

JYU DISSERTATIONS 528

---

**Matti Saarikallio**

# Improving Hybrid Software Business

Quality Culture, Cycle-time and  
Multi-team Agile Management

---



UNIVERSITY OF JYVÄSKYLÄ  
FACULTY OF INFORMATION  
TECHNOLOGY

JYU DISSERTATIONS 528

---

**Matti Saarikallio**

# **Improving Hybrid Software Business**

**Quality Culture, Cycle-time and  
Multi-team Agile Management**

Esitetään Jyväskylän yliopiston informaatioteknologian tiedekunnan suostumuksella  
julkisesti tarkastettavaksi Agoran auditoriossa 1  
kesäkuun 4. päivänä 2022 kello 12.

Academic dissertation to be publicly discussed, by permission of  
the Faculty of Information Technology of the University of Jyväskylä,  
in building Agora, Auditorium 1, on June 4, 2022, at 12 o'clock.



JYVÄSKYLÄN YLIOPISTO  
UNIVERSITY OF JYVÄSKYLÄ

JYVÄSKYLÄ 2022

Editors

Marja-Leena Rantalainen

Faculty of Information Technology, University of Jyväskylä

Päivi Vuorio

Open Science Centre, University of Jyväskylä

Copyright © 2022, by University of Jyväskylä

ISBN 978-951-39-9175-3 (PDF)

URN:ISBN:978-951-39-9175-3

ISSN 2489-9003

Permanent link to this publication: <http://urn.fi/URN:ISBN:978-951-39-9175-3>

## ABSTRACT

Saarikallio, Matti

Improving hybrid software business: quality culture, cycle-time and multi-team agile management

Jyväskylä: University of Jyväskylä, 2022, 57 p. + included articles

(JYU Dissertations

ISSN 2489-9003; 528)

ISBN 978-951-39-9175-3 (PDF)

Software delivery organizations are often the heart of new business models that deliver novel competitive advantages. However, when the business model is in place and a strategic advantage has been achieved, there is still room to improve the operational excellence. This two-phase dissertation research investigated how software-producing organizations can be analyzed and operations improved depending on contextual circumstances. Firstly, tools for understanding the context of a software business were explored. Case study to conceptualize the business model's revenue streams was conducted, and startups were compared to established organization. Secondly, design science was employed to construct a cycle-time based metrics framework, and action research interventions improved an agile multi-team software-producing organization. Datasets from semi-structured interviews (n=12, 23, and 41) were collected as well as time-stamp, and quality measures. The findings indicate that an analysis of the established business model from the viewpoint of revenue streams is useful. Differences across types of businesses were unveiled through the business model lens, and an understanding of context was important to aim improvements. Cycle-time metrics analysis was shown to produce actionable improvement ideas, such as promotion of fast customer adoption of new features and release-window redesign. Multi-team organization with hybrid business model had issues with cross-team communication and quality. Issues were fixed through interventions such as joint planning events, visual management improvements, domain team stability, quality culture promotion, and code review enforcement. This resulted in significant reduction in defects and better employee satisfaction. As a result of this research, the empirical understanding increased about how the context influences the recommended improvements. In conclusion, differences in various maturity level businesses, and their business model have an influence on the benefits gained from operational choices and contextual adaptation is key. Still, there might be some generally recommended tactical choices for the software producing organization: ensuring team stability to allow learning, providing adequate communication structures for scaling, adopting cycle-time based metrics for effectiveness, and creating a culture that values quality.

Keywords: hybrid business model, scaled agile, empirical, value, lean startup, cycle-time metrics, SAFe, code review, quality improvement, action research

## TIIVISTELMÄ (ABSTRACT IN FINNISH)

Saarikallio, Matti

Monitahoisen ohjelmistoliiketoiminnan parantaminen: laatukulttuuri, läpimenoajat ja monitiimisen ketteryyden johtaminen

Jyväskylä: Jyväskylän yliopisto, 2022, 57 s. + alkuperäiset artikkelit

(JYU Dissertations

ISSN 2489-9003; 528)

ISBN 978-951-39-9175-3 (PDF)

Innovatiiviset liiketoimintamallit ovat synnyttäneet uudenlaista kilpailuetua. Ohjelmistotuotanto-organisaatiot ovat usein näiden mallien ytimessä. Strategisen edun saavuttamisen jälkeenkin on tilaa parantaa yrityksen toimintaa. Tässä kaksivaiheisessa väitöskirjatyössä tutkittiin miten ohjelmistoja tuottavia organisaatioita voi analysoida ja niiden operatiivista toimintaa parantaa tilannekohtainen toimintaympäristö huomioiden. Ensiksi tutkailtiin liiketoiminnan tilanteen ymmärtämisen työkaluja. Tapaustutkimusten avulla kehitettiin tulovirtojen analysointia ja verrattiin vakiintuneiden ja uusyritysten liiketoimintamallien eroja. Toiseksi suunnittelutiedettä hyödyntäen rakennettiin läpimenoaikapohjainen mittarointiviitekehys ja toimintatutkimuksen interventioilla parannettiin monitiimisen koodituotantotalon toimintaa. Tietoa kerättiin puolistrukturoiduilla haastatteluilla (n=12, 23, 41). Lisäksi koottiin aikaleima- ja laatumittaridataa. Löydöksenä vakiintuneet liiketoimintamallit voivat olla monimutkaisia ja niiden tarkastelu tulovirtojen rakenteen kautta on hyödyllistä. Ohjelmisto-organisaatioissa havaittiin tyyppikohtaisia eroja. Ymmärrys liiketoimintaympäristöstä korostui toiminnan kehittämisen suuntaamisessa. Läpimenoaikamittaristoon perustuva tarkastelu tuotti parannusideoita (esim. asiakaskäyttöönoton jouduttaminen ja tuotantoon vientien ajankohtien muutos). Monitiimisellä hybridiliiketoimintamallia toteuttavalla organisaatiolla oli haasteita laadussa ja tiimien välisessä tiedonvaihdossa joihin interventioina toimi yhteissuunnittelutapahtumat, työn virtauksen visualisointi, liiketoiminta-alueisiin perustuvien tiimien pysyvyyden lisäys, laatua arvostavan kulttuurin edistäminen, ja koodikatselmointien vaatiminen. Havaittiin laatuvirheiden merkittävä vähentyminen ja henkilöstön tyytyväisyyden parantuminen. Tutkimuksen myötä laajeni ymmärrys tilannekohtaisesti tärkeistä parantamiskohteista. Erot ohjelmistoliiketoiminnoissa vaikuttavat operatiivisten valintojen hyödyllisyyteen. Kuitenkin lienee yleisestikin hyviä taktisia valintoja: tue tiimien kohtuullista pysyvyyttä oppimisen varmistamiseksi, rakenna riittävät kommunikaatiomekanismit toiminnan skaalaamiseksi, käytä läpimenoaikoihin perustuvaa mittaristoa vaikuttavuuden varmistamiseksi ja luo laatua arvostava kulttuuri.

Avainsanat: liiketoimintamalli, skaalautuva ketteruus, empiirinen, arvon luonti, lean startup, laatu, läpimenoaika, mittari, ketteryyden skaalaamisen viitekehys (SaFE), koodikatselmointi, laadunparannus, toimintatutkimus

**Author** Matti Saarikallio  
Information Systems Science  
Faculty of Information Technology  
University of Jyväskylä  
Finland  
ORCID 0000-0003-2813-9146

**Supervisor** Pasi Tyrväinen  
Faculty of Information Technology  
University of Jyväskylä  
Finland

**Reviewers** Krzysztof Wnuk  
Department of Software Engineering  
Blekinge Institute of Technology  
Sweden

Casper Lassenius  
Department of Computer Science  
Aalto University  
Finland

**Opponent** Veikko Seppänen  
Oulu Business School  
University of Oulu  
Finland

## ACKNOWLEDGEMENTS

It all started when I happened to mention to Pasi that I was going to go on a study leave to finalize my studies in cognitive science. We were having lunch at the old rifle factory after a software business training he had given at my company. He commented that you already have one master's, why not do a Ph.D. instead. I asked if I could do it while working, and he said yes, but that way it will take about 10 years. After three jobs, two stints at self-employment, two children, and almost exactly 10 years I am a bit surprised that the schedule now seems to be doable.

This work would not have been possible without the support from my advisor, Professor Pasi Tyrväinen. I am grateful for the insightful advice and inspiring discussions during this process.

I would like to thank my co-authors for their contributions. Some of the papers would not have been published without your help. It was a privilege to work with professionals. I also thank the pre-examiners of this dissertation: Professor Krzysztof Wnuk of the Blekinge Institute of Technology and Professor Casper Lassenius of Aalto University. I am also very thankful for all the anonymous reviewers who gave their feedback on the individual articles of this dissertation.

My appreciation goes to the investigated companies for providing access to the necessary research data, and thanks to all my friends and colleagues who gave the necessary support and peer pressure to push forward.

I would like to thank Jenny and Antti Wihuri Foundation for the funding that allowed me to work for six months full time to get the research started and first papers published. I would also like to thank University of Jyväskylä for funding of the very important three months study leave to finalize my last paper and write the dissertation introduction. Without taking this time off from my day-job, this work could not have been completed.

