Raising the Odds of Success: The Current State of Experimentation in Product Development[1]

Eveliina Lindgren ✉ [1], Jürgen Münch [1,2]

[1] Department of Computer Science, University of Helsinki
P.O. Box 68, FI-00014 University of Helsinki, Finland
eveliina.lindgren@alumni.helsinki.fi, juergen.muench@cs.helsinki.fi

[2] Reutlingen University
Alteburgstraße 150, 72762 Reutlingen, Germany
Juergen.Muench@Reutlingen-University.DE

Abstract

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

Objective: This paper explores the state of the practice of experimentation in the software industry. It also identifies the key challenges and success factors that practitioners associate with the approach.

Method: A qualitative survey based on semi-structured interviews and thematic coding analysis was conducted. Ten Finnish software development companies, represented by thirteen interviewees, participated in the study.

Results: The study found that although the principles of continuous experimentation resonated with industry practitioners, the state of the practice is not yet mature. In particular, experimentation is rarely systematic and continuous. Key challenges relate to changing the organizational culture, accelerating the development cycle speed, and finding the right measures for customer value and

[1] NOTICE: this is the author's version of a work that was accepted for publication in Int'l Journal on Information and Software Technology (IST). A definitive version was subsequently published in Information and Sofware Technology, [2016] DOI 10.1016/j.infsof.2016.04.008

product success. Success factors include a supportive organizational culture, deep customer and domain knowledge, and the availability of the relevant skills and tools to conduct experiments.

Conclusions: It is concluded that the major issues in moving towards continuous experimentation are on an organizational level; most significant technical challenges have been solved. An evolutionary approach is proposed as a way to transition towards experiment-driven development.

Keywords: Product Management · Hypothesis-Driven Software Development · Experiment-Driven Software Development · Continuous Experimentation · Lean Startup · Lean UX

1 Introduction

Nowadays, software-based products are often created for new or highly dynamic domains with many technical and business uncertainties. Such products often aim at offering solutions that have not existed before, or satisfying needs that their customers or users were not aware of. In such situations, it is often difficult or impossible to predict which product features or attributes might create value for the customers, even if the customers are asked. One of the reasons is that in such domains, the relation between cause and effect can only be determined in retrospect. In consequence, requirements are not obvious and cannot be defined in advance. Of course, projects for which the requirements can be determined upfront still exist, but they represent a very small percentage of all software projects [1].

If it is not possible to know what needs to be built, it is exceedingly difficult to guess what to build. Studies show that most development ideas actually create zero or negative value [2]. Consequently, the critical questions are how to avoid wasting effort by building software that does not deliver value for customers or users, and how to raise the odds of success with products that satisfy intended outcomes before running out of resources. Currently many companies rely on intuition and experience when making product development decisions, thereby exposing their organizations to high development and business risks [3].

A promising way of preventing such risks is to continuously identify, prioritize, and validate critical product assumptions in all phases of the development cycle. An example of a product assumption is that a feature will have a specific effect in a defined time span. Product assumptions can be tested with experiments: assumptions are transformed into hypotheses and the scientific method is applied in order to support or refute the hypotheses. Results from such experiments can inform product

decisions, thereby increasing the speed of learning and helping to reach product goals. New possibilities to deliver software frequently to customers allow software-centric companies to validate the value of products, functionalities, features, and capabilities in their actual marketplace by conducting a constant series of experiments. This experiment-driven approach is currently most prevalent in the cloud computing environment (especially with software as a service companies), but it is beginning to affect the development of all Internet-connected software products [4]. It can be seen as a subset of the data-driven decision making approaches since experiments offer one method of backing up decisions with verifiable data. Data-driven decision making is opposed to intuitive or opinion-based decision making.

Although a significant amount of experience exists on applying iterative delivery and continuous development methods, the integration of such methods with a process for continuously validating product assumptions through experimentation seems challenging. Despite the recent interest in experimentation as an integral part of software development, industrial experimentation experiences have not been studied widely: most examples come from eminent web-facing companies such as eBay [5], Google [6, 7], and Microsoft [2]. There has also been relatively little discussion about the obstacles and enabling factors that practitioners face.

Companies aiming at moving towards an experiment-driven approach might benefit from an overview of existing experiences and the current status of the transformation journey of other companies. Researchers might be interested in better understanding practical challenges and contexts in order to identify underlying research questions or research gaps. Therefore, the goal of this paper is to present an interview-based qualitative survey that aims at developing an understanding of the state of the practice of experimentation in software product development. Key challenges and success factors related to the approach are identified, and a proposition for a practitioners' starter kit for moving towards experiment-driven development is presented. Some of the findings in this paper were previously presented in a separate article [8].

The paper is organized as follows. Section 2 presents the problem background and related work. Section 3 defines the research questions and describes how the study was designed and executed. The results of the study are presented in Section 4, followed by a discussion in Section 5. Finally, the paper is concluded in Section 6.

2 Background and Related Work

This section presents the problem background and related work. In particular, the role of customer feedback and customer behavior, recent technological trends that support experimentation, and differences between agile methods and experimentation are outlined. In addition, an overview of related work with respect to experimentation is presented, and criteria are introduced that characterize systematic experimentation in the context of this article.

2.1 The Role of Customer Feedback and Customer Behavior

The development of new products and services is risky and hard. Often, the wrong products or services are developed. This leads to the crucial question of what should be developed in order to create customer value. The answer to this question would be easy if customers knew what they want. However, asking customers about what they want typically does not provide good answers: customers are not good in predicting what they want, they are often not aware of potential solutions, and there is a gap between what customers are actually doing and what they think or say they are doing. Therefore, the basis for product decisions should stem from observing actual customer behavior: for instance, how they use products and services.

Experiments with customers usually require that customer feedback can be elicited, but they do not always require upfront development efforts or an existing technical infrastructure. An example of such a non-technical experiment is a customer interview to analyze a problem hypothesis. However, experiments often do require that simplified versions of features or products exist that can easily be delivered to customers in order to understand how customers are using them.

