Data Driven Development: Challenges in Online, Embedded and On-Premise Software

Helena Holmström Olsson[1(✉)] and Jan Bosch[2]

[1] Department of Computer Science and Media Technology, Malmö University, Malmö, Sweden
helena.holmstrom.olsson@mau.se
[2] Department of Computer Science and Engineering, Chalmers University of Technology, Gothenburg, Sweden
jan.bosch@chalmers.se

Abstract. For more than a decade, data driven development has attracted attention as one of the most powerful means to improve effectiveness and ensure value delivery to customers. In online companies, controlled experimentation is the primary technique to measure how customers respond to variants of deployed software. In B2B companies, an interest for data driven development is rapidly emerging and experiments are run on selected instances of the system or as comparisons of previously computed data to ensure quality, improve configurations and explore new value propositions. Although the adoption of data driven development is challenging in general, it is especially so for embedded systems companies and for companies developing on-premise software solutions. Due to complex systems with hardware dependencies, safety-critical functionality and strict regulations, these companies have longer development cycles, less frequent deployments and limited access to data. In this paper, and based on multi-case study research, we explore the specific challenges that embedded systems companies and companies developing on-premise solutions experience when adopting data driven development practices. The contribution of the paper is two-fold. First, we provide empirical evidence in which we identify the key challenges that embedded systems and on-premise software solutions companies experience as they evolve through the process of adopting data driven development practices. Second, we define the key focus areas that these companies need to address for evolving their data driven development adoption process.

Keywords: Data driven development · Online software · Embedded systems · On-premise solutions · Adoption process · Challenges

1 Introduction

Over the past years, software-intensive companies in a variety of domains, with online companies leading the way, have started adopting data driven development practices to continuously assess customer value and monitor feature usage [1–3]. Using the

X. Franch et al. (Eds.): PROFES 2019, LNCS 11915, pp. 515–527, 2019.
https://doi.org/10.1007/978-3-030-35333-9_36

definition provided by [4], data driven development is the *ability of a company to acquire, process, and leverage data in order to create efficiencies, iterate and develop new products, and navigate the competitive landscape.* In recent studies, data driven development practices are proven useful for improving product performance, for optimizing system parameters and for evaluating new product concepts [5–8]. As a result, companies that are adept at acquiring, processing and leveraging customer and product data become more profitable as continuous assessment of customer value can have a profound impact on annual revenue [8]. As an additional benefit, data can help question, challenge, complement and confirm existing assumptions in the organization. In this way, collection and use of data is becoming an effective mechanism for replacing opinions-based decision-making with data-driven decision-making about customer value, system performance and overall product quality [2]. While the opportunities provided by data are already well-established in online companies, they are becoming increasingly recognized also in companies developing on-premise solutions and embedded systems. With products such as cars, trucks, phones, cameras, household appliances etc. being increasingly software-intensive and connected to the Internet, these companies are starting to explore the opportunities that online companies have benefitted from for more than a decade [9]. However, although there are examples of data driven development practices being used in embedded systems and on-premise companies, the adoption process of these practices is challenging. Typically, and due to complex systems with hardware dependencies, safety-critical functionality and strict regulations, these companies have longer development cycles, less frequent deployments and limited access to customer and product data.

In this paper, and based on multi-case study research, we explore the specific challenges that embedded systems companies and companies developing on-premise solutions experience when adopting data driven development practices. To achieve this, we first review contemporary literature on data driven development in online companies where these practices are fully adopted and successfully used, and we identify the typical stages these companies evolve through when adopting these practices. Second, and with the adoption stages from the online companies as a basis, we study a total of nine companies in the embedded systems and in the on-premise software domain with the intention to understand the specific challenges these companies experience when adopting the similar practices as the online companies.

The contribution of the paper is two-fold. First, we provide empirical evidence in which we identify the key challenges that embedded systems and on-premise software solutions companies experience as they evolve through the process of adopting data driven development practices. Second, we define the key focus areas that these companies need to address for further evolve their data driven development practices.

The remainder of the paper is organized as follows. In Sect. 2, we review contemporary literature on data driven development in online companies and we identify the typical stages that online companies evolve through when adopting data driven development. In Sect. 3, we describe the research method and the case companies. In Sect. 4, we present our empirical findings. In Sect. 5, we identify the key challenges that embedded systems and on-premise solutions companies experience when adopting data driven development and we define the key focus areas that these companies need to address to further evolve these practices. In Sect. 6, we conclude the paper.