Finally, my gratitude goes to my family who supported me during the whole process. Thank you to my mother Elisa and my father Pekka as well as my parents' in-law, Erja and Risto, especially for their help with childcare. My warmest thanks to my sons Aarno and Kaarlo and to my wife Suvi, who encouraged me to keep going.

Vihtavuori, May 2022  
Matti Saarikallio

## FIGURES

FIGURE 1	Dissertation at a glance .....	14
FIGURE 2	Two research phases: analysis tools and improvement tactics ....	24
FIGURE 3	Source, reason, and method constitute each revenue stream.....	30
FIGURE 4	The way to conduct analysis of an existing and design of a new business model differs. ....	32
FIGURE 5	Metrics framework for targeting improvement goals .....	35
FIGURE 6	Three interventions resulted in measurable improvements.....	37
FIGURE 7	Contextual antecedents, investigation tools, metrics and tactics, outcome measures.....	39

## TABLES

TABLE 1	Examples of decomposition of revenue streams .....	31
---------	--	----



# CONTENTS

ABSTRACT

TIIVISTELMÄ (ABSTRACT IN FINNISH)

ACKNOWLEDGEMENTS

FIGURES AND TABLES

CONTENTS

LIST OF INCLUDED ARTICLES

1	INTRODUCTION .....	13
2	THEORETICAL BACKGROUND .....	15
	2.1 Business model.....	16
	2.1.1 Value .....	17
	2.1.2 Revenue stream .....	18
	2.2 Improvement methods.....	18
	2.2.1 Lean startup .....	19
	2.2.2 Agile methods.....	19
	2.2.3 Quality practices and measurements .....	21
	2.3 Dyads of software-producing organizations .....	22
	2.4 Research objectives .....	23
	2.5 Relationship between the included articles .....	24
3	METHODOLOGY AND SCOPE.....	25
	3.1 Case studies .....	25
	3.2 Design science research .....	27
	3.3 Action research.....	27
4	OVERVIEW OF THE INCLUDED ARTICLES .....	29
	4.1 Article I: Revenue stream analysis .....	29
	4.2 Article II: Business model comparison .....	32
	4.3 Article III: Cycle-time reduction with metrics.....	34
	4.4 Article IV: Performance improvement with quality culture .....	36
	4.5 Synthesis of the articles .....	38
5	DISCUSSION AND CONCLUSION .....	40
	5.1 Theoretical contributions.....	40
	5.1.1 Analyzing .....	40
	5.1.2 Improving.....	41
	5.2 Managerial contributions .....	42
	5.3 Limitations .....	43
	5.4 Conclusion .....	45
	YHTEENVETO (SUMMARY IN FINNISH) .....	46

REFERENCES..... 53

ORIGINAL PAPERS

## LIST OF INCLUDED ARTICLES

- I Saarikallio, M., & Tyrväinen, P. (2014). Following the money: Revenue stream constituents in case of within-firm variation. In *Software Business: Towards Continuous Value Delivery, Proceedings of the 5th International Conference ICSOB* (pp. 88-99). Springer, Cham.
- II Vanhala, E., & Saarikallio, M. (2015). Business model elements in different types of organization in software business. *International Journal of Computer Information Systems and Industrial Management Applications*, 7, 139-150.
- III Tyrväinen, P., Saarikallio, M., Aho, T., Lehtonen, T., & Paukkeri, R. (2015). Metrics framework for cycle-time reduction in software value creation - adapting lean startup for established SaaS feature developers. In *ICSEA 2015: The Tenth International Conference on Software Engineering Advances*. IARIA.
- IV Saarikallio, M., & Tyrväinen P. (2022). Quality culture boosts agile transformation – action research in context of B2B software business. Submitted to *Journal of Software: Evolution and Process*.

### Contribution of the author to the included articles

- I Made the research plan, gathered theory and data for the research, analyzed the data, presented the findings at a conference and wrote most of the article.
- II Participated in developing the research plan, gathered half of the data, analyzed it and wrote half of the article.
- III Participated in planning of the research and writing parts of the article.
- IV Made the research plan, gathered theory and data for the research. Analyzed all the data and wrote the article.

# 1 INTRODUCTION

Businesses have been focusing on how to tackle volatility, uncertainty, complexity, and ambiguity (VUCA), a term that describes the context of the modern external environment (originating from a military context; see Whiteman, 1998). Solutions to address each of these areas have been suggested as being based on agility and flexibility, gathering new information and perspectives, restructuring operations to match external complexities, and conducting active experimentation (Bennett and Lemoine, 2014). The software industry is one cause for the change, but is also being impacted by it. Improving organizations that are continuously responding to external changes is the modern challenge that leaders have to deal with. The task is not easy, and abstract strategic-level ideas are not always enough because contextual variations call for situational adaptation.

A large body of research has investigated the adoption of the so-called agile methods in running the operations of software-producing organizations and increasingly other types of organizations as well (Dikert et al., 2016; Klünder et al., 2019; Uludag et al., 2018; Naslund and Kale, 2020). There are success stories but also cautionary tales regarding agile transitions (Denning, 2019). At the same time, new business models have appeared, especially in the software industry (Veit et al., 2014). Particularly in startups and new product development organizations, the need for speeding up time-to-market has caused a lot of interest in adopting more agile ways of organizing software delivery. However, many businesses are operating with a more complex business model compared with the simple small-team origins of agile, and many are still struggling to extract the full benefits from agile tactics. Therefore, this dissertation sets out to first investigate the business models of organizations and how established organizations compare with new organizations. Doing this allows for a deeper understanding of how such organizations can be analyzed. Thereafter, some organizational improvement possibilities are investigated in related contexts and are demonstrated in real-life situations.

Cusumano (2008) noticed that the natural life cycle of most companies goes from starting with a simple licensed product or bespoke projects to then slowly

increasing the amount of services before ultimately offering mostly services. This may lead to the situation where throughout their careers, many practitioners will be exposed to various ways of operating with shifting priorities. Here, a contextual understanding is important to adapt the management approach.

Before moving on to setting the theoretical background, for the reader to obtain a quick glance at the overall dissertation work, Figure 1 lists the main points from the dissertation's individual articles. In addition to the objectives, methods, and main findings, a small version of a relevant illustration is replicated here from each paper.



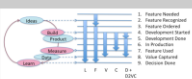

Paper	Objective	Method	Illustration	Main findings / conclusion
Following the money: revenue stream constituents in case of within-firm variation	Analyze and understand an established business model	Case study and conceptual model with 12 semi-structured interviews		Established hybrid business models can be understood starting from revenue stream analysis
Business model elements in different types of organization in software business	Compare established and new organizations	Multiple case study with 23 semi-structured interviews		Business model analysis reveals differences in new and established context and some similarities
Metrics framework for cycle-time reduction in software value creation	Improve organizations with lean/agile cycle-time metrics	Design science research utilizing process flow data		Cycle-time metrics framework can be adapted to focus an established SaaS organization's improvement efforts
Software quality in agile adoption – action research in context of B2B software business	Analyze and improve established agile organization with interventions	Action research with mixed methods data collection (with 41 initial interviews, and 3 year defect data)		Established business with complex business model and large size suggests cross-team communication structures and quality culture are keys to improvement

FIGURE 1 Dissertation at a glance

## 2 THEORETICAL BACKGROUND

The software business is a peculiar field. As “Uncle Bob” in his popular speeches has estimated, the number of software developers doubles every five years, meaning that half of the professionals in the field have less than five years of experience (Martin, 2019). A natural consequence of this is that the flow of new people leads to both positive and negative outcomes. On the one hand, there is an inflow of young people entering the field who are driving innovation; on the other hand, the lack of history leads to reinventing the wheel, hence packaging knowledge into fashionable models instead of solid and proven researched practices.

There are many definitions of the software business (Hyvönen, 2003; Cusumano, 2004; Hoch et al., 2000), but the organizations investigated in this dissertation fall into the category of software-producing organizations. It is defined here as a business that produces or maintains software source code and owns (or operates) a software production pipeline as a key part of its operations.

There is some discussion as to whether software engineering requires theory at all and if current theories can explain basic things such as software development method selection (Johnson et al., 2012). In this dissertation, the theoretical background is drawn mainly from two sources: the business model research and software practices literature (especially agile methods, lean startup, and quality improvement areas). Gregor (2006) suggests five types of theories in information systems research. Theories can be used for analyzing, explaining, predicting, explaining and predicting, along with design and action. This dissertation will use theory for both analysis and design and action. Thus, some theoretical concepts are introduced for the purpose of their relevance in relation to the empirical findings reported in the included articles.

## 2.1 Business model

“Tactics without strategy is the noise before defeat.” - Sun Tzu

In general, business strategy research is traditionally more interested in external forces and positioning a business in the industry to gain a competitive advantage. Business processes, however, dive deep into the details of operative management. A good middle ground between these is the concept of a business model. Although strategy remains an important aspect of management, business models are often more practical in highly complex and dynamic situations. It has been stated that the choice of business model is the most important strategic choice for new technology companies (Casadenus-Masanell and Ricart, 2010).

Business models have gained popularity among researchers over the past two decades. In their business model research review, Massa and his colleagues (2017) state four reasons why business models are useful in management theory: they have practical importance for competitiveness, often enabling above normal returns; they provide a new dimension of innovation above organizational innovation; they provide increased competition via lowering the barriers of entry, which forces established firms to reconfigure their businesses; and the compatibility of new types of value goals with old economic ones is an emerging research area that has been gaining ground.

