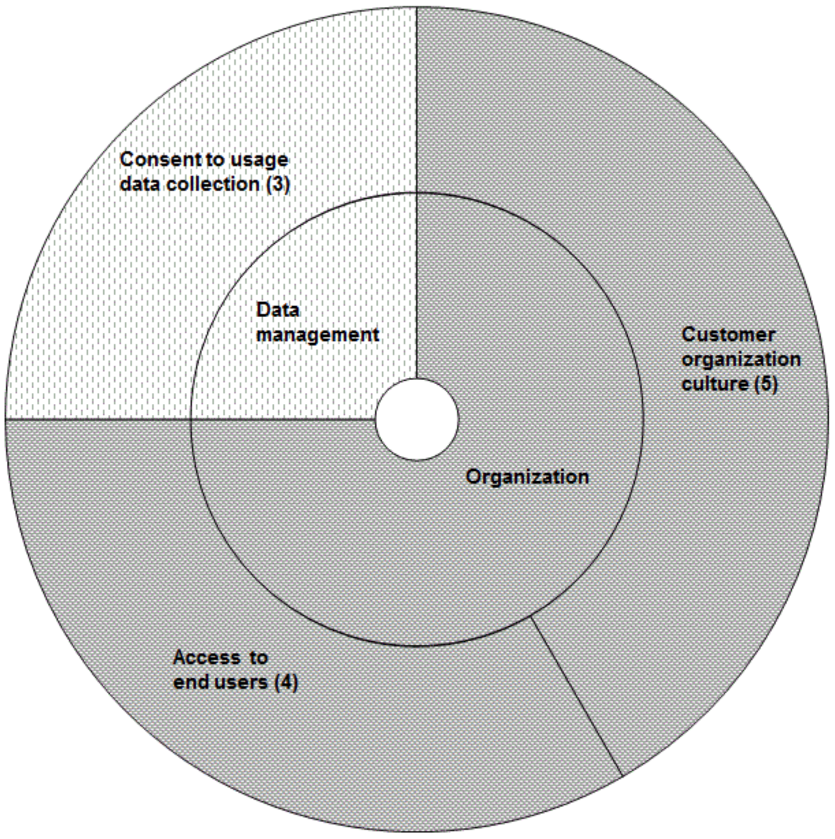

Figure 7. The key business-to-business-specific challenges with frequency of occurrence by participating company (outer circle) sorted by topic areas (inner circle).

Another interviewee also commented on the difficulty of obtaining explicit customer feedback: *“[I]t’s more like we pull the feedback from them, not that they push it to us. So people are very reluctant [...] [t]o give feedback.” [16:33]* Cultural issues also affected the possibility of running dedicated customer experiments, especially when using on-premises installations: *“[I]t’s very hard to do that [A/B testing] in the on-premises environment, because usually they are very strict on what they want to release on their [environment].” [16:49]* Furthermore, if the product was developed in customer-funded projects, customers could be unwilling to pay for experimentation if it was seen as an additional expense that did not generate direct customer value.

Lack of time emerged as the primary suggested reason for customers’ disinclination to participate and contribute more:

> *“[T]ypically at customer side, there are much less people involved [...], and [the] same people are involved in several deliveries with several suppliers. And if [...] they need to give feedback and share their experiences with all the suppliers, it’s time-consuming.” [19:24]*

A related challenge was that **access to the end users** of the products was often limited: *“[Y]ou actually can’t find the end user. You can only find some managers above him.” [16:44]* Collecting relevant customer feedback without involving the end users was considered difficult because *“the people who make the decision, [...][are] not the end user[s], so they don’t actually see the problems, or [...] if they see the problems, the problems might [...] be totally different.” [16:23]*

Some interviewees regarded that improving product usage data collection would provide a useful way to alleviate many of the aforementioned challenges. Sophisticated product instrumentation would automatically provide the supplier company with genuine end user feedback about the product. However, **consent to usage data collection** could not be taken for granted since *“it might be difficult to get some of the customers to agree that we can monitor their users and what they do.” [22:28]* Naturally, cases exist where product usage observation by external parties is not possible for instance due to strict confidentiality and safety regulations.

6.6 Success factors with respect to continuous experimentation

Another objective of this study was to gain insights into the success factors of experimentation. Interviewees were invited to reflect on their companies' strengths with respect to customer involvement and their methods of collecting and utilizing customer feedback. Figure 8 gives an overview of the key domain-independent success factors, using the same topic area classification as Section 6.5. The results are described in more detail in the following subsections.

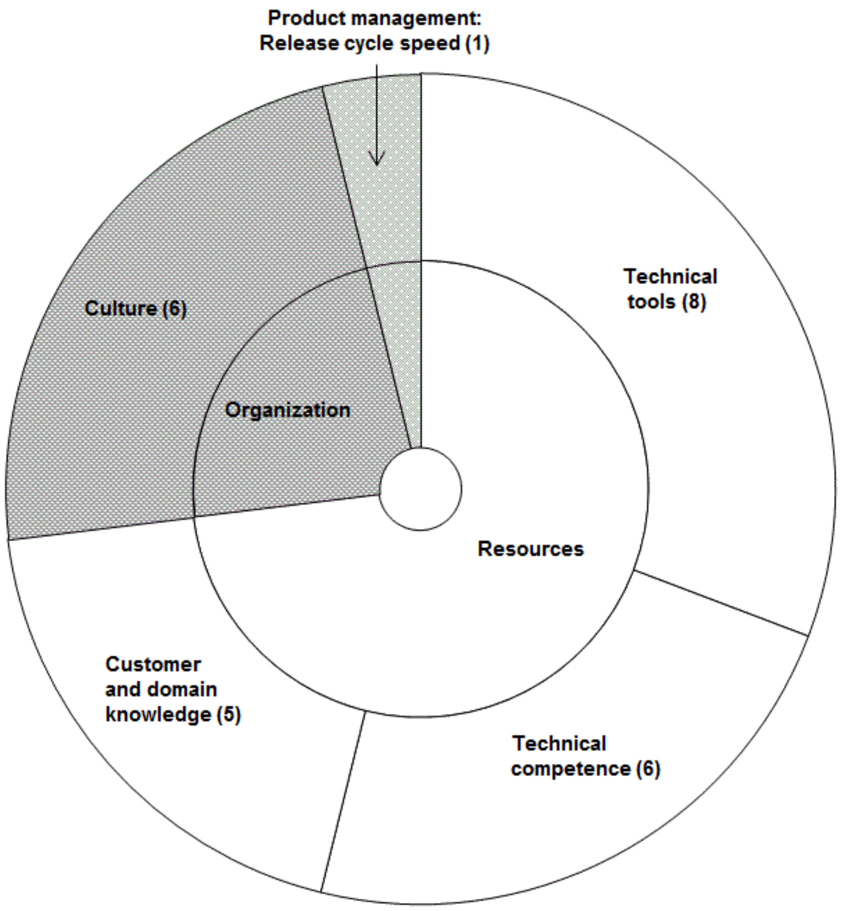

Figure 8. The key domain-independent success factors with frequency of occurrence by participating company (outer circle) sorted by topic areas (inner circle).