Many methods are available for conducting experiments with customers. The selection of an appropriate method depends on the type of hypothesis that needs to be tested, as well as the purpose and context of the experiment. Examples of experiment methods include A/B tests, multivariate tests, product analytics, landing pages, fake door tests, problem interviews, solution interviews, and tests with wireframes, mockups, or Wizard of Oz minimum viable products. All these methods require the elicitation of qualitative or quantitative customer feedback.

It should be noted that a difference must sometimes be made between customers and users. Customers pay for a product or service, while users use the product or service. In some situations (e.g., in the business-to-business domain) this difference needs to be considered. However, this article uses the terms “customer” and “user” interchangeably. Overall, having the capabilities and competencies to elicit and analyze customer feedback, including product usage data, is an important prerequisite for experimentation.

2.2 Recent Technological Trends that Support Experimentation

Recent technological trends and advancements support both the ability to rapidly deploy software chunks to customers and the ability to elicit customer feedback, including product usage data.

One of these technological trends is continuous deployment, i.e., the incremental and automated delivery of software to production environments. Continuous deployment enables increasing the speed of deployment of software chunks, thereby allowing faster learning about the real value of the deployed software chunks. It should be mentioned that continuous deployment cannot be easily established in all domains, and that it requires other capabilities such as continuous integration and test automation as prerequisites. However, the popularity of continuous deployment appears to spread from initial domains such as web development, where it is a quasi-standard, to other domains such as information systems.

Another technological trend that supports experimentation is the increasing use of software as a service (SaaS) as a delivery model. Its use is spreading from the initial SaaS domains such as mobile services to other domains such as business-to-business software. SaaS creates a closer link to customers compared to other delivery models such as shrink-wrapped software or on-premises software. It provides the technical possibilities to observe customer behavior directly and ideally in real time, thus opening many opportunities for experimentation. It enables answering questions such as: Which features are customers using? In which sequence? How frequently? For how long? Do they unexpectedly start or stop using features? Do they use features differently?

There are several other trends that support experimentation. One example is "DevOps", the close collaboration and communication between software development and IT operations. With DevOps, experiences and observations from operations can be directly fed back to developers and thus be taken into account when making product decisions.

Another trend is the closer integration of product discovery, product validation, and delivery activities. An example of such an approach is the concept of a "design sprint" that integrates discovery and testing with target customers in a five-day process. The concept was developed and popularized by Google Ventures [9].

2.3 Experimentation versus Agile Methods

The experiment-driven approach to software development tends to focus on what to develop and is closely linked to business development and customer development. Product roadmaps are seen as lists of untested assumptions that need to be systematically tested with experiments. Experiments are done either to generate insights or to understand the relationship between specific actions (such as the implementation of a specific feature) and reaching goals (such as delivering monetizable value to users). In order to test if software delivers value it is often necessary to easily deploy software (e.g., a minimum viable feature) so that customer behavior can be observed. The measure of progress is simply speaking the velocity of delivering value to customers. An example metric for this measure of progress is customer retention.

During the last decades, agile software development methods have permeated the industry [10]. They emphasize iterative and incremental development and are people-centric. Agile methods tend to target describing how to develop. The measure of progress is working software. An example metric for this measure of progress is built velocity.

Agile methods allow reprioritizing which features to develop during the development process. This is often done by reordering the so-called product backlog, a prioritized list of features that are intended to be implemented. In addition, the frequent deployment of software to customers (e.g., through continuous software development as described by Fitzgerald et al. [11]) enables the observation of changes in customer behavior. However, agile methods such as Scrum are typically development-centered and focus on building features. They usually do not guide the prioritization of the backlog based on validated results from experiments. In addition, the frequent deployment of software to customers does not necessarily mean learning more often. Both agile and continuous development methods therefore need to be combined with the constant validation of product assumptions.

It should be noted that a principle behind agile methods is that the "highest priority is to satisfy the customer through early and continuous delivery of valuable software" [12]. Agile methods provide the prerequisites for doing this, especially by welcoming changing requirements, delivering working software frequently, and emphasizing that business people and developers must work together throughout the project. However, agile methods provide only little guidance on what to

develop to deliver value. Agile development methods focus more on the building aspects while the experiment-driven approach focuses more on the testing and learning aspects.

Combining agile development with an experiment-driven approach promises to drive development efforts towards value delivery. For example, a sprint can be used to create a prototype of a new feature. Afterwards the feature is tested with an A/B test. Results of this test influence the planning of the next sprint, e.g., the full feature is implemented in case the experiment showed the expected results.

2.4 Approaches to Experimentation

Several case studies on companies' experimentation experiences have been published recently. Microsoft's experiences with systematic large-scale online controlled experiments are recounted in numerous reports, for instance [2]. Google purports to experimentally evaluate almost every change that has the potential to affect the user experience [7]. Supporting and fostering continuous innovation is a key element of the Google experiment system [6]. The Netflix "consumer data science" approach is two-staged: experiments are first conducted offline, and if they succeed, an online customer experiment is executed to provide further validation [13].

Adobe's "Pipeline" innovation process attempts to maximize the learning about a given problem through rapid prototyping and frequent customer evaluation [14]. eBay uses a multitude of experimental techniques in addition to online controlled experiments, such as usability testing, focus groups, and diary studies [5]. The diverse experimentation practices of Intuit encompass the whole software lifecycle [4]. Finally, experimentation is also a viable approach when developing software products in academia-industry collaborations [15, 16].

The interest in experimentation in product development has also generated some academic research on industrial practices. Despite the above-mentioned examples from eminent companies, Karvonen et al. [17] conclude that overall, experimentation with customers is rare in industry. The authors note that although companies see many benefits in the approach, there are also barriers. Similar work has been conducted by Sauvola et al. [18].

In addition to such reports on the practical application of experimentation, experiment-driven approaches have recently been described in the scientific literature. This related work on can be grouped along the following questions: How do we mature existing development practices in a way that experiment-driven development can be applied? How do we organize the key activities as part of the development process? How do we develop a better understanding of customers and users?

What are the challenges companies experience when introducing an experiment-driven approach? Which challenges are specific for a certain context or situation? What are the experiences from companies that apply such an approach?

Most of these questions are not comprehensively addressed by the current scientific literature. In the following, selected work is presented that addresses one or more of the above-mentioned questions.