The concept of the business model has approximately three different uses in the literature: the business model as an explanation of the business, the business model as a method to run the business, and the business model as a tool for developing the business (Spieth et al., 2014). Most interesting for the purposes of organizational improvement is this last usage, which encompasses both analyzing and designing business models. Wilson and Wnuk (2018) separate strategy and planning, daily operations, along with governance and communication as the contexts of business model. Johnson et al. (2011) explain the business model as more internal, only including customer value proposition, profit formula, key resources, and key processes as the main parts. Chesborough (2010) considers business model as a more extensive concept, defining it through the functions that it fulfills: articulating value proposition, identifying market segment and revenue generation mechanism, defining the value chain for distribution, describing the revenue mechanisms, estimating the cost and profit based on the value proposition and chain structure, positioning the firm in the value network, and formulating competitive advantage. Earlier divisions into, for example, revenue logic, a distribution model, a service and implementation model, and product strategy (Rajala et al., 2003) highlight the multitude of differing conceptualizations over time. Still, business model formulations typically consist of components (or elements). One popular and widely used division is customer segments, value propositions, channel, customer relationships, revenue stream, resources, activities, partnerships, and cost structure (Osterwalder, 2004; Osterwalder and Pigneur, 2010).

If the goal is to plan or design a new business model (such as in the case of startups), the revenue stream could be seen as resulting from the other parts of the business model. For example, Osterwalder and Pigneur (2010) see it as the result of value propositions offered to customers. However, if the goal is to evaluate or analyze a business model (such as in case of improving an established business or considering mergers and acquisitions), this dissertation will make the case that it could be more useful to actually start analysis with the revenue stream because other parts of the business model might be unclear or have changed over time.

Although the business model is increasingly used in practice, there has not been full agreement that the concept of the business model is a good approach toward management (Porter, 2001). Here, a pragmatic viewpoint is taken that although the objectivity of business model theory might be questioned, it has real consequences in the investigated organizations, where it is now often used as a guiding conceptual model, regardless of its subjectivity. Or as Thomas and Thomas (1928) put it, “if men define situations as real, they are real in their consequences.” This also relates to the fact that the business model is a powerful tool for creating an external cognition of the business (Kirsh, 2010). In practical industry use, the business model is a thinking tool used to create a shared understanding. Therefore, the components suggested by the business model canvas were chosen as the underlying conceptual frame for gathering empirical data in Articles I and II. Additionally, Article IV will analyze the case context using the conceptual model developed in Article I.

Thus, synthesized from the literature, the definition of a business model to ground the thinking in this dissertation is given as follows:

A business model is an averaged snapshot or aspired future model for a business unit’s way of organizing internally and in relation to the external world so that it can produce value for its customers and stakeholders and create revenue streams.

### **2.1.1 Value**

Most business model formulations use the term value quite liberally. Often, the business model elements are grouped into value-related categories: value proposition, value capture, value creation and delivery (Rachinger et al., 2019). The heavy interdependency of value creation and value delivery has been highlighted by Sjödin et al. (2020). This could indicate that dividing the business model based on the abstract concept of value without relation to more concrete concepts such as economic value may be problematic. Corporate governance thinking relates value to corporate performance in three forms: accounting value, stock performance, and value creation for other stakeholders (Goergen et al., 2010, p.65). The conceptual history of economic value includes a classical division into use value, exchange value, and price (Mill, 1885). Most theoretical divisions of value consider the (customer’s) use value as being relevant (Wnuk and Mudduluru, 2018, Jussila et al., 2017; Khurum et al., 2012; Rönkkö et al., 2009;



Woodal, 2003). Assuming a working market mechanism, this would be an upper bound for the price. An exception is the temporary technology-based monopolies (see Masters and Thiel, 2014, p. 48).

In this dissertation, a discussion of value relates to three important areas. Because value is typically used in the context of startups to think about customer use value, it is a good starting point for analysis in that context. However, established organizations become more focused on economic value and its capture as the market position is gained and competition intensifies. This highlights the practitioner's need to understand the revenue streams of the business. Additionally, the use value experienced by the customer relates to product and service quality, as will be discussed in Section 2.2.

### **2.1.2 Revenue stream**

Most formulations of a business model refer to some concept related to revenue. The business model literature discusses overlapping concepts such as revenues (Alt and Zimmermann, 2001), revenue model (Ojala and Tyrväinen, 2012; Stähler, 2002), revenue stream (Mahadevan, 2000, Luoma et al., 2012), revenue mix (Stähler, 2002), revenue logic (Rajala et al., 2003), earnings logic (Nenonen and Storbacka, 2010), income model (Rédis, 2009), and revenue mechanism (Chesbrough, 2007). This lack of cohesion is not only in the academic realm. When asked to describe how money is made in their company, 62% of executives had difficulties describing it (Shafer et al., 2005). Therefore, this dissertation investigates the way in which a business can be understood through its revenue streams.

It should be noted that the strategy literature uses the term pricing strategy. Sometimes, the revenue model is considered an equal term to pricing strategy (Sainio and Marjakoski, 2009). The concept chosen depends naturally on whether it is used in the context of planning a strategy or analyzing a business model. If the choice is made on the strategic level of thinking, it is not necessary to include it in the business model because it is a given. This could be typical of consumer markets. However, an established business-serving organization can have the possibility of going into customer-by-customer pricing, rendering the choice into a tactical subelement of the business model. Overall, the pricing capability is an important element of firm performance drivers in an established context (Falahat et al., 2020; Laatikainen and Ojala, 2021) and pricing choices impact operations (Saltan and Smolander, 2021).

## **2.2 Improvement methods**

*“Strategy without tactics is the slowest route to victory.” - Sun Tzu*

Business models and the operational practices of software-producing organizations are often investigated in isolation. However, there are some

practices that have originated at the team level and have become quite important in managerial thinking. Two such areas are the agile development methodology, which inspired the lean startup concept (Ries, 2011), along with the attempts that have been made to scale agile methods for larger organizations. Another one is the quality improvement methods, which extend from the practical production line tools to whole strategic initiatives such as total quality management, lean thinking, and Six Sigma (Kubiak and Benbow, 2016).

As a separation from business strategy and business models, the concept of tactics is perhaps the more accurate category for the concepts summarized in the following chapters. In the context of business models, tactics are defined as “the residual choices open to a firm by virtue of the business model it chooses to employ” (Casadenu-Masanell and Ricart, 2010). These additional choices might not be as relevant to give a competitive advantage, but if they are absent or poorly made, they can lead to numerous problems. Chess players know that even a bad implementation of a brilliant strategy wins but might require some dangerous sacrifices along the way.

### **2.2.1 Lean startup**

A business model is a good basis for planning a new venture. The startup community has also been eager to adopt another approach called a lean startup. In this approach, the lean startup cycle (Ries, 2011) is used to evaluate assumptions about customer needs by building something quickly, measuring customer acceptance, and taking the learning to iterate another loop until traction is proven (meaning that the offering has been shown it can produce value). The main assumption is that by shortening the cycle-time, it is cheaper to innovate on new business, and the learning speed is maximized.

### **2.2.2 Agile methods**

The lean startup rode the wave of agile methods movement during the last decade, and most companies now use one or another agile method. Although the iterative style of software development has most likely existed from the early days, the agile group of methods (Abrahamsson, et al., 2002) really picked up popularity after the practitioner-driven agile manifesto (Beck et al., 2001) with its four values and 12 principles. The extreme programming methodology (Beck, 2000) is often viewed as more technical and developer focused. Scrum and Kanban, on the other hand, include less technical and more organizational-workflow-related practices. Scrum, with 66% share, is the most popular one in annual industry reports, while 15% of companies are doing either Scrumban or Scrum/XP hybrid or pure Kanban (Digital, 2021).

One problem with the conceptual foundations of agile methods is that in practice, they are viewed from multiple angles, leading to different understandings (accidentally or purposefully). If the organization has continuous service development, agile is seen as part of service management. If the work is divided into projects, it is seen as a project management practice. This

can lead to quite a lot of confusion as various existing methods are incorporating agile thinking into older frameworks unrelated to software development. Analogical reasoning might not always hold if the principles are not fully understood. In fact, the classical notions of services, projects, and products might be unnecessary in close customer collaboration and advanced levels of continuous delivery, giving rise to new vocabulary suggestions, such as solution or release trains (Putta et al., 2018).

Recently, criticism of the lack of clear theory about agile methods has been raised, stating that the process level is not enough to define agility, but instead, additional factors need to be considered, such as agility of specific practices (Kuhrmann et al., 2021). This dissertation does not attempt to define agile methods conceptually. Rather, it is assumed that most organizations in the industry are already using them, at least partially. The focus here is on improving the software-producing organization in general, and agile methods can be one good source of inspiration for what could work in specific contexts.