2 Background

In this section, we review contemporary literature on the adoption of data driven development in online companies. We define online companies as companies providing web services and that use controlled experiments to determine which variant of a product, design or interface that performs the best. In recent studies, companies such as e.g. Facebook, Google, Booking, Amazon, LinkedIn and Skyscanner are often referred to in relation to successful use of controlled experimentation [6, 10].

2.1 Data-Driven Development

For decades, one of the primary challenges in software development has been how to shorten feedback cycles to customers [11, 12]. As outlined in previous research [13], the first step towards shorter feedback cycles is the adoption of agile development. These methods emphasize short iterations of increments rather than the long cycles as known from traditional development. More recently, technologies such as continuous integration [14] and continuous deployment [15] have enabled companies to further shorten feedback cycles. These technologies allow for frequent test and deployment of software and in combination with connectivity that enables diagnostic, performance and operations data to be collected, companies can significantly shorten the time it takes to learn from and respond to customers.

In online companies, data driven development is a well-established approach to software development [3, 8, 11, 16]. In these companies, data is the foundation for any decision regarding redesign or improvement of a feature, for prioritization of features from the backlog and for optimization of certain metrics. With techniques such as A/B testing and automated practices for data collection and analysis, customers are continuously part of experiments to help optimize the system and queries are processed frequently to provide software developers and managers with rapid feedback [5]. As recognized in our previous research [17], companies that adopt data driven development typically do this by starting to identify what key factors to optimize for. This is achieved by modeling the expected value of a feature in order to get a few metrics in place to then collect data that will help improve these. In online companies, common metrics are e.g. *'number of users', 'frequency of use', 'response time', 'number of successful upsells'*. In addition to identifying metrics, teams also need to identify the relative priority of these factors. This is important as some factors may improve while others decline when running an experiment. Data driven development reflects a shift from traditional development where requirements inform development [18], towards a situation in which continuous collection of data inform development throughout the lifecycle of the system [2, 5, 19]. Moreover, and as experienced in online companies, data-driven development constitutes an effective means to challenge existing assumptions held by people in the organization. Often, inaccurate assumptions result in poor decision-making, an inaccurate understanding of customer value and slow feedback cycles. As a consequence, companies end up investing development efforts in features that are not used by customers and optimizing for metrics that are no longer representative for what generates business value.

2.2 Experimentation Practices

As a critical technique in data driven development, online controlled experimentation, also known as A/B testing, allows continuous validation of value with customers [16]. Online controlled experiments constitute a practice of comparing two versions of functionality to determine which one performs better in relation to predefined criteria such as e.g. conversion rate, click rate or time to perform a certain task. In online companies, controlled experiments are the norm with companies such as e.g. Amazon, eBay, Facebook, Google and Microsoft running hundreds and even thousands of parallel experiments to evaluate and improve their services at any point in time. To achieve this, companies need an infrastructure to collect and store data from deployed products and that makes data available for analysis. There are numerous experimentation tools and platforms available on the market [6, 20]. However, the challenges of building the data infrastructure are typically not concerned with the basic technologies but rather with aspects related to customer relations, legal constraints, cost of data collection and storage. It should be noted that experimentation involves many different techniques. For example, experimentation could refer to iterations with prototypes in the startup domain, canary flying of software features, gradual rollout and dark launches [11]. With frequent experimentation, teams can adopt an increasingly iterative development approach in which features are sliced into smaller parts that can be developed in less than a sprint and for which the team collects data to guide the next steps of development. As recognized in [2], this allows teams to rapidly determine whether a feature adds value or not. In our previous research, and based on studying a large number of online experiments at Microsoft, we introduced the Experiment Lifecycle in which we outline the three main stages of every Online Controlled Experiment [11].

During recent years, online controlled experimentation has received increasing interest and there exist a number of studies describing the many benefits with this practice [8, 11, 16, 20, 21]. These studies outline the roles involved (e.g. data analysts, data scientists, product managers, software developers etc.), the task at hand (e.g. development of roadmaps, design and analysis of experiments, development of products, deployment of products etc.) and the technical infrastructure that is the platform for the experiments (e.g. the application programming interfaces, experiment databases, analytic tools, instrumentation, integration and deployment systems etc.). In particular, challenges in relation to the definition of an 'Overall Evaluation Criterion' have been carefully explored [16] as well as models that describe the experiment lifecycle in online companies [2, 8], the data collection techniques that are used [22], and the infrastructure that is required for running a successful online controlled experiment [16, 23].