Holmström Olsson et al. [19] 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 rapid experiments that utilize instant customer feedback, including product usage data, to identify customer needs.

This final stage is further systematized by Bosch [4]. He emphasizes constantly generating new ideas to test with customers, suggesting that the approach is best described as an innovation experiment system. Bosch proposes using 2-to-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. Various experiment techniques can be used throughout development. Experimentation does not necessarily require functioning software. Furthermore, the scope of experiment-driven development can vary from new products to new features and feature optimization.

Fagerholm et al. [16] combine the above-mentioned ideas with key elements from the lean startup methodology [20] 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. 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" [16]. This experiment-driven learning process is supported by a technical infrastructure that 1) enables the lightweight releasing of minimum viable products (MVP) or minimum viable features (MVF), 2) provides means for advanced product instrumentation, and 3) supports the design, execution, and analysis of experiments. Fagerholm et al. also provide a description of the roles, tasks, and information artefacts that are required to run a system of continuous experimentation.

The Hypothesis Experiment Data-Driven Development (HYPEX) model [21] is another recent approach for integrating feature experiments with customers into the software development process. The model presents a systematic set of practices designed to shorten the customer feedback loop and thus ensure a better correspondence between customer needs and development efforts. Features are seen as hypotheses to be tested by feature experiments. Feature experiments consist of the deployment of a minimum viable feature and a comparison of the expected and the actual user behavior that the feature drives. Based on the results, decisions are made about the full implementation of the feature, the selection of the next features to test or the abandonment of the feature implementation. The heavy utilization of customer feedback in software development enables informed, data-driven decision making.

Experimentation with customers is also a key part of the customer development approach first outlined by Blank [22], and later developed within the lean movement for instance by Alvarez [23]. The catchphrase "get out of the building" encapsulates the customer development approach of seeking out and learning from real-life customers. Similarly, continuous experiments are also used in lean user experience (UX) to validate design ideas [24]. This experimental approach is often referred to as "hypothesis-driven development". Like the approaches mentioned earlier in this section, these approaches also strive to highlight customer centricity and aim at raising the odds of product success by reducing uncertainties.

To summarize, various approaches to experimentation have been proposed, and many eminent software companies actively use experimentation. However, documented experiences of the various approaches are scant in the scientific literature, and not many scientific studies on current industrial practices were found.

2.5 Systematic Experimentation

Based on the aforementioned approaches, this paper uses the following criteria to characterize systematic experimentation:

- Assumptions need to be tied to higher-level business considerations and prioritized based on these considerations. A risky assumption to be prioritized could be, for instance, that the implementation of a specific feature drives results towards a user or business goal.
- The assumptions to be tested need to be transformed into falsifiable hypotheses.
- The experiments need to be designed and conducted appropriately to test these assumptions.

- The experiments need to be analyzed and interpreted in the context of the higher-level business considerations.
- If the experiment does not show the expected results, an analysis of the reasons needs to be done.
- The experimental results need to be used 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 Study Approach

This section provides an overview of the study approach. It begins by presenting the research questions, and then goes on to describe the research methods. The particulars of executing the study are presented next, followed by a discussion on the study's validity.

3.1 Research Questions

Based on the study goals, the following research questions were defined:

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?

3.2 Study Design

The study was founded on a qualitative survey design. A qualitative survey is similar to the widely used multiple case study method [25, 26], but rather than providing an in-depth analysis of particular cases, the survey is more concerned with providing a multifaceted view of the topic of interest [25, 27].

Fink [27] 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, the sample size is limited due to access or resource constraints.

Finally, qualitative surveys are not to be confused with quantitatively oriented statistical surveys. They do not depend on the use of statistical survey instruments and analysis methods, or seek statistical representativeness and generalizability [25, 27].

3.3 Data collection, sampling, and analysis

Semi-structured individual interviews with industry practitioners were used to collect data, since they enable focusing on predefined research topics while also being highly flexible to allow for unforeseen information [28]. Flexibility in conducting the interviews was necessary due to the exploratory nature of the study. To structure the interviews, an interview guide was developed, outlining the key topics, questions, and prompts. Easy "warm up" and "cool down" questions were asked at the beginning and end of the interviews. The main topics of the interviews, along with example questions, are defined in Fig. 1 (the complete interview guide is available in Appendix 1).

A purposive, non-probability sample was chosen for the study, as is common in qualitative surveys [25, 27]. Software development companies of various sizes, domains of operation, and stages of life cycle (e.g., startup, scaling, and established companies) were sought to achieve a diverse set of participants. Companies with proprietary software product development were preferred in order to gain an understanding of their particular situation, as opposed to companies specializing exclusively in consultation or customer projects. Furthermore, interviewees from different roles and with solid experience in the software industry were sought to enable varied, knowledgeable views on the study subjects.

The interview data was examined through thematic coding analysis [28]. Thematic analysis was selected since it offers an accessible and flexible approach to the analysis of qualitative data. The analysis was based on an iterative coding process, during which a codebook was developed inductively based on the interview data. Descriptive, analytic, or category marker codes were generated depending on the analytic needs. The analysis was performed partially in parallel with data collection. The process was therefore evolutionary: preliminary analysis affected subsequent data collection, and vice versa. The analytic codebook is available in the Figshare repository [29].

Fig. 1. Main interview topics and example questions.

1. Current software development practices
   a) What kind of software development process do you use?

2. Current practices of customer feedback elicitation and use
   a) How do you make sure that you are building the right product?
   b) How do you collect customer feedback?
   c) Do you collect data about customer behavior, for example in the form of product usage data?
   d) How do you use the collected customer feedback and other data?

3. Future practices of customer feedback elicitation and use
   a) Do you think your current practices of customer feedback collection and customer involvement are adequate?
   b) Are there any obstacles to obtaining deeper customer insights?
   c) What are your company's strengths with respect to generating customer insights?