Considering agility in light of business strategy, agility can be seen through the resource-based view (Barney, 1991) in the sense that a delivery organization with an agile culture used to be a key resource (or capability) of many early software producers in the form of organizational capital. However, this is becoming less of a unique competitive advantage because most software-producing organizations are gravitating to agile methods. Similarly, the ability to seize the sensed opportunities, which is behind the idea of dynamic capabilities (Teece, 1998), is in practice made easier through the agility of the organization because with an agile organization, it is no longer as difficult to reconfigure internal resources to keep up with changing environments, making the transition to new strategy execution faster. It should be noted that even the dynamic capabilities view of strategy is sometimes considered to apply only to relatively stable environments (Eisenhardt and Martin, 2000). Still, consultants applying agility principles to manufacturing firms have provided a multitude of business-agility-related concepts, such as the on-demand-sense-and-respond organization, adaptive enterprise, real-time enterprise, and agile enterprise (Oosterhout, 2010). Thus, the appeal of various agile methods seems to extend from software development methods to planning new business models and even to strategic thinking.

Although agility is a worthy goal in our VUCA world, the size of the software-producing organization can influence how easy it is to transform to agility in practice. Multi-team organizations do not always obtain the expected benefits (Kalenda et al., 2018; Karhapää et al., 2021). Research on scaling agile methods typically considers anything above six teams to be a large-scale team (Dikert et al., 2016; Abrar et al., 2019). The most typical problems that emanate from scaling agile methods to multiple teams are testing (quality) and coordination related (Petersen and Wohlin, 2010; Bentzen et al., 2021).

### 2.2.3 Quality practices and measurements

Relating to dynamic capabilities, the quality performance of a firm can be influenced by the way in which production is coordinated or organized by management (Teece, 1998). One limitation in agile methods, especially Scrum, is that for managers, the quality practices are easy to consider as a developer skill, not an organizational issue. The notion of leaving the “how” of software development to the team (Schwaber, 2004, p. 105) is a solid idea driving team self-management. However, it may be the root cause of this limiting consideration.

Quality has many conceptual definitions, and each organization (and customer) has preferences. The Committee for the Coordination of Statistical Activities (CCSA) has listed the following dimensions of quality: relevance, accuracy, accessibility, timeliness, punctuality, clarity, comparability, integrity, credibility, coherence, and methodological soundness. Al-Qutaish’s quality standard comparison (2010) suggests that the ISO 9126 is the most useful for software engineering. It standardizes some terminology and differentiates two categories: external and internal quality, as well as quality in use. For software quality, there also is now the ISO 25010, which lists reliability, functional suitability, performance, compatibility, operability, security, maintainability, and transferability. The American Society for Quality (ASQ) gives two technical meanings to quality: fitness for use and conformance to requirements. Early work on software quality models by McCall (1977) divides quality into three parts that are important to the quality model: factors or user point of view, criteria or developer point of view, and metrics to measure quality.

When the goal is to improve the operations of a software-producing organization, the conceptual understanding of quality is good, but measures of quality are more useful. The classic and widely adopted standard family ISO 9000 defines quality as the degree to which a set of inherent characteristics of an object fulfills requirements. For software, this means that when a customer requires something, the organization promises to deliver it, and it works as expected, then the software has a high degree of quality. Similarly, this dissertation will use a simple and practical quality measure: the trend of customer-reported defects that require code changes.

Boehm et al. (1976) have been an inspiration for many conceptual categorizations of quality, but what might be less widely known is that they also suggest quite an extensive list of practices that an organization could use to ensure quality: setting explicit objectives for quality, benchmarking, using quality checklists, establishing quality assurance activity, using machine-analyzable specifications, having testable requirements, establishing the requirements-properties matrix, standards for structured code, automatic code standards checking, and performing design and code inspections. Article IV in this dissertation will highlight the contemporary usefulness of two of these: quality checklists and code inspections.

The traditional quality management perspective to improving software engineering processes is the implementation of software process improvement

(SPI) programs such as CMMI or IEC/ISO 15504 (SPICE). These programs have not always resulted in the desired improvements, which has been attributed to failed implementations because of change resistance, lack of evidence, imposed SPI, lack of resources, and commercial pressures (Baddoo and Hall, 2001). Still, there is evidence that an increase in process maturity has a beneficial impact in the form of reducing the cycle time and effort via the mediating factor of improved quality (Harter, et al., 2000). Organizations implementing a general quality management system such as ISO 9001 have a goal of improving predictability of projects (Auer, et al., 1996). One reason for not adopting SPI could be the complexity of these process models compared with the simplicity of agile models.

As noted, implementing a complex quality system is not an easy change. The reason for this is that larger organizations in particular tend to have established cultures that define the ways of working. Understanding the culture before attempting quality improvement improves the likelihood of success (Maull et al., 2001; Gambi et al., 2015). A proponent of total quality management, Crosby (1989) explains that creating a quality culture boils down to doing the things right the first time and making people proud to work for the organization.

### **2.3 Dyads of software-producing organizations**

This dissertation will investigate and improve software-producing organizations in light of the following dyads: small organization versus large organization, new (startup or otherwise) organization versus established organization, and simple versus complex revenue structure.

When a new business grows, it can be seen as going through different stages in the following order: existence, survival, success, take-off, and resource maturity (Churchill and Lewis, 1983). In this dissertation, the term established refers to an organization that has reached the success stage. At this stage, the original founders might have left the company, and developing structures in preparation for growth or ascertainment of current profits with operational improvements becomes more relevant. Size often correlates with maturity, but in niche markets, this is not always the case. This is, in some cases, therefore, an additional dimension.

The complexity of the revenue structure is a phenomenon that typically happens with maturity in highly competitive markets, where there are less means to differentiate in other ways. A good example is the telecom market, where intense competition has given rise to quite complex and confusing business-to-consumer (B2C) pricing. Sometimes, the complexity is accidental and is related to historical reasons in small business-to-business (B2B) firms. The organization might have started with one revenue model and switched to another model for a different customer. The benefits from multiple revenue streams are not always achieved (Chikoto and Neely (2014) call this the mythology of revenue

diversification) because the impact that the increased complexity has on operations can have cost-increasing consequences.

## 2.4 Research objectives

The context determines what should be optimized. As the old saying goes, no business plan survives direct contact with the customer. Similarly, all business improvement tactics are likely to be highly context specific. The theoretical background summarized earlier suggests that a lot of research exists with a multitude of approaches towards conceptual foundations for analysis of organizations or prescribed improvement methods. What is perhaps less researched is the interplay of contextual dimensions, business model analysis, diagnosis of real-life organizational issues, selection of improvement tactics, cycle-time in the established organization context, and impact of tactics on organizational performance. Understanding how these pieces fit together is a very relevant puzzle for organizations.

Hence, this dissertation will work on the assumption that the context of the business determines what should be optimized when attempting to improve organizations. Consequently, the overall objective of the present research is to first increase the understanding of the established software-producing organization's business context, thereafter empirically exploring and demonstrating improvements in such organizations.

The main question that this dissertation investigates is as follows:

- How can an established software business be analyzed through a business model lens, and how can software organization improvement methods be applied to such organizations?

To answer this question, six sub-questions are investigated, which were considered the most important to answer in the light of literature. The first three relate to analyzing and understanding the context of a software business:

1. What are the relevant constituents of the revenue stream concept within a B2B software services company?
2. How can revenue streams as part of the business model be analyzed within a firm?
3. How is the organization or business type reflected in the emphasis of the business model elements in software firms?

Further, three questions relate to the tactics and tools for improving the effectiveness of software producing organizations:

4. What metrics would guide cycle-time-driven software engineering process development in established organizations?

5. What are the limitations of agile transformation in a multi-team organization with hybrid business model, and how can they be addressed?
6. What is the impact of promoting quality aspects?

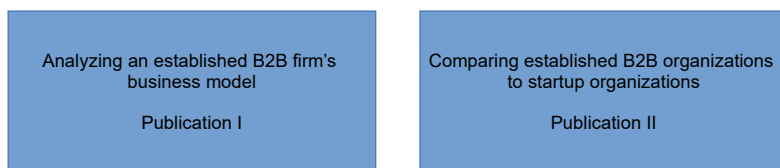
## 2.5 Relationship between the included articles

The research reported in this dissertation was conducted in two phases (Figure 2). The first phase investigated the tools for analyzing and understanding the context of software business and software-producing organizations. This included two separate studies. The first study analyzed an established B2B firm’s business model, focusing on bringing about a deeper understanding of the revenue stream aspect of the business model in the case organization. Article II compared established and startup organizations to better understand the use of business model, along with how practitioners in real organizations understand them.

The second phase of the research moved deeper to evaluate and use tactics and tools for improving the effectiveness of software-producing organizations. Article III focused on extending the lean startup ideas to improve an established organization by developing an applicable cycle-time based metrics framework to guide improvement actions. This was done in a small one-team context. Thereafter, Article IV looked at a larger seven-team context. The revenue stream model was used to gain a quick overview, but the main focus was on improving the organization by fixing the problems in scaled agile implementation, while also improving the quality culture of the organization.