6.6.1 Organizational culture

A positive **organizational culture** was a recurrent theme in the context of success factors (Figure 8). The positive evaluations were typically made by the representatives of small companies, in particular the startup representatives. One interviewee specified his company's key strength with respect to experimentation: *"That we do it! That we un-*

derstand that we should do it." [27:45] He described their approach as follows:

> *"I think we have a good company culture in that everyone understands the value of this thing [experimentation]. [...] [O]f course people will do more or less of it [experiment design and analysis]. [...] But still, to some degree, we are all looking at 'Okay, the data is going that way.'" [27:15]*

Other interviewees also stressed the importance of cultivating an open culture in which the whole team is involved in discussing the direction of product development and *"everybody are free to tell what they think." [14:39]* In addition, the proper alignment of employees' authorities and responsibilities was mentioned. An empowered organizational culture was considered desirable: *"[W]e try to be empowering our every employee as much as possible and give them the freedom where they work, what they work with." [16:46]*

One interviewee reflected on the common nominators of established companies who have succeeded in transforming their culture to an experimental mindset: *"There are people who are rebellions. They don't obey this traditional way of working. They are strong enough to do something different." [18:19]* He also stressed that successful companies are organized in a way that supports and respects the grassroots-level of the company, because *"innovations [...] happen [...] between us and our customers, [...] in that dialogue. [...] [I]t doesn't happen on [the] managerial level, nor up in the leadership teams [...]. It happens in the frontline of our company." [18:23]* This viewpoint parallels the aforementioned findings regarding the active involvement and empowerment of the product development team.

Finally, practical support functions were a concrete reflection of a positive approach to experimentation. The support included assistance in designing experiments and experiment artefacts, scheduled data analysis meetings, and the use of sophisticated development and test environments. The objective of these support functions was that individual team members *"don't [...] start from scratch, we have guys to help in that. Which [...] is a huge asset." [21:41]*

6.6.2 Product management

Achieving a rapid release cycle was one of the key challenges of experimentation (Section 6.5.2). However, **release cycle speed** was also regarded as a success factor by one company representative (Figure 8). The following quotation demonstrates that a faster cycle speed compared to competitors provided a competitive advantage:

> *"We can develop the service based on the customer needs, which is, of course, the [way] [the competitor] is doing it, but they have a huge amount of customers and it might be unlikely to get a new feature in during [...] one year. We can do it in three months." [13:26]*

6.6.3 Resources

The good **availability of technical tools** and the perception of **technical competence** were among the most often recognized success factor (Figure 8). Interviewees considered that the tools exist for implementing and improving experimentation, even if they were not all actively used due to the challenges discussed in the previous section: *"We have the tools. [...] We tried it [a particular experimentation tool] [...] once and it looked good." [12:18]* Interviewees also appeared to trust the technical capabilities of their companies: *"[I]t feels like we can get the technical issues sorted out." [27:18]*

The importance of extensive **customer and domain knowledge** was another frequent theme in the interviews (Figure 8). As one interviewee said: *"I would say we have good connection to customers [...]. We know our customers." [12:29]* In some cases, the knowledge had been acquired organically during a long shared history. In other cases, various measures had been taken to actively involve customers in product development. Besides knowing one's customers, expertise in the domain of operation was important: *"People understand [...] what it's about." [14:21]*

7 Discussion

The findings of the present study are discussed in this chapter. The chapter is divided into three sections. Section 7.1 examines the state of the practice of continuous experimentation. Section 7.2 explores the key challenges and success factors of building an experiment system. Finally, threats to the study's validity are considered in section 7.3.

7.1 State of the practice of continuous experimentation

The first goal of this study was to develop an understanding of the state of the practice of continuous experimentation. Insights into how continuous experimentation is applied in software development companies were sought by exploring 1) software development practices and principles, 2) techniques of customer feedback collection, and 3) how the feedback is utilized in the software product development process.

The study found that the principles of continuous experimentation resonated well within the software industry: there was a general wish to focus on customer value creation and data-driven decision making. Many of the contributing companies' current practices supported these aspirations: agile development had supplanted traditional, waterfall-like development, continuous integration was utilized, and release cycles were reasonably short even though the use of continuous deployment was rare. Companies were attempting to further shorten release cycles for instance by focusing on key functionalities – a goal which experimentation may help to achieve. These practices suggest that the importance of a rapid customer feedback loop is widely acknowledged in the industry.

The contributing companies collected a wide range of direct customer feedback, but the collection of implicit customer feedback in the form of product usage data was not yet prevalent. Moreover, the rudimentary level of product instrumentation often impeded the use of product data to gain deeper insights into customer behaviour. However, the potential in product usage data had been acknowledged and most companies had plans to develop their procedures in this respect. These findings are in line with [27, 28], suggesting that there is untapped learning potential in product usage data.

The study found experimentation to be systematic and continuous in only one startup company. In addition, several companies expressed interest in A/B testing. This suggests that many practitioners are aware of the possible benefits of embracing an experimental approach to software development. It is also noteworthy that besides controlled experiments, a wide array of customer feedback collection techniques can be used systematically (for examples, see [11, 27]). A related minor finding was the practitioners' uncertainty concerning the purpose of A/B testing. This uncertainty may reflect the relative novelty of formal experimentation techniques among the general ICT community.

The study found that the primary use for the collected customer feedback was the extraction and prioritization of work items. There was variation in how much the development team participated in customer interaction and customer feedback analysis: generally less in larger companies and more in smaller companies, particularly the startups. While the natural effects of company size and different business cases are potential explanations for this, it may also indicate differences in organizational culture. The role of organizational culture in experimentation is discussed in Section 7.2.

The connection between product vision, business strategy, and technological product development is central to continuous experimentation [23] and business alignment [5]. Experiments integrate these aspects by providing empirical data to support both product-level and strategic decision making. This study found a highly flexible approach to business strategy management to only be typical of startups. As Fagerholm et al. [23] note, the continuous experimentation model is derived from a startup environment, and different variants of the model may be required to support other scenarios, possibly in a domain-specific manner.

Innovation is an essential factor in a well-functioning experiment system [11]. The significant connection between experimentation and innovation is discussed in more detail for instance by Thomke [61]. Previous studies have found deficiencies in the use of product usage data as a basis for product improvements and innovations [27, 28]. This study considered a broader scope of customer feedback, but the previous finding was echoed in the sense that the connection between innovation and customer and product data did not arise very strongly. Furthermore, the results suggest that innovation potential may be lost if the collaboration between the R&D organization, product management, and customers is insufficient. The abovementioned observation about the limited involvement of the development team in customer interaction and customer feedback analysis is one such example.