3.4 Study Execution

Study participants were recruited among the affiliates of the Need for Speed research program [30] and, outside the research program, through the professional contacts of the authors. Due to practical constraints, only companies operating in Finland were considered. Gatekeepers were contacted at each company, who either participated in the study themselves or suggested a suitable interviewee. In accordance with ethical guidelines [31], the purpose and procedures of the study were shared with the participants via an information sheet, in addition to which they were asked to give voluntary informed consent to partake in the study.

The recruitment resulted in the participation of ten ICT companies. The focus was on their software product development functions. Table 1 gives a characterization of the companies by size, domain, and product orientation (more details are not disclosed due to confidentiality reasons). Three of the companies can be described as startups.

The companies were represented by thirteen interviewees. 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 individual interviews were conducted in Finland between February and April 2014. 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 audio-recorded to allow for accurate, detailed data analysis. Eleven interviews were conducted by one researcher, and in the remaining two cases two researchers were present. Eleven interviews were performed face to face on the interviewees' company premises, one via video conferencing, and one as a voice over IP call. To further facilitate data analysis, interview recordings were transcribed verbatim shortly after each interview. The transcripts were coded and analyzed using ATLAS.ti [32].

Table 1. Participating companies (size classification: small < 50, large > 250)

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

Unlike the other companies who only had one representative, company C was represented by four interviewees. Their answers were merged together to form an overall impression of the company. As regards company E, their software development practices 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 not included in the results presented in Sections 4.1 and 4.2, and is only included in Sections 4.3–4.5.

3.5 Validity considerations

Different frameworks exist for assessing the validity and trustworthiness of empirical studies. A commonly used framework consists of four criteria: internal validity, construct validity, reliability,

and external validity [28]. Some researchers propose alternative frameworks for assessing flexible design (qualitative) studies. One such proposal is given by Lincoln and Guba [33] (credibility, transferability, dependability, and confirmability). However, the former framework can also be operationalized for flexible design studies [28], and it is therefore used in the discussion below. Internal validity is not discussed since causal relationships were not examined in the present, mainly descriptive study.

Various steps were taken to improve the study's construct validity. First, the overall goals and procedures of the study and the central concept of continuous experimentation were shared with participants prior to the interviews. It was also explained that their companies were not expected to follow particular procedures or possess particular capabilities related to software development or other functions. Second, using semi-structured interviews for data collection enabled the asking of clarifying questions for all involved parties. The goal of these activities was to prevent misunderstandings between researchers and participants, and also to avoid interviewee bias.

Nevertheless, the reported results are based on the personal perceptions of each interviewee. Since only one interview was conducted with most companies, all relevant aspects of the study topic may not have been covered. Interviewees may also have given answers which do not fully reflect the reality of their companies. The latter threat is mitigated by the fact that the interviewees had no apparent incentive to polish the truth. Since contact with the participants was brief, misunderstandings on the researchers' part cannot be excluded, although email clarifications were requested from the interviewees when in doubt.

Further steps to improve the study's construct validity included the development of an interview guide and an analytic codebook. These artefacts were developed and reviewed among the researchers. A pilot interview was conducted by the researchers to test the interview guide (since the guide was found to be functional, the pilot interview data was also included in this paper). The analytic codebook was developed iteratively and incrementally during data collection and analysis with the aim of representing the whole data set. The above-mentioned steps also aimed at improving the study's reliability by combating researcher bias. To further improve reliability and openness, the interview guide and the codebook were made publicly available [29], and the study procedures were reported in detail.

External validity is restricted due to sampling procedures and the limited scope of the study. Although a variety of companies from different domains contributed to the study (Table 1), some domains, such as embedded systems, were not represented. Furthermore, the study scope was

limited to the Finnish software industry. The results therefore cannot be directly transferred to other contexts, although analytical generalization may be possible in similar contexts.

4 Results

This section first outlines the software development practices of the participating companies. The companies' practices of eliciting and using customer feedback are considered next, after which the challenges and success factors with relation to continuous experimentation are presented.

4.1 Software Development Practices

All companies mentioned that they utilize agile methods such as Scrum, Kanban, Lean, or a customized agile approach. All companies also stated that they use continuous integration (CI) but, consistent with previous research [34], there was variability in how CI was interpreted and implemented. These findings are based on the interviewees' informal descriptions of their development approach rather than a formal questionnaire or definition provided by the researchers. The general impression of the companies' development practices was similar to a recent survey [10], although the use of Extreme Programming was not mentioned in the present study.

The release cycle length for new product versions ranged from under one month (56%) to less than three months (33%) or more (11%). 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 was that deployments were quite lightweight and flexible, except for customer-specific installations of business-to-business software, in which case the customers' varying business cases complicated matters. However, the particularities of the companies' deployment pipelines were not investigated further in this study.

4.2 Practices of Eliciting and Using Customer Feedback

The companies used a wide array of techniques to learn about customer needs. 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.

Implicit customer feedback in the form of product usage data was collected by five companies (55%). In many cases the product instrumentation only covered 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." (senior software architect)* The interviewees also recognized that despite the unavoidable upfront investments, the enhanced capacity to learn about customer value through product data *"ha[s] a positive impact […][on] our business as well, eventually." (senior manager)*

Nevertheless, the participants did not regard purely quantitative product usage data as a panacea for the extensive knowledge requirements of software development. They considered quantitative data 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." (senior manager)* Rigorous data analysis was required for *"figur[ing] out root causes. So it's […] brainwork, […] that's the difficult bit." (middle manager)* Some participants brought forward the viewpoint that qualitative customer feedback can facilitate this analysis by illuminating the reasoning behind customer behavior: *"I think both are […] needed […]. What the users say or think and […] what they actually do, [it] might be […] a little bit different in some cases."(middle manager)*

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. Four companies (44%) used A/B or multivariate testing, 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.

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 participants thought that A/B testing may be hard to justify to stakeholders if it causes additional R&D expenses.