---

Phase 1: Tools for analyzing and understanding the context of a software business




---

Phase 2: Tactics and tools for improving the effectiveness(vaikuttavuus) of software developing organizations

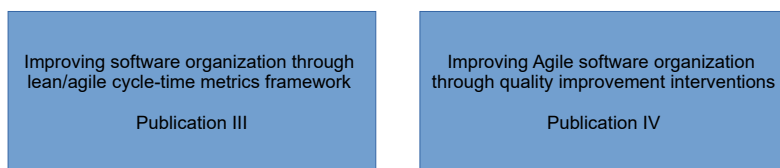


FIGURE 2 Two research phases: analysis tools and improvement tactics

### **3 METHODOLOGY AND SCOPE**

To answer the research questions about business organizations, a qualitative research approach is more appropriate because the goal is to explore the phenomena in real-life situations. The overall strategy for the research differs in the first and second phases. Answering the questions relating to analyzing organizations, a case study strategy was chosen for the first part. Because the goal of the second part was to explore improvement possibilities in the investigated organizations, design science research and action research were chosen.

Empirical research in software-producing organizations has limited validity because of the fast pace at which industry changes occur, but it is still the only way to bring the research closer to the industrial real-life context (Fernandez and Passoth, 2019). Additionally, there are multiple viewpoints to management research, and according to contingency theory, a contextual understanding is needed for decision making regarding the relevant approach to management (Dahlgaard-Park et al., 2018).

#### **3.1 Case studies**

A case study was used in two studies in the first research phase to explore the business model in the context of established business in detail and compare it to startup organizations in general. The first study was a single case study, and the second was a multiple case study. Case study research is recommended for the early stages of research as a way to provide new perspectives into already researched topics (Eisenhardt, 1989). Recently, Eisenhardt (2021) explains the goal of her method is about “building a set of constructs linked together in relationships that are supported by theoretical arguments that seek to explain a focal phenomenon.” Yin (2003) suggests that a case study design is suitable when trying to answer “how” or “why” questions and when attempting to examine the contextual conditions that are relevant to a phenomenon.



Article I elaborates on how revenue streams as part of the business model can be analyzed in the context of a specific B2B software firm. When seeking to clarify the concept of the revenue stream and its related sub-components, it was necessary to analyze the underlying patterns and gain an in-depth understanding. A qualitative research approach was chosen to improve the understanding of the investigated phenomenon (Yin, 1994).

Extreme examples are appropriate when seeking to extend theoretical understanding (Eisenhardt, 1989). Instead of the norm of one revenue model per business model, multiple co-existing revenue models and high within-firm variation in the revenue streams were chosen for the study reported in Article I, here with the goal of finding relevant new information about this specific context. Article II further compared business models in multiple cases, as motivated by the suggestion from Eisenhardt (1989) that the multiple case study method can be used for studying patterns in the cases and aiming for new theoretical perspectives.

### **Data collection and analysis**

Data were collected to understand the business model of an established business and another data set from multiple newer organizations. They both used semi-structured questions based on the theoretical frame of the business model canvas.

Article I was based on data collected through semi-structured interviews with 12 people representing corporate management, business unit management, and account management. Additionally, written materials were examined to verify contractual relationships. Qualitative content analysis was used with the steps of summarizing key themes, further explication of the data based on the themes, and using the emerging dimensions to present the results (Kohlbacher, 2006). The transcribed interview data were summarized into key themes in order to capture the main ideas from the interviews. Themes were grouped based on the theoretical model and described in its light.

The second study came about from wanting to compare different types of organizations, and this led to using a comparison of businesses in the axes of established business to startups, medium-sized to micro-sized, and organizations with different business types and fields. This enabled the finding of differences. Because in larger organizations the competitive strategy or organization structure would have had a greater impact, studying the business model concept in medium or smaller organizations was considered easier to arrange in practice. Thus, Article II included previously unpublished interview data points originally gathered for Article I, along with another previously unpublished interview data from the author. Additionally, the co-author's previous research (Vanhala and Kasurinen, 2014) with semi-structured questions using the same theoretical frame (business model canvas) was added to create a new combined dataset and structure the data based on comparing the results across the different organization and business types. Additionally, one extra interview was conducted. This created a dataset totaling 23 semi-structured interviews with people in business unit, account management, technical management, and CEO or owner-manager positions.

## 3.2 Design science research

The first phase of the research had related to investigating software-producing organizations through the lens of business model and developing revenue stream concepts. The second phase went on to explore possibilities to improve such organizations. For such a goal, design science was chosen because the goal was to construct a metrics framework based on lean startup ideas and evaluate the framework's applicability in the novel context of an established software-producing organization.

Thus, the approach suggested by the design science method (Hevner et al., 2004) was taken in Article III. First, an artifact was constructed in the form of a metrics framework by analyzing and synthesizing the previous research literature selected in light of the research question. Second, the metrics framework's ability to generalize was investigated by applying it to several contexts, here by choosing from a variety of metrics to target the different process development needs. Further, the metrics framework was applied to a case, and the impact was discussed on how it influenced the goals for additional process improvement actions in the organization.

### Data collection and analysis

Case data were collected to evaluate the example metrics. The data were collected from the software development process flow in the form of new features being deployed and flowing through the steps of the process. The time stamps of each event were recorded.

Analysis of the data was conducted by calculating three metric values for the case features: development cycle in workdays, lag to production from done to deployed in calendar days, and most interestingly the time from development done to value captured (meaning the feedback data from the actual use of the feature had been received).

## 3.3 Action research

Article IV was the longest research undertaken during this dissertation project because of the nature of action research, which can require multiple iteration rounds. The research iterations extended for about two and a half years, and quantitative data were drawn from a three-year period. The main goal of this research was to first analyze the organization and then make changes based on theoretical and empirical suggestions. Therefore, action research was chosen.

Action research aims to change the organization instead of merely describing it. The initial change recommendations are founded upon theory and lead to actions that produce changes in the organization; the actions and their impacts are reported, and learnings are discussed (Coghlan and Brannick, 2001; Lewin, 1947; Järvinen, 2021).

### **Data collection**

Mixed-method data collection was used. Qualitative data included semi-structured interviews, observations, and documents. The initial 41 interviews covered most people in the seven-team organization and gave a deep understanding of the starting situation.

In order to understand the results of the interventions, a quantitative defect count measure was used to increase the reliability of the results. This was collected from the organization's issue management system. Access was gained to data on all ongoing software development work of the investigated business unit over a three-year period. Here, the whole business unit was the level of analysis.

### **Data analysis**

Inductive content analysis (Elo and Kyngäs, 2008) was used for qualitative data collected from the interviews. The interview data were summarized and translated into English and coded using open and in vivo coding (Strauss and Corbin, 1998) to create categorizations that allowed for focusing the action plans for the intervention.

The quantitative data that supported the observed, qualitatively noted improvements were analyzed based on the assumption that defect data follow a Poisson distribution, as is the expectation for integer data. The likelihood was calculated to determine whether the results had changed by chance in order to increase the reliability of the qualitative results.

## 4 OVERVIEW OF THE INCLUDED ARTICLES

### 4.1 Article I: Revenue stream analysis

Saarikallio, M., & Tyrväinen, P. (2014). Following the money: Revenue stream constituents in case of within-firm variation. In *Software Business: Towards Continuous Value Delivery, Proceedings of the 5th International Conference IC SOB* (pp. 88-99). Springer, Cham.

“Who wishes to fight must first count the cost.” — Sun Tzu

#### Research objectives

The aim of this exploratory paper was to investigate how an established B2B software-producing organization can best be understood and to create a conceptual model for doing this. The focus was on improving the understanding of the hybrid business models.

The article started with the realization that it is challenging to analyze business models, especially the revenue streams of a firm with multiple customers and offerings combined with high variability in revenue stream configurations. As one informant put it, “It’s hard to tell which revenue stream contributes what because it seems the money goes into one bucket.”

Business model research can be about definitions, components (or elements), taxonomies, representations, change methodologies, and evaluation models (Pateli and Giaglis, 2003). This article is in the field of component research, in which the aim is to decompose a business model into its fundamental constructs, here especially focusing on the revenue-related component of the business model. Massa et al. (2017) suggest that one of the uses of a business model is to also isolate the components of focus from an organization’s activities. Expanding on this idea, the revenue stream component is investigated in detail.

The chosen research questions were as follows: What are the relevant constituents of the revenue stream concept within a B2B software services company? How can a revenue stream as part of the business model be analyzed within a firm?

### Findings

It was noticed that based on the literature, there was an incoherence of revenue aspects relating to business models. The most suitable term chosen was revenue stream because it is seen as having the potential for clarity by grounding it in the concrete and measurable money flow into the company.

Based on the literature and empirical data, the revenue stream was decomposed into three sub-components describing the revenue stream constituents of source, reason, and method (Figure 3).