2.3 Team – System – Business Metrics

As a prerequisite for an experiment, teams need to define an 'Overall Evaluation Criterion' (OEC) [11, 16]. The OEC is a structured set of metrics consisting of success, guardrail and data quality metrics that are used to define performance goals and desired outcomes of an experiment. For teams, an OEC can consist of improving conversion rate on a website, increase throughput or improve a specific feature. At a system or

product level, the OECs cover system and product performance and metrics are used to track the overall product portfolio. At the highest level, business metrics are defined to track overall business goals. While metrics at the team level are leading indicators that teams can influence on a daily basis, business metrics are lagging indicators that are hard to influence in the short term but instead metrics that change over a longer period of time. In previous research [11], and based on our insights from working with four companies in the online domain, we introduced a framework for how to scale experimentation and in which the definition of OECs at the team, at the system and at the business level are critical elements. The goal is to have efforts at the team level positively influence business metrics [17]. If so, companies can effectively scale experimentation, advance their data driven development practices and successfully use data as the basis for decision-making throughout the organization.

2.4 Data Driven Development Adoption Process

Based on the learnings from our literature review, as well as from on our own experiences when studying companies in the online domain, we have identified five stages that we see online companies evolve through when adopting data driven development (Fig. 1).

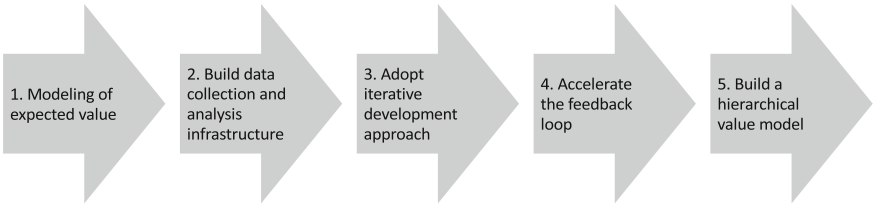

Fig. 1. Data driven development adoption process: the five stages we see online companies evolve through when adopting data driven development. The model is derived from previous literature as well as from our own experiences when studying online companies.

The first stage online companies enter when adopting data driven development is to have development teams identify what factors to optimize for. This is achieved by *modeling the expected value* of a new or existing feature and works as the basic stage in order to get a few metrics in place to then collect data to help improve these. These factors are used to guide experimentation and to track the performance of subsequent releases of the feature. In the second stage, companies develop an infrastructure to *collect and store data* from deployed products and that makes data available for analysis. The third stage is concerned with increasing the effectiveness of development teams by *adopting an iterative development approach*. In this approach, features are sliced into smaller parts that can be developed in less than a sprint and for which the team collects data to guide the next steps of development. In the fourth stage, companies seek to further *accelerate the feedback loop*. To achieve this, they develop the shortest possible cycle between development of a feature and deployment in the field. In online companies, the feedback loop ranges from hours to minutes and even seconds

and as a result, these companies are able to use data to effectively direct their development efforts. As the final stage, and as the mechanism to ensure alignment between team, system and business level metrics, companies develop a *hierarchical value model* where feature level metrics that are modeled as part of the first stage of the process are connected and aligned with high-level business key performance indicators (KPIs).

3 Research Method

The goal of this study is to explore the challenges that embedded systems companies and companies developing on-premise solutions experience when adopting data driven development. In our study, embedded systems companies are companies that develop larger systems and complete devices including hardware and mechanical parts and in which software is one part [24]. On-premise software is software that is installed and runs on the premises of the organization using the software, rather than at a remote facility such as the cloud [25]. Our study builds on multi-case study research in companies from these two different domains as well as on our previous learnings from the online domain. Case study research focuses on providing a deeper understanding of a particular context and it emphasizes the importance of peoples' experiences [26]. In our study, and as a first step, we reviewed contemporary literature on the adoption of data driven development practices in online companies. In addition, we built on our own experience from working with companies in the online domain and as reported in [1–3, 8, 11, 13, 16, 17, 22, 23]. Based on this, we engaged with nine companies in the embedded systems and on-premise domain to understand the challenges these companies experience when adopting the similar process. The case companies (Table 1) were at different maturity levels in the adoption process of data driven development. At the time of our study, the practices in company E, H and I reflected the initial stages of the process, company C, D, F, G and A were approaching or at the middle stages and company B was aiming for a hierarchical value model. During our study, we engaged in workshop sessions at each company in which we facilitated, as well as documented, their experiences with the different stages in the adoption process. At each company we had developers, product managers, technical specialists, software architects, system engineers, agile coaches and data scientists present. Each workshop session involved between 6–10 people and lasted for 3-5 h. In companies where data driven practices were immature, we had larger groups of 15–20 people as the workshops served the additional purpose of introducing the organization to the concept. Our study involved nine companies (Table 1).