This section has discussed the state of the practice of continuous experimentation. It was shown that although the majority of companies have not yet reached the stage of continuous experimentation, many appear to be proceeding towards it as outlined by the “Stairway to Heaven” model [29]. However, there are several challenges along this path, as well as certain enabling factors. These are discussed in the next section.

7.2 Key challenges and success factors

The second goal of this study was to identify the key challenges and success factors which industry practitioners face with relation to continuous experimentation. Noticeably fewer success factors than challenges were uncovered during the interviews. The reasons for this may include that it is easier for study participants to recognise currently problematic factors than something that is already functioning well within their organization, especially in a time-limited interview. Nevertheless, the identified challenges and success factors mirror each other to a certain extent (see Figures 6–8), and hence they are discussed together in this section.

The most significant finding of the current study was that most of the major obstacles to continuous experimentation related to such wide-ranging issues as organizational culture, product management, and resourcing. A possible explanation for this might be that out of the companies in this study, only one appeared to have a highly mature system of continuous experimentation in place. For the remaining companies these wide-ranging issues still await attention before the focus can be moved to the intricacies of running experiments. Furthermore, many of the interviewees had managerial roles in which a broader, less technical perspective to software development is perhaps customary. Nevertheless, the main implication of this finding is that an organization-wide perspective is required when attempting to move towards continuous experimentation.

As mentioned, the broad influence of organizational culture on experimentation was a recurrent theme in the study. Concerns over inadequate agility, proactivity, transparency, and tolerance for uncertainty were expressed by the interviewees. Collaboration challenges between business stakeholders, discussed in Section 7.1, also relate to the issue. Due to these deficiencies, the customer feedback loop may not function optimally.

Based on the findings of this study, organizational culture appears to be more supportive of experimentation in startup companies. Fewer fundamental problems concerning the organizational culture were mentioned and more positive remarks were made with relation to engaging and empowering the whole team. This is perhaps natural as it may be easier to cultivate a flexible, empowered atmosphere in a small, fledgling company.

An experiment-driven approach to software development is still a relatively novel approach [11], and companies have consequently had little time to transform their culture and practices accordingly. On the other hand, agile development is a well-established practice, but organizational culture is still cited as the key barrier to further agile adoption, as well as a leading cause of failed agile projects [63]. Similarly, the present study indicates that in many cases, further efforts are required to promote an experimental organizational culture and assimilate agile principles throughout the whole company, not just the R&D organization.

Previous case studies on the experimentation practices of established companies have also noted the role of organizational culture. In the case of Adobe's “Pipeline” innovation process, the main challenge was to obtain upper management support [3]. The support was secured by demonstrating how the experiments helped to avoid wasting R&D resources on valueless features or products. Kohavi et al. [34] describe measures taken at Microsoft to promote an experimental culture. They include various formal and informal procedures designed to educate personnel about experimentation and raise awareness of its possibilities. The authors note that achieving cultural change is a gradual, continuous process.

Other measures to promote experimentation include using small, empowered product development teams with the necessary knowledge and resources for rapid experimentation. This approach is recommended for instance by Thomke [61]. He also considers it essential that companies embrace a flexible “fail early and often”-mindset to drive innovation. A similar “test-and-learn” mentality is promoted by Davenport [18]. Finally, it is important to align companies’ employee evaluation and rewarding systems to support experimentation and reduce fear of failure [39].

The present study found release cycle speed to be a highly recurrent theme with respect to product management: most companies were attempting to achieve a faster release pace. This is, of course, a central goal in agile development [6] and likewise important in continuous experimentation [23]. However, it should be noted that pushing for shorter release cycles may have unintended implications: it may, for instance, result in fewer bugs getting fixed [33]. The viewpoint emerged from this study that rather than simply attempting to do things faster, the focus should be on prioritizing work according to customer value. This may help speed up development without jeopardizing software quality or development team morale.

In order to measure customer value, appropriate product metrics are needed. Although the results of this study indicate that practitioners have acknowledged the need for clearly defined metrics, their identification was a challenge. The lean startup methodology concentrates on actionable metrics, and examples of defining software product metrics are presented for instance in [17, 50]. On a related note, Bosch et al. [10] propose the Early Stage Software Startup Development Model (ESSSDM) as an extension to lean startup practices. ESSSDM provides operational guidelines on identifying, validating, and scaling product ideas. On a more general level, the GQM+Strategies method [5] provides steps for connecting organizational goals and strategies with software measurement, helping to align the whole company towards customer value creation.

In terms of the other findings of the study, shortage of time and funding were some of the most often identified obstacles to experimentation. This is not entirely surprising since limited resources are likely par for the course in most modern companies. The question should perhaps be how to take experimentation into account in the division of resources. An interesting viewpoint on the matter arose from the study, stressing the value of experimentation as a strategic investment: it aims to boost customer satisfaction and thus ensure the viability of the company in the long term.

The purely technical aspects of experimentation were rarely regarded as obstacles in this study; in fact, the availability of technical tools and competence were repeatedly identified as success factors. This did not imply that the companies already had the technical capacity in place for experimentation, but rather that given the resources, practitioners felt that the technical implementation was not a problem. However, as the participants of this study predominantly represented managerial roles, this finding must be interpreted with caution: it may be that technical personnel would have divergent opinions. Nevertheless, the overall impression from this study suggests that technology has a supporting role in an experiment system, and that the more significant issues lie elsewhere.

Finally, the current study found that operating in a business-to-business (B2B) market signified additional challenges with respect to experimentation. Fagerholm et al. [23] note that continuous experimentation in the production environment may indeed be more suitable for business-to-consumer companies. A dedicated test environment might be needed in B2B scenarios, to which experimental product versions can be continuously deployed without compromising production operations. The authors also suggest using early-access and beta versions to collect end user feedback in cases where consent to product usage data collection in the production environment is unavailable.

However, customers would still need to have the inclination and the resources to take part in experiments and product evaluation. The findings of this study suggest that this may prove to be challenging. Furthermore, it may be difficult to obtain a large enough sample to provide meaningful answers to controlled experiments [36] in test environments. To summarize, building an experimentation system in B2B scenarios necessitates an active dialogue between all involved parties. It appears that the value of investing and participating in experimentation should be better demonstrated to customer organizations. Small-scale experiments with easily identifiable gains might bring about positive experiences in this respect.

This section has provided a discussion of the key challenges and success factors of continuous experimentation. It has been argued that a broad perspective is needed when attempting to promote continuous experimentation: conducting the actual experiments is merely the tip of the iceberg. Measures that support experimentation should be taken throughout the organization, beginning with an examination of the organizational culture. Adequate resources should be allocated for these development initiatives, and particular attention should be paid to improving collaboration between stakeholders. The specific challenges of experimentation in a B2B environment should also be considered. However, caution must be applied when considering these findings, as the present study is subject to certain validity threats. These are reflected upon in the next section.

7.3 Threats to validity

In accordance with Easterbrook et al. [21], four commonly used criteria for validity are discussed below in the context of this study. Although these criteria have been rejected by some as unfitting to qualitative enquiries due to their positivist roots (e.g. [40]), others see their use as a way to affirm the scientific value of qualitative research (e.g. [51]).