The collected customer feedback was typically analyzed to extract work items which were then prioritized into a product backlog. There was some variation in how the interviewees described their approach to this process. Particularly the startup representatives emphasized the need to explore the customer feedback beyond face value in order to generate new ideas and innovations: *"The interesting thing is their complaint, not the solution that they are providing." (senior manager)*

The level of involvement of different stakeholders in analyzing customer feedback varied: in some cases, both management and the development team were heavily involved with the analysis and the responsibility was shared. In other cases, the responsibility was on management roles but all the material was reviewed together with the team. Lastly, 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 since it may lead to the loss of valuable insights: *"[T]here is still a lot [of room] for improvement in that area […] [I]t's very, very important […] to spread all this [customer] information even to the developers, because they […] typically [have] great ideas." (middle manager)*

Two approaches emerged regarding the influence of customer feedback on business strategy and goals. First, some company representatives considered that the strategy is continuously being revised based on the feedback. This approach was predominant among the startup companies. As one interviewee said: *"Our strategy is to experiment." (senior manager)* In the second approach, the emphasis was on the business strategy and goals guiding the development activities, rather than the other way around: *"[O]f course everything is connected, but it [the customer feedback] doesn't direct our business goals." (middle manager)* This approach appeared to be more typical for established companies.

4.3 Domain-Independent Challenges of Continuous Experimentation

Table 2 provides an overview of the challenges that were mentioned in this study. The challenges are sorted by the frequency of occurrence in the data set. Although the challenges emerged during the interviews and the interviewees were not asked for specific topics, it should be noted that the occurrence of the challenges does not necessarily reflect the accurate distribution over all companies. In addition, the ranking of occurrence should not be interpreted as a ranking of importance or urgency. Some topics might, for instance, have been forgotten by the interviewees or not mentioned due to time constraints.

Limited resources were a highly recurrent theme in the interviews. There was a lack of time and funding to invest in experimentation. On the other hand, some interviewees emphasized the potential long-term benefits of investing in experimentation.

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. In the words of one interviewee: *"To measure the right thing is the hard thing, to know […] 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." (senior manager) Similarly, taking into consideration the different characteristics of products was a challenge: *"We measure the wrong things for some of [the] products." (middle manager)* Particular difficulties related to which metrics and techniques of collecting customer feedback to use when scaling up a product: *"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 growth]." (middle manager)*

Table 2. Challenges and frequency of occurrence.

Challenge	**Freq. of occurrence by company**
Time	7
Funding	6
Identifying metrics	6
Release cycle speed	6
Organizational culture	5
Availability and sharing of data	4
Data analysis	4
Defining product roadmap	4
Technical obstacles	3

Concern over slow release cycles was one of the central themes in terms of product management. Despite the already relatively short cycle lengths (Section 4.1), the wish to further accelerate development was evident in the interviewees' comments. Reasons for this perceived sluggishness included R&D task overload and bottlenecks in the development process. Focusing on products and features that create the most customer value was seen as a way to speed up development: *"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." (middle manager)*

Half of the company representatives considered organizational culture a major obstacle to moving towards an experimental mode of operation. We refer with the term "organizational culture" to the collective "values and behaviors that contribute to the unique social and psychological environment

of an organization" [35]. One interviewee mentioned: *"I would say that the technical things are not […] even close to the weight of the cultures' obstacles." (senior software architect)* Another interviewee agreed that trouble in embracing experimentation *"has nothing to do with technology". (middle manager)* The overarching issues with respect to organizational culture in established companies included a perceived lack of agility, proactivity, and transparency – either within the company or in relation to the company's customers. Startup representatives commented on issues such as undefined company values and the turbulence of constantly re-evaluating ways of working.

Interviewees also expressed concern over deficiencies in the analysis of collected customer feedback and other data: *"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." (senior manager)* Lack of time and analytic expertise emerged as possible reasons for inadequate data analysis. Obstacles were also encountered in the availability and sharing of data with all relevant stakeholders. For instance, suitable data repositories were missing. As one interviewee said: *"The data is scattered all over the place. […] [W]e are quite far from providing [a] really convenient, broad spectrum of data to all of the employees." (middle manager)*

A further set of issues was related to defining the product roadmap. Identifying a truly viable MVP was considered *"very easy to say, very hard to do." (middle manager)* As regards established products, one interviewee described formulating a product backlog as *"black magic" (middle manager)* as it could be so challenging to combine both the product vision and the changing requests and demands of various customers.

Technical obstacles to experimentation were barely featured in the interviewees' commentaries. There were only three cases in which technical concerns restricted experimentation or had done so in the past. Moreover, these concerns appeared to be primarily linked to insufficient resources rather than insurmountable technical problems.

4.4 B2B-Specific Challenges of Continuous Experimentation

In addition to the domain-independent issues discussed above, challenges rather specific to the business-to-business (B2B) domain emerged. These were the customers' organizational culture (five occurrences in the interviews), access to end users (four occurrences), and consent to product usage data collection (three occurrences).

Many B2B company representatives considered their customers' organizational culture an obstacle to experimentation. For instance, customers were not always able to give feedback or participate in development and experiments: *"[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." (senior manager)* Lack of time was the main supposed reason for the customers' disinclination to participate more.

A second obstacle was limited access to end users. Some interviewees considered that improving product usage data collection would alleviate the above-mentioned challenges. However, as the third identified challenge indicates, customer consent to product usage data collection could not be taken for granted: *"[I]t might be difficult to get some of the customers to agree that we can monitor their users and what they do." (middle manager)*

All three aspects, i.e., customers' cultural frameworks, limited access to end users, and unavailable consent to product usage data collection might also apply to business-to-customer (B2C) companies. However, the level of control over these aspects is usually lower in B2C situations.

4.5 Success Factors of Continuous Experimentation

Another objective of this study was to gain insights into the success factors of experimentation. Since systematic, continuous experimentation in itself was rare, the interviewees were invited to reflect on their companies' strengths with respect to customer involvement and their methods of collecting and utilizing customer feedback. Nevertheless, noticeably fewer success factors than challenges were uncovered in the interviews. Furthermore, the identified success factors appeared to be domain-independent since no clear domain divisions arose in the data, as opposed to the challenges. Table 3 gives an overview of the success factors.