In the continuation of the paper, we provide a summary of the experiences from the case companies in order to establish an understanding for the specific challenges these companies experience when adopting data driven development. For validity of results [24], and to address construct validity, we started each workshop with sharing our definition of the key concepts. This established a common understanding of the topic and we could discuss alternative interpretations already before we ran into potential misunderstandings. With respect to external validity, our contributions provide rich insight in different company domains and we identify implications for research and for practice.

Table 1. The case companies and the domain(s) they operate in.

Case	Description	Embedded systems	On-premise
A	Provider of systems and equipment for network operators	x	x
B	Developer of navigational information and optimization solutions	x	x
C	Developer of network video surveillance solutions	x	x
D	Developer of food packaging and processing systems	x	
E	Provider of systems and solutions for military defense and civil security	x	
F	Developer of automotive technology	x	
G	Engineering and electronics company	x	
H	Manufacturer of vehicles	x	
I	Manufacturer of trucks, buses and construction equipment	x	

4 Findings

Below, we summarize the key findings from our study. When reporting on our findings, we use the five stages of adopting data driven development that we identified in our literature review.

Stage 1: Modeling of Feature Value

To introduce the embedded systems and on-premise software companies to the first stage of adopting data driven development, we initiated a series of workshop sessions in which we met with developers and product managers in order to model the value of a selected feature. As part of the workshops, the teams selected a feature, identified key value factors, prioritized these factors and their relative importance. In the end, a few groups managed to develop a value function to quantitatively express what they optimize for. While the majority of the companies selected existing features to work with we also had companies that used the workshops to model new features that were not yet developed and for which value was not yet proven. In company B, one of the teams succeeded in developing a complete value function for one of their mobile applications. They expressed it as: *0.1*feedback time + 0.2*success rate + 0.2*number of users + 0.2 successful drops – 0.3 cost of ownership* where each value factor was given a relative weight and where the formula indicates whether you look to increase or decrease the value of each factor.

Stage 2: Build Data Collection and Analysis Infrastructure

In the second stage, the companies realized the need for a data collection and analysis infrastructure. As experienced in these companies, the initial focus should be on keeping things simple by collecting data only for the selected feature and only from friendly customers as this allows easier access to data. As the companies had infrastructures for data collection in place already, this stage was mostly concerned with

complementing these with metrics that would allow for measuring value according to the new value function. As a common experience, this stage revealed lack of effective analysis tools and often the approach was manual solutions and/or existing web-based solutions.

Stage 3: Adopt Iterative Development Process
As most of the case companies have a hardware and mechatronics background, the adoption of iterative development required a significant change in mind-set. For the companies, the identification of parts of their organizations where these new ways-of-working were feasible was an important step and typically, they selected already agile teams within their software organization. With these teams, we developed hypotheses that could be tested during each sprint and we ran validation workshops to evaluate experiments. Most companies were able to identify 1–4 hypotheses to test during the next 2–3 sprints and with the goal to either (1) increase the number of hypotheses to be tested within their current sprints, or to (2) shorten their current sprints to increase the total number of hypotheses tested.

Stage 4: Accelerate the Feedback Loop
All companies have a tradition in traditional development and they have adopted agile development. To further accelerate the feedback loop, they have started adopting continuous integration and continuous deployment. However, until these practices are fully in place, it is difficult to further accelerate the feedback loop. In all companies, huge efforts were put in place to drive CI and CD initiatives as well as to minimize customer-specific branches of a product and instead strive for a single product branch with configuration opportunities for different customers. In this way, feedback loops were shortened and the companies could benefit from frequent releases.