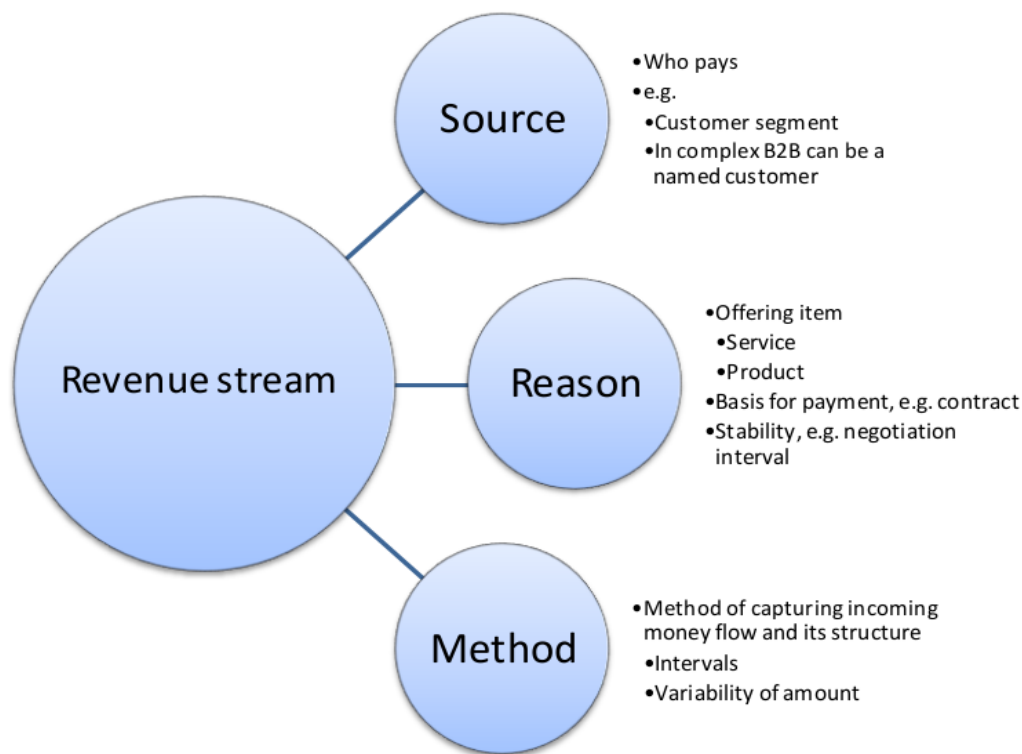


FIGURE 3 Source, reason, and method constitute each revenue stream

The source means who pays for the revenue stream, that is, a specific B2B customer or maybe a customer segment. The complexity of the investigated case partially emerged because of the sources of the revenue streams. It seemed that the dynamic nature of the customer was also reflected in the dynamic nature of the revenue stream. The case firm had five different kinds of sources of revenue, four individual companies, and one group of companies similar in their revenue streams. There were typically extensive customer-specific negotiations. Customer negotiation intervals seem to have an effect on the predictability of revenue.

The reason dimension was another explanatory factor for the variation in revenue streams. This was typically the offering item (product or service), and it typically had a contractual basis of some form. The case firm had nine separate reasons for revenue streams, which highlights the complexity. There was variation in packaging levels, offerings tended to change over time, and there was an active push towards maintenance over development.

For the method dimension, analysis was a bit more complex because there were differences across streams but similarities as well. Therefore, an analysis matrix was used, where the x-axis represented the reason dimension and the y-axis the source dimension. Going through the case data with this matrix analysis resulted in four revenue stream types that were similar in method.

Once the revenue stream methods were identified, it was possible to calculate the contribution of each type of revenue stream, and obtain an overview of the total revenue stream structure of the case firm. This helped the case company get clarity and support for their decision making going forward.

Thus, the conceptual model was validated as a useful tool to analyze an existing established business with a complex business model mix. The components are summarized in Table 1. The main finding is that an established software-producing organization can have a really complex implicit business model, and analyzing the business model needs to clarify the revenue streams of the organization in this kind of an established business. The CEO of the firm commented that the analysis provided her with a better understanding of the organization.

TABLE 1 Examples of decomposition of revenue streams

Component	Definition	Examples
Source	The originating source of revenue flow. From whom does the money come from.	Specific customer, customer segment, consumer segment.
Reason	The reason(s) why someone is paying the money.	Offering item, service or product, contractual relationship.
Method	The method of how the payment occurs and how it is structured.	Paid every month based on amount of worked hours with a minimum invoicing.

### Summary and relation to the whole

While the business model of the startup is built up starting from the market positioning and value proposition aspects, it was demonstrated that for established companies, a recommended approach is to start by analyzing the revenue streams of the organization. This can then help guide the design of operations through improved clarity. In essence, it is suggested that the business model analysis approach is turned upside down when compared with the traditional approach (see Figure 4). Analysis with this conceptual lens helped the case organization's management understand the economic reality of their business more clearly.

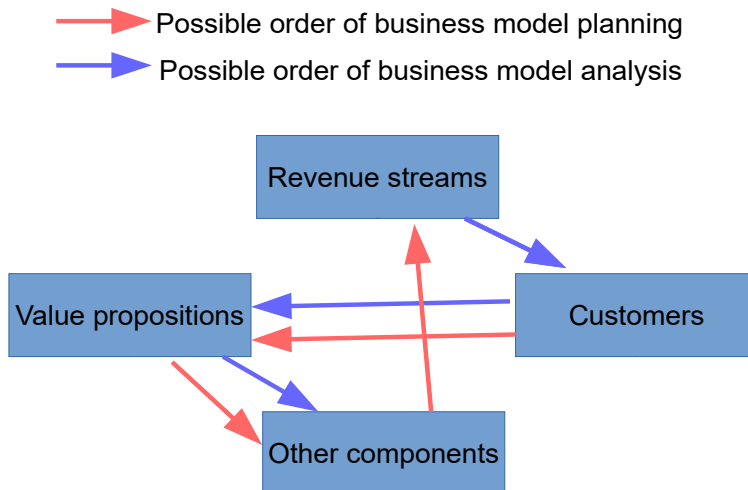


FIGURE 4 The way to conduct analysis of an existing and design of a new business model differs.

This article has started to answer the first part of the research question relating to analysis of an established software business via a business model lens, providing clarity that can give a good contextual basis for designing improvement for the operational side of the organization so that the economic reality of the firm is considered. Although this article focused on the revenue stream concept, the research project included questions related to the overall business model, giving rise to the idea for Article II, where the understanding of the business model lens is widened by comparing established and startup organizations.

## 4.2 Article II: Business model comparison

Vanhala, E., & Saarikallio, M. (2015). Business model elements in different types of organization in software business. *International Journal of Computer Information Systems and Industrial Management Applications*, 7, 139-150.

“You may be as different as the sun and the moon, but the same blood flows through both your hearts.” — George R.R. Martin, *A Game of Thrones*

### Research objectives

The aim of this article was to investigate the differences between different types of organizations and how different elements of business models are impacted. Similarities within the software business were also identified. Although business models have been analyzed quite extensively, earlier studies have not focused specifically on a comparison of how practitioners understand business models and similarities and differences between startups and established organizations. The data were collected from multiple case studies by two researchers who utilized the same business model framework.

## Findings

The business model canvas framework (Osterwalder and Pigneur, 2010) was found to be a good starting point for analyzing business models, but for deeper analysis, some adjustments might be needed to the components of the elements. The framework was designed to suit a range of business areas. For software businesses, the importance of human resources could be highlighted a bit more. This finding can be seen as analogical to the idea that in the software industry, the knowledge-based view of the firm becomes more important than the classical resource-based view (Håkansson, 2010; Grant, 1996). As Grant (1996) has highlighted, this increases the importance of effective coordination structures (in addition to collaboration structures) in such organizations, which is reflected in, for example, the agile methods' idea of cross-functional teams.

Going into the components of the business model, the key activities and key resources were the ones that were considered the most important, as expected by the importance of human capital in software business. The established larger organizations considered people being in specific roles as important, but this was not the case with startups that valued generalists instead. The working process differed as well: larger organizations had more systematic processes and metrics in place compared with the ad-hoc work found in startups.

The context of B2B had the effect of making customer segmentation more important as a way to obtain good customer references. The new B2C organizations saw the segmentation mainly as an extra cost in the form of additional marketing; references for B2C were also different, mainly coming in the form of getting game reviews in media and user feedback in app stores. Also, for startups in product business, the need to do marketing and build a brand from the start was seen as important.

Small companies tended to form partnerships with other small organizations and did not consider them as rivals. For the larger organizations, informants considered other parts of the organization as partners, which was a bit surprising and may indicate the need for more cross-organizational communication. The B2B organizations tended to conduct personal services to their customers, whereas the startups provided self-service and tried to create communities where users could help each other.

The value proposition in the B2B case focused on automating the customer's business processes. The B2C startups, on the other hand, focused on the user experience and trying to keep users.

The startups in this study were relying on external funding and did not yet have a revenue model built. Most were planning to use industry standard revenue models. The revenue model of the established organizations was more complex, and customer specific adaptations of the revenue model were in place.

## Summary and relation to the whole

By comparing their business model components, this article demonstrated the usefulness of the business model conceptual lens in analyzing organizations. The findings indicate that there are differences that can be unveiled and considered when analyzing software businesses through the business model lens. A deep



understanding of the business context is important when planning improvements to organizations; especially if the leader has previously worked in a different type of organization, this can help to avoid the hammer-nail a.k.a. one-size-fits-all -syndrome.

### 4.3 Article III: Cycle-time reduction with metrics

Tyrväinen, P., Saarikallio, M., Aho, T., Lehtonen, T., & Paukkeri, R. (2015). Metrics framework for cycle-time reduction in software value creation - adapting lean startup for established SaaS feature developers. In *ICSEA 2015: The Tenth International Conference on Software Engineering Advances*. IARIA.