The good availability of technical tools and the perception of technical competence were the most often recognized success factors. Interviewees thought that the tools exist for embracing experimentation, even if they were not all actively used due to the challenges discussed in the previous sections: *"We have the tools. […] We tried it [a particular experimentation tool] […] once and it looked good." (middle manager)* Interviewees also appeared to trust the capabilities of their companies to solve any technical matters related to experimentation: *"[I]t feels like we can get the technical issues sorted out." (senior manager*) It might appear contradictory that technical competence was seen as an important success factor but technology only as a minor obstacle. On the one hand, the interviewees emphasized that the existence of tools or a tool infrastructure that allows for experimentation is an essential prerequisite for continuous experimentation. On the other

hand, installing the tool infrastructure requires tools and technical competence but it is not a major obstacle.

Table 3. Success factors and frequency of occurrence.

Success factor	**Freq. of occurrence by company**
Technical tools	8
Technical competence	6
Organizational culture	6
Customer and domain knowledge	5
Release cycle speed	1

A positive organizational culture was a recurrent theme in the context of success factors. The positive evaluations were typically made by the representatives of small companies, in particular the startup representatives. One senior manager specified his company's key strength with respect to experimentation: *"That we do it! That we understand that we should do it."* 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.'"*

Other interviewees also stressed the importance of cultivating an open organizational culture in which the whole team is involved in discussing the direction of product development: *"[E]verybody [..][is] free to tell [us] what they think." (senior manager)* In addition, the proper alignment of employees' authorities and responsibilities was mentioned. The participants considered an empowered organizational culture 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." (senior manager)*

One middle manager reflected on the common nominators of established companies that have succeeded in transforming their organizational culture to an experimental mindset: *"There are people who are rebellio[u]s. They don't obey this traditional way of working. They are strong enough to do something different."* He also stressed that successful companies are organized in a way that supports and respects experimentation at 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." This viewpoint parallels the aforementioned findings regarding the active involvement and empowerment of the product development team.

Finally, interviewees considered practical support functions on an organizational level to be beneficial for endorsing experimentation. These support functions 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 […][with] that. Which […] is a huge asset." (middle manager)*

As regards other findings, the importance of extensive customer and domain knowledge was another frequent theme. As one interviewee said: *"I would say we have good connection to customers […]. We know our customers." (middle manager)* In some cases, this 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, interviewees saw expertise in the domain of operation as important: *"People understand […] what it's about." (senior manager)*

Achieving a rapid release cycle was one of the key challenges of experimentation (Section 4.3). For one company, a swift release cycle facilitated experimentation and 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 […] one year. We can do it in three months." (middle manager)*

5 Discussion

This section puts key findings from the study in perspective to expectations and related findings in literature. In addition, it discusses potential implications for practice and research.

5.1 Relating the Findings to Expectations and Other Studies

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 goals of continuous experimentation resonated well within the software industry: there was a wish to focus on customer value creation and data-driven decision making. Although the majority of companies involved in the study had not yet reached the stage of continuous experimentation, many were interested in the concept. It is noteworthy that in contrast to the expectations of the authors, the companies understood the purpose of the concept relatively easily. Many of the contributing companies' current practices could be seen as supportive of experimentation: agile development was prevalent, continuous integration was utilized, and release cycles were reasonably short. Companies were attempting to further shorten release cycles for instance by focusing on key software capabilities – a goal which experimentation may help to achieve.

The companies collected a wide range of direct customer feedback, but the collection of product usage data was not ubiquitous and was often hampered by insufficient product instrumentation. The findings match related studies showing that a wide array of customer feedback collection techniques is available and already used in industry (for examples, see [36]). Furthermore, 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 [37, 38], suggesting that there is untapped learning potential in product usage data.

The present study found experimentation to be systematic and continuous in only one startup company. The reason for this might be that the continuous experimentation model is derived from a startup environment [16], and startup companies can therefore be expected to be among its early adopters. It may also be easier to adopt experimental practices in a fledgling company. Although several other companies had some experience with A/B testing, or expressed an interest in it, the

findings suggest that experimentation as a constant part of an overall development process is not yet widely established in the software industry. In consequence, decision making with respect to strategy, product vision, and development is not yet linked closely to experimental results in many companies. Such a missing link is also reported in other studies. Holmström Olsson and Bosch [21], for instance, observed in many companies an ‘open loop’ between customer feedback and product management decisions. A multi-case study conducted by Sauvola et al. [18] showed that all of the examined case companies front-load some customer involvement during the requirements phase but leave most of the customer validation until after the software or one of its features is released.

The second study goal 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. The reasons for this may include the rarity of systematic experimentation in the participating companies. It may also be easier for participants to recognize 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, and hence they are discussed together.

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. 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 also relate to the issue. Due to these deficiencies, the customer feedback loop may not function optimally. 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.

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 [14]. The support was secured by demonstrating how the experiments helped to avoid wasting R&D resources on valueless features or products. Kohavi et al. [39] 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 [40]. 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 [5]. By enabling the early and continuous prioritization of work according to customer value, these measures may also help in accelerating the release cycle. This was a highly recurrent goal of the companies in the present study.

In order to measure customer value, appropriate 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 emphasizes the need for actionable metrics, and examples of such metrics are presented for instance in [20, 41]. On a related note, Bosch et al. [42] 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 [43] provides steps for linking software development and business strategy through 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 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 retention and thus ensure the sustainability of the underlying business model in the long term.

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. The availability of technical tools and competence were repeatedly identified as success factors, while technical obstacles were rarely reported. 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 in this study predominantly represented managerial roles, this finding must be interpreted with caution.

Finally, the current study found that operating in a business-to-business (B2B) domain signified additional challenges with respect to experimentation. In this environment, value for users might not directly translate into value for customers. Previous research [44] supports the finding that B2B imposes specific challenges and identifies several additional challenges that are especially pronounced in the B2B domain.

To summarize, the study presented a state of the practice analysis on the participating companies' experiences with experimentation. It aimed at giving an impression of the practices and procedures which companies currently apply in relation to experimentation, and how these activities are typically motivated and linked. Although many isolated findings of the study are confirmed by other studies, this study revealed in a more comprehensive way where links between activities (e.g., collecting product usage data and using the data in decision making) or activities themselves (e.g., experiments with customers) are typically missing.