Stage 5: Build a Hierarchical Value Model
The last stage is to build a hierarchical value model to ensure that team metrics and business metrics align. While online companies have a complete hierarchy of metrics at team – system – business level, the establishment of such a hierarchy proved challenging in the embedded and on-premise companies. While a well-defined set of metrics, such as e.g. customer satisfaction, revenue, sales, customer retention, net promoter score etc., existed at the business level, these did not necessarily translate into executable metrics for teams to optimize for. In the case companies, we noted a willingness to establish a hierarchical value model and we started aligning metrics in a couple of the companies. However, as the previous four stages have to be in place in order to successfully create a value model for the entire business, we did not achieve this within the time span of this study.

5 Key Challenges When Adopting Data Driven Development

The intention with our study was to explore the specific challenges that embedded and on-premise software companies experience when adopting data driven development. Below, we identify the key challenges that these companies experience as they evolve through the process of adopting data driven development.

Stage 1: Modeling of Feature Value
Based on our experiences, the first stage comes with at least four challenges:

Difficulties in agreeing on value factors and the relative priority of these: The workshops revealed that it is very challenging for a team to agree on the relevant factors and the relative priority of these. This stage surfaced deeply held beliefs about the system and its customers that were far from agreed upon among teams. And as the typical development cycles in the companies were long, and with few opportunities for customer feedback, the assumptions that evolve were rarely questioned. This made improvement efforts difficult as there was no shared understanding on what metrics to optimize for.
Painful quantification of value: Especially product managers were reluctant to explain their reasoning behind prioritizing a certain feature and to quantify the expected value of this feature. Instead, value was described in qualitative terms which made prioritizations easier to defend as quantitative metrics did not exist.
Lack of end-to-end understanding of value: Even if a team agreed on the relevant factors and their relative priority, the relationship between the value of the feature and the business impact proved hard to define.
Illusion of alignment: By abstracting topics of contention to a level of vagueness that everyone could agree on, teams in all companies created a false sense of unity. We interpreted this as a way to avoid tension as to get precise might upset the existing illusion of alignment.

Stage 2: Build Data Collection and Analysis Infrastructure
In the second stage, the case companies experienced an increasing organizational resistance against the adoption of data driven development. Often, people used excuses centered around the customer:

"Don't go data driven because customers don't want to": Non-software people raised the concern that customers don't want to share data and that adopting data driven development would be to go against the interests of these customers.
"Don't go data driven because it is risky": Security, safety and reliability issues were brought forward as reasons to not adopt data driven development.
"Don't go data driven because it is expensive and effort-consuming": A common belief was that to iteratively develop a smaller slice of a feature is difficult with the standard argument being: *"You can't deploy something half done…"*, and with many people uncertain about the value of an MVF ('minimal viable feature').
"Don't go data driven because you can't have all customers do this": In the companies with a strong background in traditional development, there was a tendency to think that all customers had to be involved at the same time. This mind-set revealed lack of experience with starting small scale and with only a selected set of friendly customers.

Stage 3: Adopt Iterative Development Process
In the third stage, the case companies faced a number of challenges in relation to the adoption of a more iterative development approach:

Stuck in waterfall development: All case companies struggled with adopting shorter development cycles and more frequent deployment of software. Although both the embedded systems and the on-premise software companies had agile practices in place in parts of their organizations, people failed in realizing that iterative development requires a change of mind-set in relation to communication, coordination and control of teams.
One feature versus several small MVFs: The case companies have a strong engineering background and people who pride themselves based on the completeness of a feature. This made it difficult to break a feature into smaller increments and think in terms of a 'minimal viable feature' (MVF) with only slices being developed at a time.
Surfacing hidden misalignment: The case companies experienced situations in which questions were raised on how to develop and test hypotheses. During this stage, people who thought they agreed on something realized that this was not the case. Also, what sounded as an easy hypothesis to test often turned out to cause the teams major difficulties in actually realizing within the scope of a sprint.
Retrospective reinterpretation of data: The companies experienced situations in which people, whenever data conflicted with their beliefs, sought explanations that would make their beliefs still true. This was evident both in development teams and among product managers and reflected low trustworthiness in data.