“I failed. Good. Now go fail again.” – Ser Davos, George R.R. Martin, *A Game of Thrones*

#### Research objectives

Established organizations tend to have an established feature development process. Article II compared startups to established organization to understand the differences. Article III moves from analyzing and comparing to the theme of improvement possibilities. It investigated the possibilities of applying the practices promoted by the lean startup movement in the context of an established organization. With a renewed interest in so-called ambidextrous organizations (Benner and Tushman, 2003; Tushman and O’Reilly, 1996), it is becoming increasingly important to understand how to apply techniques originating from exploration-focused startup organizations to the context of established development processes.

#### Findings

The main undertaking in Article III was the construction of a metrics framework for addressing a gap between two types of metrics: the traditional software engineering process metrics and the customer value of offering (or value of startup) metrics. The main idea suggested by the previous research literature was that such a metrics framework could be built on the measurement of timestamps in various events related to the new feature development process.

The constructed metrics framework gives guidance to selecting what to measure, here depending on the improvement areas that the organization needs to focus its actions on. The measurement points of the process are listed on the right side of Figure 5 (feature needed, recognized, ordered, development started, done, in production, feature used, value captured, and decision done). The example cycles that can be derived from the framework are listed in the middle:

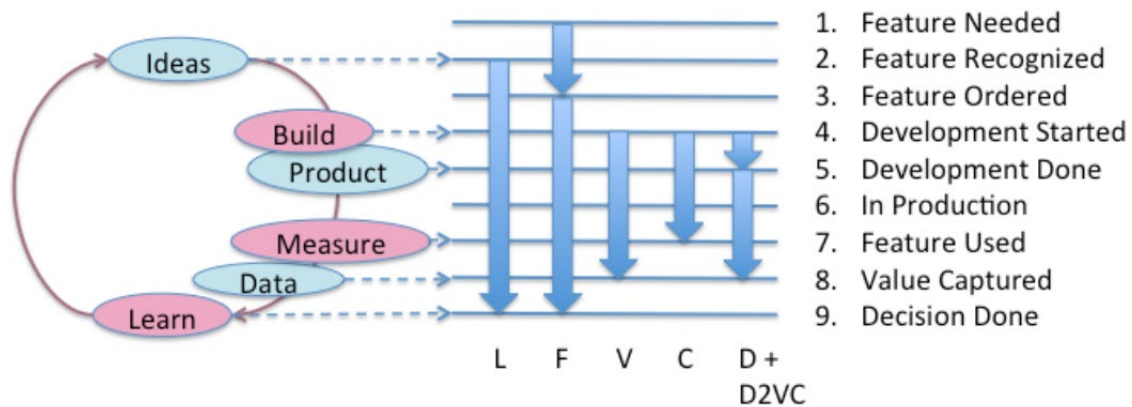


FIGURE 5 Metrics framework for targeting improvement goals

- L = Lean Startup cycle,
- F = Full cycle,
- V = Value cycle from starting the development to value capture,
- C = Core cycle from development start to first feature use,
- D = Development cycle from start of development to production readiness,
- D2VC = Time from development done to value captured).

On the left is the mapping to the popular lean startup cycle. The validation of the metrics framework in the case organization's process flow was conducted by demonstrating three different metrics: development cycle from start to done, lag to production from done to deployed, and done to value captured.

The application of the metrics framework's instantiations revealed interesting improvement possibilities in the organization. There were some features whose use had been discontinued quite soon, triggering a discussion about removing the feature from production or improving it further. It was noticed that in this organization, the lag from done to deployment was sometimes longer than the actual development cycle, which suggested focusing improvements on promoting customer adoption of new features. Extending the metrics all the way to customer feedback allowed for noticing some additional patterns that the organization was not aware of earlier. The features were heavily used on certain weekdays and less on others. Even more so, the vacation seasons had an impact on how long it took to get customer feedback for deployed new features. This led to redesigning the release windows in the case organization to reduce the negative effects related to feedback delay to the development team. Other identified examples of process development focus areas that are easily implementable via the metrics framework were speed of the software development process, evaluating continuous deployment capability, the interoperation of customer-facing and development teams, integrating analytics capabilities with the previous ones, profitability on the feature level (not possible to measure in all cases), post-development processes of value capturing, and capability of finding customer needs in the actionable market (fuzzy front end).

## Summary and relation to the whole

The agile methods used in most organizations aim to shorten the cycle time of software delivery. However, many organizations have been implementing agile method as a prescription of practices, not necessarily using the proper metrics to drive the optimal improvements that are likely to be context dependent. In relation to the previous investigation of business model differences and the complexity of the revenue streams in mixed business model B2B organizations, the metrics framework pointed out that although the lean startup inspired iterative improvement might not always be implementable as such in established organizations, there is a useful built-in idea of optimizing the organization with cycle-time based metrics. Once the business model context of the established organization is understood, the selection of the proper metrics is the next step, and the resulting measurement can be used to design the actual process improvements.

## 4.4 Article IV: Performance improvement with quality culture

Saarikallio, M., & Tyrväinen P. (2022). Quality culture boosts agile transformation – action research in context of B2B software business. Submitted to *Journal of Software: Evolution and Process*.

“Staying alive is not enough to guarantee survival. Development is the best way to ensure survival.” – Liu Cixin, *The Dark Forest*

### Research objectives

Article III took the approach in which lean startup metrics were adapted to the context of established software-producing organizations. The concept of lean startup shares its origins with the agile family of practices such as Scrum, Kanban, and eXtreme Programming. In Article IV, an established organization similar in its business model complexity to the case of Article I was first analyzed based on its revenue streams to understand its context. Then, improvement possibilities were identified and applied to the organization, and the resulting improvements in organizational capability were measured. Because the organization was already using Scrum, the size of the organization was larger (seven teams), and the established value-capture-focused nature of the organization did not make the lean startup metrics approach as relevant (although a basic cycle-time measure was in fact introduced), the objective was to look into other operational improvements and especially investigate the impact of promoting quality culture and practices in the organization as a potential area of organizational capability improvement.

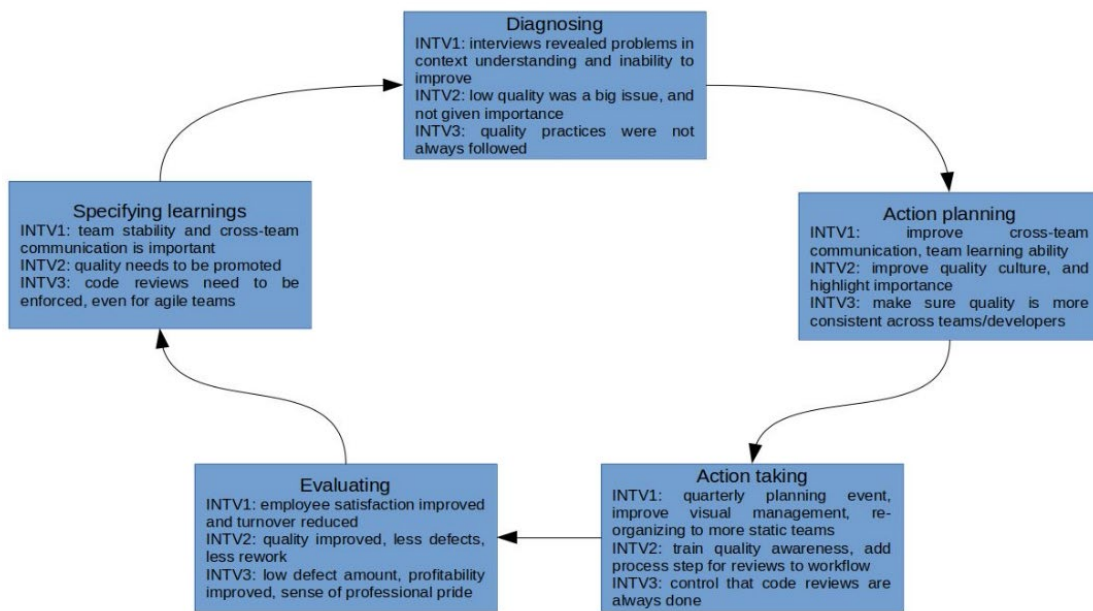


FIGURE 6 Three interventions resulted in measurable improvements

## Findings

The action research conducted in the fourth study revealed multiple problems in the investigated organization that had used Scrum for some years, and the diagnosed problems were remedied with the chosen interventions. An overview and highlights of the action research loop are given in Figure 6. Validation of the conceptual framework for analyzing the established organizations (developed in Article I) was strengthened during this research because it was used to describe the business model's revenue aspects to obtain insights into the complexity of the case business.

The initial diagnosis of the organization was based on semi-structured interviews revealing categories of topics considered important to either improve or maintain. The categories were cultural issues and the instability of teams, vision and context understanding, process and quality, product and quality, project management, along with knowledge and customer. These areas of interest emerging from the empirical data formed the starting point for the interventions.

The first intervention was motivated by the realization that scaling agile was failing in the organization, and it was theorized that learning to work as a team was not given the appropriate time to occur because people were often moved around. Additionally, a lack of context understanding was voiced. Intervention one targeted improving communication across teams, visual management unification, and maintaining more stable teams. The learning was that in this type of organization, it is important to keep the teams mostly stable and make sure some form of visualization of the workflow is provided. Furthermore, communication structures to share information over team or project boundaries must be in place. In this case, a quarterly increment planning event was utilized.