Construct validity focuses on whether the concepts being studied have been properly defined and interpreted [21]. In this study, construct validity was mainly threatened by potential misunderstandings and ambiguities between researchers and interviewees. To diminish this risk, the overall goals of the study and the central concept of continuous experimentation were shared with participants prior to the interviews. Furthermore, the use of semi-structured interviews enabled the asking of clarifying questions for all involved parties. This also helped to counteract the possibility of interviewee bias. Clarifications were also requested afterwards from the interviewees when necessary. Moreover, the recording and verbatim transcribing of the interviews helped to increase the transparency, accuracy and level of detail of data collection and analysis.

External validity is concerned with the generalizability of the results [21]. Qualitative surveys do not aim at producing statistically representative results: rather, their goal is to provide a multifaceted account of the study subject [24, 32]. However, despite the limited scope of this study, care was taken to include a variety of companies represented by interviewees from different roles. The results are therefore considered to be well grounded in actual practice.

Reliability focuses on whether the results are independent of the researchers and therefore replicable [21]. Steps taken to improve reliability included the development of the interview guide and the analytic codebook. These artefacts were also reviewed by the study's primary supervisor. Additionally, the study procedures were detailed in this thesis.

Finally, internal validity is mainly concerned with the reliability of claimed causal relationships [21]. This threat was not highly relevant to the present, mainly descriptive study.

This section has reviewed the key threats to the study's validity. Steps were taken to enhance validity when possible in light of practical constraints. However, the limited scope of the study does not guarantee representativeness, in addition to which a possibility for researcher and interviewee bias remains. The study results therefore need to be interpreted with caution, and a follow-up study should be conducted to validate and extend the results.

8 Summary

This chapter gives a synopsis of the study’s main findings and presents an outlook on future research prospects.

8.1 Summary

The purpose of this study was to develop an understanding of the state of the practice of continuous experimentation and to identify the main challenges and success factors associated with it. To this end, a qualitative survey was conducted based on interviews with selected industry practitioners.

The study found that while many of the current development practices supported experimentation, the state of the practice was not yet mature. Although a broad array of techniques was employed to collect customer feedback, systematic experiments with customers were rare. Moreover, many companies did not use product usage data to learn about customer needs, and product instrumentation was often inadequate. Finally, the collaboration between the R&D organization, product management, and customers sometimes appeared insufficient for supporting an innovative, experimental approach.

Key challenges in embracing experimentation related to transforming organizational culture, achieving sufficiently rapid release cycles, identifying metrics for evaluating customer value and product success, and ensuring that the collected customer and product data was carefully analysed by relevant stakeholders. Adequate resources also needed to be secured. Additional challenges were faced by business-to-business companies.

Conversely, the good availability of technical tools and competence was found to facilitate experimentation. Supportive organizational culture along with in-depth customer and domain knowledge were additional important success factors.

8.2 Future research

The findings of the present study suggest that concrete tools are needed in the industry for embracing, designing, and managing experimentation. Specific questions include how to design useful and systematic experiments, how to identify appropriate metrics, and how to ensure proper analysis of experiment data. A further set of questions relates to aligning experiments with business and product strategies, and on identifying and prioritizing the business assumptions to be tested.

This study revealed more of the challenges of continuous experimentation than its success factors. Further research into the success factors would therefore help to validate and extend the present findings. Utilizing a different data collection method, such as direct observation, might provide a different viewpoint to the matter.

The results of this study indicate that operating in a business-to-business environment may complicate the application of continuous experimentation. Further investigation into the special characteristics of experimentation in this domain would consequently be interesting. On a related note, the experiences of customer organizations should be explored to understand their viewpoint on the matter.

Finally, most of the industry practitioners interviewed in this study held management positions in their respective organizations. Further exploration of the viewpoints of other roles such as software architects, developers, and quality assurance personnel would help broaden the understanding of software development as an experiment system.

References

[1] Need for Speed research program (N4S). http://www.n4s.fi [2014/12/7].

[2] ATLAS.ti Scientific Software Development GmbH. http://www.atlasti.com [2014/12/7].

[3] Adams, R.J., Evans, B. and Brandt, J. Creating Small Products at a Big Company: Adobe's Pipeline Innovation Process. In: CHI '13 Extended Abstracts on Human Factors in Computing Systems, pp. 2331-2332. ACM, New York, 2013.

[4] Amatriain, X. Beyond Data: From User Information to Business Value Through Personalized Recommendations and Consumer Science. In: Proceedings of the 22nd ACM International Conference on Information and Knowledge Management, pp. 2201-2208. ACM, New York, 2013.

[5] Basili, V. et al. GQM+Strategies: A Comprehensive Methodology for Aligning Business Strategies with Software Measurement. In: Proceedings of the DASMA Software Metric Congress (MetriKon 2007): Magdeburger Schriften zum Empirischen Software Engineering, pp. 253-266. Shaker Verlag GmbH, Aachen, 2007.

[6] Beck, K. et al. Manifesto for Agile Software Development. www.agilemanifesto.org [2014/08/30].

[7] Beck, K. Extreme Programming Explained: Embrace Change. Addison-Wesley, Reading, 2000.

[8] Blank, S. Why the Lean Start-Up Changes Everything. *Harvard Business Review,* 91, 5, pp. 64-72, 2013.

[9] Blank, S.G. The Four Steps to the Epiphany: Successful Strategies for Products That Win. Cafepress.com, Foster City, California, 2007.

[10] Bosch, J. et al. The Early Stage Software Startup Development Model: A Framework for Operationalizing Lean Principles in Software Startups. In: Fitzgerald, B. et al. (ed.): Lean Enterprise Software and Systems, pp. 1-15. Springer, Heidelberg, 2013.

[11] Bosch, J. Building Products as Innovation Experiment Systems. In: Cusumano, M., Iyer, B. and Venkatraman, N. (eds.): Software Business, pp. 27-39. Springer, Heidelberg, 2012.

[12] Bosch, J. and Bosch-Sijtsema, P.M. Introducing Agile Customer-Centered Development in a Legacy Software Product Line. *Software: Practice and Experience,* 41, 8, pp. 871-882, 2011.

[13] Bower, M. The Will to Manage. McGraw-Hill, New York, 1966.

[14] Boyatzis, R.E. Transforming Qualitative Information: Thematic Analysis and Code Development. SAGE Publications, Thousand Oaks, 1998.

[15] Braun, V. and Clarke, V. Using Thematic Analysis in Psychology. *Qualitative Research in Psychology,* 3, 2, pp. 77-101, 2006.

[16] Cole, R.E. From Continuous Improvement to Continuous Innovation. *Total Quality Management*, 13, 8, pp. 1051, 2002.

[17] Croll, A. and Yoskovitz, B. Lean Analytics: Use Data to Build a Better Startup Faster. O'Reilly, Sebastopol, 2013.