Stage 4: Accelerate the Feedback Loop
To accelerate the feedback loop in companies that are used to long development cycles involves a number of challenges:

Shortening of QA cycles: It became evident that in order to align with the shortened time between the end of a development sprint and deployment at customer site, it was critical to shorten feedback cycles for quality assurance (QA) This was experienced as very difficult in all case companies.
Changing practices for QA: The companies realized that the QA teams need to change ways-of-working. Especially, test automation practices were identified as a key practice QA needed to apply. For this to happen, there needs to be the willingness to deploy to customer, test post-deployment and roll back if any issues.
Data-driven versus Requirements-driven: In most companies, situations in which data driven practices will co-exist with situations in which regulations and standards specify requirements. Therefore, the capability to select the most suitable approach is important and this challenge surfaced when aspiring to accelerate the feedback loop.

Stage 5: Build a Hierarchical Value Model
Although the companies that we studied didn't reach the stage of building a hierarchical value model, they reached far enough to have people reflect on why it is critical for data driven development. In these discussions, we noted the following challenges:

Involvement of all company functions: To build a hierarchical value model involves all company functions. While people close to development might be more enthusiastic to data driven development this is not necessarily the case in other parts of the organization.
Alignment of metrics: To agree on low and high-level metrics is difficult as it forces the company to start aligning metrics and to establish relationships between these.
Model maintenance and evolution: As experienced in our work with online companies, metrics need to continuously evolve to not inscribe an inaccurate understanding of value. We foresee this as a relevant challenge also in the embedded and on-premise companies.
Anecdotal prioritization of resources: Senior leaders have to abandon anecdotal prioritization of resources and instead use data to prioritize customer requests. This is important at all stages, but even more so in relation to the creation of a hierarchical value model as this model is intended replace assumptions and encourage data driven decision-making.

5.1 Key Focus Areas

In the above sections, we presented the key challenges that the case companies experience when adopting data driven development. When reflecting on these challenges, we identify three key focus areas that we believe these companies need to address to further evolve their data driven development practices (Table 2).

Table 2. Key focus areas that the case companies need to address to further evolve their data driven development practices.

Key focus areas	Description
Organizational resistance	Due to a tradition in hardware, mechanics and electronics, these companies experience significant *organizational resistance.* Although this might be true for any change initiative, it is especially so when adopting software-based practices that have the power to radically question existing assumptions while at the same time fundamentally change the basis for decision-making in an organization with a non-software background and tradition
Data quality and trustworthiness	*Data quality* is challenging as it involves collection, processing, sharing, storing and management of large and distributed data sets. As a result of the high complexity involved, *trustworthiness* is low and people tend to rather lean back on existing assumptions than trust the accuracy and quality of facts revealed in the data
Development cycle time	To shorten *development cycle time* is problematic. Despite modularized architectures and advice on how to combine and evolve cycle times for mechanics, hardware and software, the concept of iterative development and incremental development of features remains an issue

6 Conclusions

In this paper, and based on multi-case study research, we explore the specific challenges that embedded systems companies and companies developing on-premise solutions experience when adopting data driven development. When reflecting on these challenges, we see that there are three key focus areas that these companies need to address to further evolve their data driven development practices. First, due to a tradition in hardware, mechanics and electronics, these companies experience significant *organizational resistance.* Second, *data quality and trustworthiness* are challenging as it involves collection, processing, sharing, storing and management, as well as trust, in data. Finally, to *shorten development cycle time* is problematic in systems with highly complex architectures and dependencies. In future research, we aim to further explore the challenges the case companies encounter as these provide valuable input for the open research challenges in relation to organizational resistance, data quality and trustworthiness and development cycle time.