5.2 Implications for Practice

The findings from the present study reveal several activities that companies can conduct when transitioning towards a systematic experiment-driven development approach. Based on the data from this study and findings from related work we extract the following key takeaways for organizations that aim at establishing an experimental approach to product development:

Key takeaways

- Expose deliverables to potential customers or users early and frequently in order to elicit feedback.
- Collect product usage data in order to understand customer value.
- Complement quantitative analyses for eliciting customer feedback with qualitative analyses for clarification, interpretation and exploration of areas outside the initial focus.
- Involve different stakeholders (such as business, development, product management, and customers) in hypothesis generation and in the analyses of customer feedback.
- Use results from experiments as input for data-driven decision making.
- Internalize values and principles (such as fast learning, constant questioning of product assumptions, experimental mindset) that are relevant for continuous experimentation.

- Provide sufficient resources and establish necessary competencies, practical support functions, and the necessary tool infrastructure for continuous experimentation.

A challenging question is how to master the transition in practice. Based on the study findings this transition depends especially on the size of the organization (e.g., a startup vs. a large enterprise) and the type of business transactions (e.g., B2B, B2C). However, we assume that there are many other criteria that have an impact on the transition such as the business model, the technical domain, or the market situation.

Currently there is very little experience available on how to conduct this transition. One example is the "Stairway to Heaven" model [19] that describes the staged evolution of company practices. The last stage of this model can be characterized as an experiment-driven R&D approach.

Other related approaches focus on the lean and agile transformation of organizations. An example of lean transformation is the "Lean Enterprise" approach by Humble et al. [45]. In this approach, different aspects including innovation culture, governance, risk, compliance, and financial management are considered.

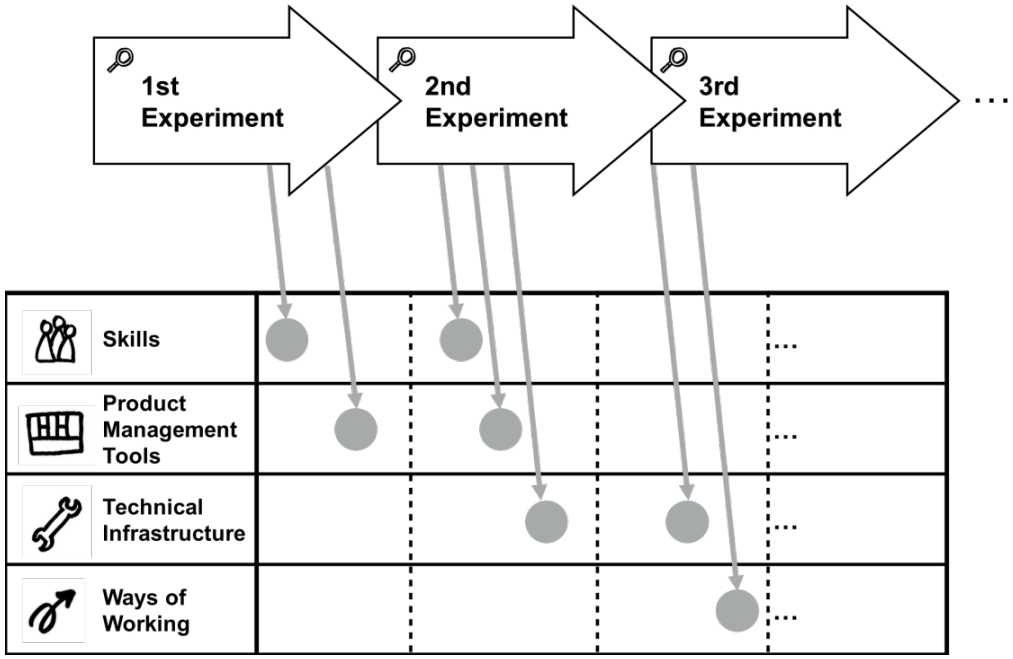
The findings from the study described in this article indicate that most large companies start small, for instance by conducting A/B tests and integrating these tests into their development practices. This matches the authors' experiences from other projects. As indicated by the study findings and supported by project experiences of the authors, young and small companies like startups are in a different situation and can more easily start with a broad adoption of the experimental approach right from the beginning.

Nevertheless, the question of how large organizations could start the transition remains. Fig. 2 portrays a potential incremental transition path, beginning with establishing first data analysis and product management capabilities. These could be applied, for instance, when conducting the first qualitative experiments (e.g., conducting problem interviews and defining a product vision). With future experiments further relevant topics could be addressed. This proposal, although based on experiences of the authors from projects in large organizations, is not meant as a prescriptive model and requires adoption to the company context. For instance, if an organization has already established significant skills and an appropriate product management framework, it might be better to begin with setting up the infrastructure first.

Regardless, the pattern as described in Fig. 2 captures the following considerations: even without product management, infrastructure, or changed ways of working it is possible to conduct small

experiments. For designing the first experiments and analyzing the results, basic skills are required. Therefore we recommend starting with establishing the basic skills on how to conduct experiments (e.g., skills for conducting and analyzing customer interviews). Appropriate product management helps to define and prioritize relevant hypotheses for experiments in a systematic way. Consequently we recommend introducing initial product management tools (e.g., a product vision board, a validation board) early in the transition process. Without a technical infrastructure (e.g., feedback channels, data analysis tools, data storage capabilities) only rather simple experiments can be executed. Building up an appropriate infrastructure can therefore be seen as a next logical step. Finally, without changing ways of working (e.g., changing processes for making product decisions) and potentially also organizational structures (e.g., establishing cross-functional product teams) the experimental approach to product development will likely not be sustainable. Overall, this model should be seen as an example and other transition paths might be appropriate depending on the company context.

Fig. 2. Transitioning towards experiment-driven development.



5.3 Implications for Research

The study addresses many avenues for future research. Potential directions include building systems for eliciting and analyzing customer feedback, integrating experimentation into the development process, managing organizational change towards an experimental development mode, developing tool-based infrastructures for experimentation, customizing the experimental approach to different domains, and linking experimentation with strategy definition. Further open issues and directions

can be found in the cited literature, e.g., Huomo et al. [46] outline research directions in the field and provide a comprehensive list of related open research questions.