[18] Davenport, T.H. How to Design Smart Business Experiments. *Harvard Business Review,* 87, 2, pp. 68-77, 2009.

[19] Deming, W.E. Out of the Crisis. Cambridge University Press, Cambridge, 1986.

[20] Dybå, T. and Dingsøyr, T. Empirical Studies of Agile Software Development: A Systematic Review. *Information and Software Technology,* 50, 9, pp. 833-859, 2008.

[21] Easterbrook, S., Singer, J., Storey, M. and Damian, D. Selecting Empirical Methods for Software Engineering Research. In: Shull, F., Singer, J. and Sjøberg, D.I.K. (eds.): Guide to Advanced Empirical Software Engineering, pp. 285-311. Springer, London, 2008.

[22] Engström, E. and Runeson, P. A Qualitative Survey of Regression Testing Practices. In: Ali Babar, M., Vierimaa, M. and Oivo, M. (eds.): Product-Focused Software Process Improvement, pp. 3-16. Springer, Heidelberg, 2010.

[23] Fagerholm, F. et al. Building Blocks for Continuous Experimentation. In: Proceedings of the 1st International Workshop on Rapid Continuous Software Engineering, pp. 26-35. ACM, New York, 2014.

[24] Fink, A. Analysis of Qualitative Surveys. In: Fink, A. (ed.): The survey handbook, pp. 61-78. SAGE Publications, Thousand Oaks, 2003.

[25] Fitzgerald, B. and Stol, K. Continuous Software Engineering and Beyond: Trends and Challenges. In: Proceedings of the 1st International Workshop on Rapid Continuous Software Engineering, pp. 1-9. ACM, New York, 2014.

[26] Glaser, B.G. and Strauss, A.L. The Discovery of Grounded Theory: Strategies for Qualitative Research. Aldine de Gruyter, New York, 1967.

[27] Holmström Olsson, H. and Bosch, J. Post-deployment Data Collection in Software-Intensive Embedded Products. In: Herzwurm, G. and Margaria, T. (eds.): Software Business. From Physical Products to Software Services and Solutions, pp. 79-89. Springer, Heidelberg, 2013.

[28] Holmström Olsson, H. and Bosch, J. Towards Data-Driven Product Development: A Multiple Case Study on Post-deployment Data Usage in Software-Intensive Embedded Systems. In: Fitzgerald, B. et al. (ed.): Lean Enterprise Software and Systems, pp. 152-164. Springer, Heidelberg, 2013.

[29] Holmström Olsson, H., Alahyari, H. and Bosch, J. Climbing the "Stairway to Heaven": A Multiple-Case Study Exploring Barriers in the Transition from Agile Development towards Continuous Deployment of Software. In: 38th EUROMICRO Conference on Software Engineering and Advanced Applications (SEAA), pp. 392-399. IEEE Press, New York, 2012.

[30] Humble, J. Why Enterprises Must Adopt DevOps to Enable Continuous Delivery. *Cutter IT Journal,* 24, 8, pp. 6-12, 2011.

[31] Humble, J. and Farley, D. Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation. Addison-Wesley, Upper Saddle River, 2011.

[32] Jansen, H. The Logic of Qualitative Survey Research and its Position in the Field of Social Research Methods. *Forum: Qualitative Social Research,* 11, 2, pp. 1-21, 2010.

[33] Khomh, F. et al. Do Faster Releases Improve Software Quality? An Empirical Case Study of Mozilla Firefox. In: 9th IEEE Working Conference on Mining Software Repositories (MSR), pp. 179-188. IEEE Press, New York, 2012.

[34] Kohavi, R., Crook, T. and Longbotham, R. Online Experimentation at Microsoft. Peer-reviewed workshop paper, Third Workshop on Data Mining Case Studies. http://www.dataminingcasestudies.com/DMCS2009_ExP_DMCaseStudies.pdf [2014/9/1].

[35] Kohavi, R. et al. Online Controlled Experiments at Large Scale. In: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1168-1176. ACM, New York, 2013.

[36] Kohavi, R. et al. Controlled Experiments on the Web: Survey and Practical Guide. *Data Mining and Knowledge Discovery,* 18, 1, pp. 140-181, 2009.

[37] Kohavi, R. and Round, M. Front Line Internet Analytics at Amazon.com. Presentation at eMetrics Summit 2004. http://ai.stanford.edu/~ronnyk/emetricsAmazon.pdf [2014/2/11].

[38] Larman, C. and Basili, V.R. Iterative and Incremental Developments: A Brief History. *Computer,* 36, 6, pp. 47-56, 2003.

[39] Lee, F. et al. The Mixed Effects of Inconsistency on Experimentation in Organizations. *Organization Science,* 15, 3, pp. 310-326, 2004.

[40] Lincoln, Y.S. and Guba, E.G. Naturalistic inquiry. SAGE Publications, Beverly Hills, 1985.

[41] Maurya, A. Running Lean: Iterate from Plan A to a Plan That Works. O'Reilly, Sebastopol, 2012.

[42] McKinley, D. Design for Continuous Experimentation. Presentation at Warmgun design conference. http://www.slideshare.net/danmckinley/design-for-continuous-experimentation [2014/8/29].

[43] Münch, J. et al. Creating Minimum Viable Products in Industry-Academia Collaborations. In: Fitzgerald, B. et al. (ed.): Lean Enterprise Software and Systems, pp. 137-151. Springer, Heidelberg, 2013.

[44] Ohno, T. Toyota Production System: Beyond Large-Scale Production. Productivity Press, New York, 1988.

[45] Osterwalder, A. and Pigneur, Y. Business Model Generation. John Wiley & Sons, Hoboken, 2010.

[46] Pedersen Notander, J., Höst, M. and Runeson, P. Challenges in Flexible Safety-Critical Software Development – An Industrial Qualitative Survey. In: Heidrich, J. et al. (ed.): Product-Focused Software Process Improvement, pp. 283-297. Springer, Heidelberg, 2013.

[47] Poppendieck, M. and Poppendieck, T.D. Implementing Lean Software Development: From Concept to Cash. Addison-Wesley, Upper Saddle River, 2007.

[48] Poppendieck, M. and Poppendieck, T. Lean Software Development: An Agile Toolkit. Addison-Wesley, Boston, 2003.

[49] Richards, C. Certain to Win: The Strategy of John Boyd, Applied to Business. Xlibris, Philadelphia, 2004.

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

[51] Robson, C. Real World Research: A Resource for Users of Social Research Methods in Applied Settings. Wiley, Chichester, 2011.

[52] Rodríguez, P. et al. Survey on Agile and Lean Usage in Finnish Software Industry. In: Proceedings of the ACM-IEEE International Symposium on Empirical Software Engineering and Measurement, pp. 139-148. ACM, New York, 2012.

[53] Royce, W.W. Managing the Development of Large Software Systems: Concepts and Techniques. In: ICSE '87 Proceedings of the 9th International Conference on Software Engineering. Reprinted from Proceedings, IEEE Wescon, August 1970, pages 1-9., pp. 328-338. IEEE Press, Los Alamitos, 1970.