References

1. Holmström Olsson, H., Bosch, J.: Towards data-driven product development: a multiple case study on post-deployment data usage in software-intensive embedded systems. In: Fitzgerald, B., Conboy, K., Power, K., Valerdi, R., Morgan, L., Stol, K.-J. (eds.) LESS 2013. LNBIP, vol. 167, pp. 152–164. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-44930-7_10
2. Olsson, H.H., 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 EUROMICRO, Software Engineering and Advanced Applications (SEAA), 27–29 August, Verona, Italy (2014)
3. Olsson, H.H., Bosch, J.: Towards evidence-based development: learnings from embedded systems, online games and internet of things. IEEE Softw. **4**(5) (2017)
4. Patil, D.J.: Building Data Science Teams, pp. 1–25. Oreilly, Radar (2011)
5. Bosch, J.: Building products as innovations experiment systems. In: Proceedings of 3rd International Conference on Software Business, 18–20 June, Cambridge, Massachusetts (2012)
6. Kohavi, R., Longbotham, R.: Online controlled experiments and A/B tests. In: Encyclopedia of Machine Learning and Data Mining, no. Ries 2011, pp. 1–11 (2015)
7. Fagerholm, F., Guinea, A.F., Mäenpää, H., Münch, J.: Building blocks for continuous experimentation. In: Proceedings of the 1st International Workshop on Rapid Continuous Software Engineering (RCoSE), pp. 26–35 (2014)
8. Fabijan, A., Dmitriev, P., Olsson, H.H., Bosch J.: The evolution of continuous experimentation in software product development: from data to a data-driven organization at scale. In Proceedings of the 39th International Conference on Software Engineering (ICSE), May 20–28th, Buenos Aires, Argentina (2017)
9. Bosch, J., Eklund, U.: Eternal embedded software: towards innovation experiment systems. In: Margaria, T., Steffen, B. (eds.) ISoLA 2012. LNCS, vol. 7609, pp. 19–31. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-34026-0_3
10. Van Nostrand, R.C.: Design of experiments using the taguchi approach: 16 steps to product and process improvement. Technometrics **44**(3), 289 (2002)

11. Fabijan, A., Dimitriev, P., Vermeer, L., Olsson, H.H., Bosch, J.: Experimentation growth: Evolving trustworthy A/B testing capabilities in oline software companies. J. Softw.: Evol. Process **30**(12), e2113 (2018)
12. Bosch-Sijtsema, P., Bosch, J.: User involvement throughout the innovation process in high-tech industries. J. Prod. Innov. Manag. **32**(5), 793–807 (2015)
13. Olsson, H.H., Alahyari, H., 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, 5–7 September, Cesme, Izmir, Turkey (2012)
14. Ståhl, D., Bosch, J.: Modeling continuous integration practice differences in industry software development. J. Syst. Softw. **87**(1), 48–59 (2014)
15. Humble, J., Farley, D.: Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation. Addison-Wesley, Boston (2010)
16. Fabijan, A., Dmitriev, P., Olsson, H.H., Bosch, J., Vermeer, L., Lewis, D.: Three key checklists and remedies for trustworthy analysis of online controlled experiments at scale. In: Proceedings of 41st International Conference on Software Engineering (ICSE), 25–31 May, Montreal, Canada (2019)
17. Olsson, H.H., Bosch, J.: Make up your mind: towards a comprehensive definition of customer value in large scale software development. CLEI Electron. J. **21**(1) (2018)
18. Pohl, K.: Requirements Engineering: Fundamentals, Principles, and Techniques. Springer, Heidelberg (2010)
19. Ries, E.: The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses. Crown Business, New York (2011)
20. Dmitriev, P., Frasca, B., Gupta, S., Kohavi, R., Vaz, G.: Pitfalls of long term online controlled experiments. In: Proceedings of IEEE International Conference on Big Data (Big Data), pp. 1367–1376 (2016)
21. Xia, T., Bhardwaj, S., Dmitriev, P., Fabijan, A.: Safe velocity: a practical guide to software deployment at scale using controlled rollout. In Proceedings of the 41st International Conference on Software Engineering (ICSE), 25–31 May, Montreal, Canada (2019)
22. Fabijan, A., Olsson, H.H., Bosch, J.: Customer feedback and data collection techniques in software R&D: a literature review. In: Fernandes, J., Machado, R., Wnuk, K. (eds.) Software Business (ICSOB). LNBIP, vol. 210, pp. 139–153. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-19593-3_12
23. Issa Mattos, D., Dmitriev, P., Fabijan, A., Bosch, J., Holmström Olsson, H.: An activity and metric model for online controlled experiments. In: Kuhrmann, M., et al. (eds.) PROFES 2018. LNCS, vol. 11271, pp. 182–198. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-03673-7_14
24. Heath, S.: Embedded Systems Design. EDN Series for Design Engineers, 2nd edn. Newnes, London (2003)
25. https://www.webopedia.com/TERM/O/on-premises.html. Accessed 20 Sept 2019
26. Maxwell, J.A.: Qualitative Research Design: An Interactive Approach, 2nd edn. SAGE Publications, Thousands Oaks (2005)