6 Conclusions

This paper presented a qualitative survey on companies' experiences of experimentation in software product development. The study found that while many of the current development practices supported experimentation, the state of the practice is not yet mature. Although a broad array of techniques was employed to collect customer feedback, systematic experiments with customers are rare. Moreover, many companies do not use product usage data to learn about customer needs, and product instrumentation is often inadequate. Finally, the collaboration between the R&D organization, product management, and customers sometimes appears insufficient for supporting an innovative, experimental approach.

Key challenges in embracing experimentation are 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 is carefully analyzed by relevant stakeholders. Adequate resources also need to be secured. Business-to-business companies face additional challenges.

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

Acknowledgements. We wish to thank the participants in the study for their time and contributions and the reviewers for their valuable comments. We would also like to thank the Finnish technology agency, Tekes, for funding the Cloud Software Factory project, and the Need for Speed program, under which the proposed study was undertaken.

References

[1] Wasserman, A.I. Low Ceremony Processes for Short Lifecycle Projects. In: Kuhrmann, M. et al. (ed.): Managing Software Process Evolution: Traditional, Agile, and Beyond. How to Handle Process Change? Pp. (in press). Springer International Publishing, 2016.

[2] 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, 2013.

[3] Giardino, C. et al. Key Challenges in Early-Stage Software Startups. In: Lassenius, C., Dingsøyr, T. and Paasivaara, M. (eds.): Agile Processes, in Software Engineering, and Extreme Programming, pp. 52-63. Springer International Publishing, 2015.

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

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

[6] 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.

[7] 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, 2010.

[8] Lindgren, E. and Münch, J. Software Development as an Experiment System: A Qualitative Survey on the State of the Practice. In: Lassenius, C., Dingsøyr, T. and Paasivaara, M. (eds.): Agile Processes, in Software Engineering, and Extreme Programming, pp. 117-128. Springer International Publishing, 2015.

[9] Knapp, J., Zeratsky, J. and Kowitz, B. Sprint: How to Solve Big Problems and Test New Ideas in Just Five Days. Simon & Schuster, New York, 2016.

[10] Papatheocharous, E. and Andreou, A.S. Empirical evidence and state of practice of software agile teams. *Journal of Software: Evolution and Process,* 26, 9, pp. 855-866, 2014.

[11] Fitzgerald, B. and Stol, K. Continuous Software Engineering: A Roadmap and Agenda. *Journal of Systems and Software,* pp. (online), 2015.

[12] Beck, K. et al. Principles behind the Agile Manifesto. http://agilemanifesto.org/principles.html [2016/2/22].

[13] 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, 2013.

[14] 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, 2013.

[15] 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, 2013.

[16] 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, 2014.

[17] Karvonen, T. et al. Hitting the Target: Practices for Moving Toward Innovation Experiment Systems. In: Fernandes, J.M., Machado, R.J. and Wnuk, K. (eds.): Software Business, pp. 117-131. Springer International Publishing, 2015.

[18] Sauvola, T. et al. Towards Customer-Centric Software Development: A Multiple-Case Study. In: Proceedings of the 41st EUROMICRO Conference on Software Engineering and Advanced Applications (SEAA), pp. 9-17. IEEE Press, 2015.

[19] 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: Proceedings of the 38th EUROMICRO Conference on Software Engineering and Advanced Applications (SEAA), pp. 392-399. IEEE Press, 2012.

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

[21] Holmström Olsson, H. and Bosch, J. From Opinions to Data-Driven Software R&D: A Multi-case Study on How to Close the 'Open Loop' Problem. In: Proceedings of the 40th EUROMICRO Conference on Software Engineering and Advanced Applications (SEAA), pp. 9-16. IEEE Press, 2014.

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

[23] Alvarez, C. Lean Customer Development: Building Products Your Customers Will Buy. O'Reilly Media, Sebastopol, CA, 2014.

[24] Gothelf, J. and Seiden, J. Lean UX: Applying Lean Principles to Improve User Experience. O'Reilly Media, Beijing, 2013.

[25] 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.

[26] 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.

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

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

[29] Lindgren, E. and Münch, J. Interview guide and codebook for the paper “Software Development as an Experiment System”. http://dx.doi.org/10.6084/m9.figshare.1254619 [2015/01/01].

[30] Need for Speed (N4S) Research Program. http://www.n4s.fi [2014/12/07].

[31] 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.

[32] ATLAS.ti Scientific Software Development GmbH. http://www.atlasti.com [2014/12/07].

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

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

[35] The Business Dictionary. Organizational culture. http://www.businessdictionary.com/definition/organizational-culture.html [2015/11/08].

[36] Fabijan, A., Holmström Olsson, H. and Bosch, J. Customer Feedback and Data Collection Techniques in Software R&D: A Literature Review. In: Fernandes, J.M., Machado, R.J. and Wnuk, K. (eds.): Software Business, pp. 139-153. Springer International Publishing, 2015.

[37] 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, 2013.

[38] 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, 2013.

[39] 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].

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

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

[42] 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, 2013.

[43] 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, 2007.

[44] Rissanen, O. and Münch, J. Continuous Experimentation in the B2B Domain: A Case Study. In: Proceedings of the 2nd International Workshop on Rapid Continuous Software Engineering (RCoSE 2015), Florence, Italy, 2015.

[45] Humble, J., Molesky, J. and O'Reilly, B. Lean Enterprise: How High Performance Organizations Innovate at Scale. O'Reilly Media, Sebastopol, CA, 2014.

[46] Huomo, T. et al. Need for Speed (N4S) Research Program. Strategic Research Agenda. https://www.researchgate.net/publication/258860684_Strategic_Research_Agenda_For_Need_For_Speed [2015/11/08].

Appendix 1. 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 customer value elicitation and utilization practices.

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 behavior, 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 customer value elicitation and utilization practices.

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 collecting and using deeper customer insights?
    (Possible prompts: technical issues, lack of resources, personnel skills, company culture etc.)

20. What are the factors that support collecting and using customer insights in your company?
    (Possible prompts: technical know-how, ample resources, personnel skills, company culture etc.)

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 further comments or questions related to this interview or the study in general?

Thank you for your time and contribution to the study!