[54] Runeson, P. and Höst, M. Guidelines for Conducting and Reporting Case Study Research in Software Engineering. *Empirical Software Engineering,* 14, 2, pp. 131-164, 2009.

[55] Runeson, P., Andersson, C. and Höst, M. Test Processes in Software Product Evolution – A Qualitative Survey on the State of Practice. *Software Maintenance and Evolution,* 15, 1, pp. 41-59, 2003.

[56] Schwaber, K. and Beedle, M. Agile Software Development with Scrum. Prentice-Hall, Upper Saddle River, 2001.

[57] Ståhl, D. and Bosch, J. Modeling Continuous Integration Practice Differences in Industry Software Development. *Journal of Systems and Software,* 87, pp. 48-59, 2014.

[58] Stapleton, J. DSDM: Business Focused Development. Addison-Wesley, London, 2003.

[59] Steiber, A. and Alänge, S. A Corporate System for Continuous Innovation: The Case of Google Inc. *European Journal of Innovation Management,* 16, 2, pp. 243-264, 2013.

[60] Tang, D. et al. Overlapping Experiment Infrastructure: More, Better, Faster Experimentation. In: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 17-26. ACM, New York, 2010.

[61] Thomke, S.H. Enlightened Experimentation. The New Imperative for Innovation. *Harvard Business Review,* 79, 2, pp. 66-75, 2001.

[62] Thomke, S.H. Managing Experimentation in the Design of New Products. *Management Science,* 44, 6, pp. 743-762, 1998.

[63] Version One. The 8th Annual “State of Agile” Survey. http://www.versionone.com/pdf/2013-state-of-agile-survey.pdf [2014/7/31].

[64] Vinson, N.G. and Singer, J. A Practical Guide to Ethical Research Involving Humans. In: Shull, F., Singer, J. and Sjøberg, D.I.K. (eds.): Guide to Advanced Empirical Software Engineering, pp. 229-256. Springer, London, 2008.

[65] Wang, X., Conboy, K. and Cawley, O. “Leagile” Software Development: An Experience Report Analysis of the Application of Lean Approaches in Agile Software Development. *Journal of Systems and Software,* 85, 6, pp. 1287-1299, 2012.

[66] Williams, L. and Cockburn, A. Agile Software Development: It's About Feedback and Change. *Computer,* 36, 6, pp. 39-43, 2003.

[67] Womack, J.P., Jones, D.T. and Roos, D. The Machine That Changed the World. Rawson Associates, New York, 1990.

Appendix 1. Covering letter

Dear software industry professional

The Finnish software intensive industry is evolving towards a value-driven, adaptive, experimental business model based on the intensive use of customer feedback and usage data.

To allow for the proper targeting of future R&D efforts, it is useful to have an understanding of the current state of the practice regarding value identification and experimentation or testing with customers. It is especially important to find out what kinds of barriers organizations face in moving towards an experimental mode of operation – or what are its potential enablers.

Those are the goals of the study to which we are now seeking participant organizations. Research data is to be collected by interviewing a representative of the organization on themes related to:

- current customer value identification methods
- current experimentation or testing practices involving customers and
- thoughts on future experimentation practices.

Potential interviewees include senior or middle management and experienced development personnel. Planned interview length is approximately one hour. Interview material will be handled confidentially and presented anonymously in research reports. Interview material will be destroyed after the completion of the research project.

Research results will be sent to participating organizations, enabling their use in development efforts.

If your organization would like to take part in the study, please contact us as soon as is convenient for you so that we can schedule the interview. Also contact us if you would like to receive more information about the study before participation. You can find the contact details below.

The study belongs to a Master’s thesis project at the Department of Computer Science at the University of Helsinki. The study is supervised by Prof. Dr. Jürgen Münch and Prof. D.Sc. (Tech.) Tomi Männistö.

Best regards
Eveliina Lindgren
Student
Email: *(information removed)*
Phone: *(information removed)*

Appendix 2. Research information sheet

RESEARCH INFORMATION SHEET

In this study, we explore customer value identification methods and their use in guiding software development. The focus is on the current and possible future use of experimentation with customers, and the barriers and enablers associated with it (see page 2 of this information sheet).

Research data is gathered through individual interviews with representatives of Finnish software development organizations. Interview length is approximately one hour. Interviews will be recorded to facilitate data analysis.

Interview recordings and their transcripts will only be handled by persons involved in the execution of this study, or persons with requisite non-disclosure agreements in place. Direct anonymous excerpts from the interviews may be quoted in the thesis and possible subsequent publications. Organization names will also be disguised. Interview recordings along with their transcripts will be destroyed after the thesis and possible subsequent publications are completed.

Research results will be presented in a Master’s thesis, which will be made publicly available. The study may also be published as an article. Furthermore, research results will be sent to participating organizations.

Participation is voluntary and you can withdraw from the study, without any consequences, at any point either before or during participation by informing the researcher. Taking part in the study is not envisaged to cause any negative consequences.

The study belongs to a Master’s thesis project at the Department of Computer Science at the University of Helsinki. The study is supervised by Prof. Dr. Jürgen Münch and Prof. D.Sc. (Tech.) Tomi Männistö.

You may request additional information about the study at any time. Contact details of the researcher can be found below.

If you agree to participate in the study, please sign the attached consent form.

<u>Contact details:</u>

Eveliina Lindgren
Student
University of Helsinki, Department of Computer Science
Email: *(information removed)*
Phone: *(information removed)*

WHAT IS CONTINUOUS EXPERIMENTATION?

Every software product, functionality and feature includes assumptions or hypotheses about the value it is thought to provide to its users. These may be explicitly stated ("feature X should speed up process Y by 10%"), or remain implicit ("feature Z feels like a good idea"). However, the value of an idea cannot be accurately judged before it has been exposed to its intended users, the customers. It is therefore useful to try out the idea on customers as early on in the development process as possible.

In continuous experimentation, small elements of the product being built are systematically validated through experiments, or tests, with customers. These can include, for example, gathering verbal feedback on an early prototype, or online experiments. Results are collected in the form of descriptive and/or numeric data. The results are analyzed and used to guide the course of development and to generate new ideas. This iterative process of innovation, development, experimentation, and data analysis increases the likelihood of "building the right product".

For more information:

Strategic Research Agenda for Need for Speed, 2013. Digile (ICT SHOK), Finland. Available online, e.g.
http://www.digile.fi/N4S

Appendix 3. Informed consent form

CONSENT TO PARTICIPATE IN RESEARCH

I voluntarily agree to participate in Eveliina Lindgren's MSc thesis study on "Industrial experiences and needs regarding continuous experimentation" *(working title)*.

The purpose and procedures of the study have been explained to me in sufficient detail for me to understand them.

I understand that I can withdraw from the study, without any consequences, at any point either before or during my participation simply by informing the researcher. I may also request additional information about the study at any time. Contact details of the researcher can be found below.

I give permission for my research interview with Eveliina Lindgren to be recorded.

I understand that my anonymity will be ensured in the thesis and possible subsequent publications by disguising my identity. Organization names will be similarly disguised.

I understand that anonymous direct excerpts from my interview may be quoted in the thesis and possible subsequent publications.

I have been informed that the interview recordings along with their transcripts will be destroyed after the thesis and possible subsequent publications are completed.

Two copies have been made of this consent form, one of which will remain with me and the other with Eveliina Lindgren.

________________________________	__________________
Location	Date

____
Signature

____
Name in print

____
Eveliina Lindgren
Student
University of Helsinki, Department of Computer Science
Email: *(information removed)*
Phone: *(information removed)*

Appendix 4. Interview guide

Background of interviewee:

1. What is your current position in the company?

2. How many years have you worked in the company?

Company information:

3. Can you briefly describe the industry sector your company operates in and the type of software you develop?

Theme 1. Current software development practices.

4. What kind of a software development process do you use?

5. Do you use continuous integration?

6. How often do you deploy new versions to production?

Theme 2. Current practices of customer feedback elicitation and use.

7. How do you make sure that you are building the right product?

8. How do you collect customer feedback?
   a. Before development
   b. During development
   c. After deployment

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

9. How often are the aforementioned customer feedback collection methods used?

10. Do you collect data about customer behaviour, for example in the form of product usage data?

11. How do you use the collected customer feedback and other data?
    Is there a link to:
    a. Product development
    b. Further innovation
    c. Business goals and strategy

12. Who is involved in reviewing the collected customer feedback and other data? (Possible prompts: managers, development personnel etc.)

13. How do you prioritize new feature requirements?

14. How do you prioritize implementation options?

15. Do you evaluate whether a newly implemented feature delivered customer value?

16. Who is responsible for customer insight management in your company?

17. How do you see your customer involvement practices in relation to those of other companies in your industry sector?

Theme 3. Future practices of customer feedback elicitation and use.

18. Do you think your current practices of customer feedback collection and customer involvement are adequate?
    a. If not already ideal: How should they ideally be performed in the future?

19. Are there any obstacles to obtaining deeper customer insights? (Possible prompts: technical issues, lack of resources, personnel skills, company culture etc.)

20. What are your company's strengths with respect to generating customer insights? (Possible prompts: technical know-how, ample resources, personnel skills, company culture etc.)

Final questions.

21. Do you have any further comments on customer value -related issues in the context of your company?

22. Do you have any comments or questions related to this interview or the study in general?

Thank you for your time and contribution to the study!

Appendix 5. Analytic codebook

Code Label		Code Family and Code Description
BAC BACKGROUND INFORMATION	■	Top category. Descriptive background information about the interviewee / their company / the company's product(s).
BAC: INTERVIEWEE	■	Subcategory. Descriptive background information about the interviewee.
bac: interviewee: length of employment	■	Interviewee's length of employment in the current company.
bac: interviewee: position	■	The interviewee's position in the current company.
BAC: COMPANY	■	Subcategory. Descriptive background information about the interviewee's company.
bac: company: industry sector	■	A description of the industry sector the company operates in.
bac: company: product description	■	A description of the software product(s) the company develops (if several, those the interviewee is involved with).
BAC: DEVELOPMENT METHODOLOGY	■	Subcategory. Descriptions of the software development methodology used in the company.
bac: development: scrum	■	Scrum/a scrum variant is used. Does not imply a textbook application, only that the interviewee mentions the methodology.
bac: development: unspecified agile	■	A specific agile methodology is not named but the agility of the development practices is either explicitly stated or implicitly visible in the interviewee's statements.
bac: development: kanban	■	Kanban is used. Does not imply a textbook application, only that the interviewee mentions the methodology.
bac: development: lean	■	Some type of lean methodology is used / lean principles are followed. Does not imply a textbook application, only that the interviewee mentions the methodology.
BAC: DEVELOPMENT: PRINCIPLES AND PRACTICES	■	Subcategory. Descriptions of the development practices and principles used in the company.
bac: development: principles: continuous deployment	■	Continuous deployment is used. Use the code for descriptions of a system where new versions are automatically deployed to customers after changes are committed to the code repository (and they have passed through a continuous integration system). NB continuous deployment was not explicitly defined to interviewees.
bac: development: principles: continuous integration	■	Continuous integration is used. NB continuous integration was not explicitly defined to interviewees.
bac: development: principles: MVP	■	The lean startup-inspired MVP approach is used. This is an "umbrella approach" which may include several specific experimentation techniques.

BAC: DEVELOPMENT: PRINCIPLES: RELEASE CYCLE LENGTH	■	Subcategory. Descriptions of the length of the release cycle, i.e. the interval of product version releases to customers (not internal releases which may or may not be more frequent). Applies to scheduled upgrades, not unscheduled, urgent bug fixes.
bac: development: principles: release: long	■	Release cycle > 3 months.
bac: development: principles: release: medium	■	Release cycle <= 3 months.
bac: development: principles: release: short	■	Release cycle <= 1 month. Includes continuous deployment.
CHA CHALLENGES	🟥	Top category. Descriptions of the challenges related to implementing/moving towards continuous experimentation.
CHA: ORGANIZATION	🟥	Subcategory. Descriptions of the challenges related organizational issues.
cha: organization: culture	🟥	The organizational culture is not sufficiently conducive to an experimental way of working. Includes a range of issues with respect to roles, responsibilities, and ways of working.
CHA: B2B DOMAIN SPECIFIC	🟥	Subcategory. Descriptions of the particular challenges of operating in a business-to-business environment.
cha: b2b: access to end users	🟥	Collecting relevant feedback is difficult as sufficient access to end users does not exist.
cha: b2b: consent to usage data collection	🟥	Customer consent to collecting product usage data is withheld or uncertain.
cha: b2b: customer organization culture	🟥	The customer organizations' culture is not sufficiently conducive to an experimental way of working. Includes a range of issues with respect to roles, responsibilities, and ways of working.
CHA: DATA MANAGEMENT	🟥	Subcategory. Descriptions of the challenges related to data (customer feedback and product usage data) management.
cha: process: data analysis	🟥	The collected feedback / product data is not analyzed rigorously enough.
cha: process: sharing of data	🟥	The collected feedback/data (or the resulting analyses) are not easily available to or shared with all relevant stakeholders within the company.
CHA: PRODUCT MANAGEMENT	🟥	Subcategory. Descriptions of the challenges related to product management.
cha: product: defining product roadmap	🟥	Challenges in extending or aligning the roadmap of an established product or an MVP/MVF. Includes e.g. challenges in prioritizing requirements, and obstacles in deciding how far to develop a product before exposing it to the market.
cha: product: identifying metrics	🟥	Challenges in identifying and applying the appropriate metrics to evaluate the customer value or success of a software product or feature (in experiments/general usage).

cha: product: productization	🟥	Challenges in productizing or commercializing the product/service.
cha: product: release cycle speed	🟥	The release cycle is not rapid enough.
CHA: RESOURCES	🟥	Subcategory. Descriptions of the challenges related to a lack of resources.
cha: resources: budget	🟥	A limited budget restricts experimentation.
cha: resources: domain knowledge	🟥	Lack of domain knowledge restricts experimentation.
cha: resources: technical	🟥	Technical obstacles restrict experimentation.
cha: resources: time	🟥	Lack of time restricts experimentation.
STRENGTHS	🟩	Top category. Descriptions of the enabling factors of experimentation.
strengths: customer and domain knowledge	🟩	Close customer relationships and deep customer and domain knowledge facilitate experimentation.
strengths: organizational culture	🟩	The organizational culture facilitates experimentation. Includes a range of issues with respect to roles, responsibilities, or ways of working.
strengths: release cycle speed	🟩	A rapid release cycle facilitates experimentation.
strengths: technical competence	🟩	Technical competence facilitates experimentation.
strengths: technical tools available	🟩	The availability of technical tools facilitates experimentation.
TEC TECHNIQUES OF CUSTOMER FEEDBACK COLLECTION	🟪	Top category. Descriptions of the techniques used to collect explicit and implicit customer feedback.
TEC: ANALYSIS - PROBLEM ANALYSIS AND REQUIREMENTS ENGINEERING	🟪	Subcategory. Customer feedback collection techniques which are used during problem analysis and requirements engineering.
tec: analysis: customer representatives	🟪	A customer representative (who is in touch with the end user needs) provides feedback.
tec: analysis: internal brainstorming	🟪	Company's internal brainstorming sessions are used to generate ideas for product development. Applicable only to products where the company's personnel also represents potential customers.
tec: analysis: proofs of concept	🟪	PoCs are used to collect customer feedback.
tec: analysis: prototyping	🟪	Prototypes are used to collect customer feedback. Includes conceptual, paper, and programmatic prototypes.
tec: analysis: stakeholder interviews	🟪	Potential or current customers are interviewed specifically in order to try to understand their needs, opinions, and problems. Includes focus groups. NB do not confuse with general customer meetings.
tec: analysis: use cases	🟪	Use cases or use scenarios are used to collect customer feedback.
TEC: DEVELOPMENT -DESIGN, CONSTRUCTION AND TESTING	🟪	Subcategory. Customer feedback collection techniques which are used during product or feature development (design, programming and quality assurance).

tec: development: acceptance testing	■	Acceptance testing performed by customers.
tec: development: alpha/beta testing	■	Alpha or beta testing performed by customers.
tec: development: informal end user tests	■	Informal customer tests are used to collect feedback. For example "get out of the building"- type scenarios, where potential customers are asked about their opinion of a product under development.
tec: development: internal experiments	■	Company's internal experiments are used to generate feedback. Applicable only to products where the company's personnel also represents potential customers.
tec: development: labs website	■	A "labs" website is used to collect customer feedback. The site includes early prototypes/versions of products.
tec: development: pilot customers	■	Pilot customers (users) are used to collect feedback.
tec: development: usability/UX testing	■	Different forms of usability or UX testing/experiments/evaluation methods are used to collect customer feedback. Includes usability testing, UI labs, UI evaluation, UX diary studies etc.
TEC: EVOLUTION AND MAINTENANCE	■	Subcategory. Customer feedback collection techniques which are used after the initial deployment of a product or a feature.
tec: evolution: A/B testing	■	A/B testing is used to collect customer feedback. Includes multivariate testing and segmentation of the customer base for running different versions. NB A/B testing was not explicitly defined to interviewees.
tec: evolution: bug report analysis	■	A helpdesk/support/bug reporting system is used to collect customer feedback.
tec: evolution: feature voting	■	Feature voting (voting for existing features or bug fixes etc.) is used to collect customer feedback. The system may include the option of suggesting new features.
tec: evolution: in-product surveys	■	In-product surveys or polls are used to collect customer feedback.
tec: evolution: usage data analysis	■	Product usage data (including product performance data) collection and analysis is used to obtain implicit customer feedback.
TEC: GENERAL, PHASE-INDEPENDENT	■	Subcategory. Generic customer feedback collection techniques which are used during all phases of product development.
tec: general: customer surveys	■	Customer surveys are used to collect customer feedback. Includes separate end user surveys in a B2B environment.
tec: general: direct customer feedback	■	Direct customer feedback concerning the product is collected. Includes means such as face to face meetings, seminars, demos, email, phone, web-based contact or feedback forms etc.
tec: general: market research	■	Market research/analysis is used to collect customer feedback or information about customer needs.
TEC: PROSPECTIVE TECHNIQUES	■	Subcategory. Customer feedback collection techniques whose implementation is planned or currently under development.

tec: prospective: A/B testing	■	There are plans to use A/B testing to collect customer feedback in the future. Includes plans to improve current practices.
tec: prospective: usage data collection	■	There are plans to collect and analyze product usage data (including product performance data) in the future to obtain implicit customer feedback. Includes plans to improve current practices.
USE OF COLLECTED FEEDBACK	■	Top category. Descriptions of how the collected customer feedback is used in the software product development process.
use: integration into product development	■	Descriptions of how the collected customer feedback is integrated into the product development process. Includes descriptions of how and by whom feedback is analyzed, how it relates to the product roadmap / backlog etc.
use: basis for innovation	■	Descriptions of using the collected customer feedback specifically as a basis for innovation or improvement (not simply for troubleshooting or straightforward requirements elicitation).
use: connection to business strategy	■	Descriptions of the connection between the collected customer feedback and the company strategy or business goals. Includes descriptions of the lack of connection.
USE: PRIORITIZATION OF REQUIREMENTS	■	Subcategory. Descriptions of requirements prioritization practices.
use: prioritization: co-operation	■	Requirements are prioritized in co-operation with the customer(s). Applicable in a business-to-business environment.
use: prioritization: customers	■	Requirements are prioritized by the customer(s). Applicable in a business-to-business environment.
use: prioritization: internally	■	Requirements are prioritized within the software development company.
USE: RELEVANCE OF FEEDBACK	■	Subcategory. Descriptions of the reasoning behind current or planned ways of using customer feedback.
use: relevance: fact-based decision making	■	Data-driven approach to decision making in order to minimize decisions based on intuition or individual opinions.
use: relevance: identifying customer value	■	Identifying and focusing on products and features that create customer value.
USE: REQUIREMENTS, SOURCE OF	■	Subcategory. Descriptions of where requirements originate from.
use: requirements: explicit customer requirements	■	Requirements originate from explicitly defined customer requirements.
use: requirements: implicit in customer feedback	■	Requirements originate from the collected customer feedback. However, they are implicit within the feedback (not explicitly predefined by customers).
use: requirements: internal ideas	■	Requirements originate from the software development company's internal ideas. Customer feedback plays a minor role.