The second intervention was motivated by the interview data that pointed to complaints about a lack of testing, along with process and product quality issues. It was theorized that problems could relate to the fact that Scrum was the main agile method in use and that Scrum does not promote quality very much. Therefore, the promotion of a quality culture and suggestion to do code reviews were the main intervention targets. The learning from this intervention was that the promotion of quality can lead to reduced rework, which can be very good for the overall profitability of a mixed-business-model organization.

The third intervention was encouraged by the results of the second intervention. Measurement of how much code reviews were done revealed that although quality improvement in the form of fewer defects was observed, there was still quite a lot of development that did not go through the code review process. Theory related to code reviews suggests that an improvement in quality in the form of fewer defects is quite predictable; thus, further improvement would be possible. Therefore, the enforcement of code reviews was implemented. As the defect count dropped even more, the takeaway from this intervention was that even for agile teams, code reviews do not necessarily happen automatically, and controlling that quality practices are actually followed is a way to improve the quality of the organization's output.

#### **Summary and relation to the whole**

Article IV investigated an established software-producing organization that was similar in its revenue stream structure to the case in Article I. The main focus, however, was not only to analyze but also to investigate how this type of organization could improve its operational effectiveness. It turned out that the interplay of the complexity of the business model and larger size of the organization influenced the ability of this organization to gain the benefits of agile transformation using the Scrum method. To improve the situation, additional organizational activities relating to cross-team communication and providing a shared context were required. Additionally, a quality culture seemed to be very important, and the transformation to agile practices had not on its own effectuated high enough quality.

### **4.5 Synthesis of the articles**

Figure 7 summarizes and synthesizes the combined findings from the four articles. First, the contextual dimensions of software-producing organizations were identified in the initial step. Second, it was realized that business-model-based conceptual analysis can be used for a comparison of different types of organizations, and established organizations should start analysis from a revenue stream angle. A deeper understanding of the context requires an empirical diagnosis of the organization's situation, which here revealed the categories of issues listed in the diagram. Third, a framework of metrics to aim organizational process improvement was constructed and validated, and a

selection of improvement tactics was identified and implemented in the field. Fourth, a selection of output measures was found as useful to ascertain if improvements have the planned impact.

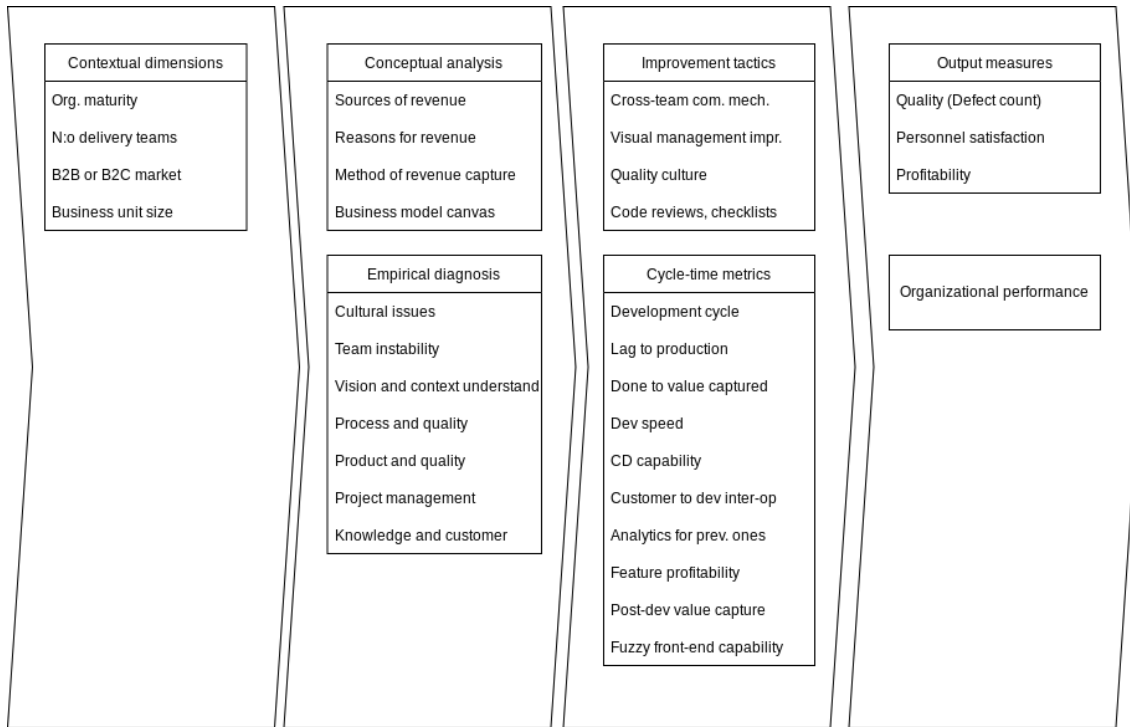


FIGURE 7 Contextual antecedents, investigation tools, metrics and tactics, outcome measures

## **5 DISCUSSION AND CONCLUSION**

Contributions to theory and practice, together with some future research directions, are discussed next. Thereafter, limitations are given, and the research is concluded.

### **5.1 Theoretical contributions**

In this dissertation, it was explored how an established software business can be analyzed through a business model lens and how improvement methods can be applied to such organizations.

#### **5.1.1 Analyzing**

Article I attempted to develop and test a finer-grained conceptual tool for examining revenue formation in a software context. A contribution has been made with an analytical and illustrative case study by showing the challenges of managing revenue streams with software service and products that are typically under constant development or updates.

Support has been provided for the concept of the business model as a good management tool to unveil the differences between startups and established organizations. This dissertation validates the usefulness of business model conceptually in multiple software-producing organizations. Attempting to use the business model in various contexts is sometimes quite complex. Therefore, it is suggested that the ordering in which analyses of business model components take place is relevant. If the goal is to obtain an understanding and snapshot view of the current de facto business model, it is easier to first analyze the revenue streams of the firm because without them, the firm would not exist (for long). On the other hand, if the goal is to design a new business, it becomes essential to start from customers and value creation because the urgency to prove the value creation ability is essential for the early stage of the firm's life-span.

Thus, extending the definition this dissertation started with to the following two-sentence definition is suggested: A business model is an averaged snapshot or aspired future model for a business unit's way of organizing internally and in relation to the external world so that it can produce value to its customers and stakeholders, along with revenue streams. The business model can be divided into components, and depending on the business type, different partitions are relevant, with the importance of the components varying.

### 5.1.2 Improving

Other researchers have formulated models that also highlight the relevance of continuous experimentation practices in the context of B2B-established companies if close trusting customer collaboration is possible (Mattos et al., 2021). This dissertation presented a metrics framework that can be a useful tool to support implementation of this kind of practices, and extend the cycle-time measurements toward continuous customer value optimization. This can be a useful tool to support the trending transition into fast cycle-time, highly integrated businesses (Järvinen et al., 2014).

The empirical findings regarding the importance of both openness for internal communication and quality, is a contradiction in the sense of business model in relation to organizational cultures (Hock et al., 2015). Quinn and Rorhbaugh (1983) have placed communication openness and quality on the opposite dimensions when comparing business model orientation on the axis of novelty-oriented cultural values and efficiency-oriented cultural values. This does not seem to hold in the context of scaling agility in the case of an established complex business model and a multi-team organization. In such case, both seem to be important. To speculate on the reasons for this, it may be that studies have traditionally grouped control mechanisms for finance, quality, and efficiency into the same bucket because they used to be managed by the same person. However, because agile methods have reduced the need for external control for efficiency, maybe the assumption has been that quality control can be dropped at the same time so that only finance control remains a management responsibility. This is a clear target for further research.

The research on agile methods has supported the idea that feature teams are preferable (Olsson et al., 2013; Larman and Vodde, 2008). This dissertation found instead that transforming to business domain competence-based teams worked well. The reason for this is most likely that there was a larger, more complex business domain in the investigated case, which would require an exceptional amount of learning for individuals in general-domain feature teams. Future research on the impact of business domain complexity on the optimal software team structure (e.g., based on product, project, component, feature, business domain, or customer) is called for.

Generally, the Scrum method is the most used agile method. There are still issues when it comes to making agile transformation a success. The usefulness of taking ideas from action research in boosting the agile transformation and providing the necessary double-loop learning (Argyris, 1991) for the



organization is suggested instead of just using it in a single-loop learning style prescription of a one-size-fits all procedure. Contextual adaptation is key.

The Scaled Agile Framework (SAFe) promotes the use of PI planning events (PI meaning program increments or sometimes product increments) to improve communications across teams (Bajpai et al., 2020; Paasivaara, 2017). This dissertation provides evidence that incorporating such structured information-sharing events can be used as a tool to provide a better contextual understanding for teams. The novelty here is that the investigated multi-team organization had a complex mix of products, services, and projects, and implementing a typical product-line-oriented SaFE model was not feasible because of existing contractual structures. The adapted event still provided benefits.

The largest improvement that was observed in the empirical investigation related to introducing a quality culture to boost agile transformations that had remained stagnant. The key tactical tools were general training to raise quality awareness, specifically the promotion of code review practices and personal check-lists. These are well-known techniques, but they are not always observed in practice in organizations rushing to adopt the latest agile method. It could be an industry-wide problem that the focus on managing speed causes the quality to become a lesser target for management, which ultimately ends up reducing speed. However, leaving quality as a mere personal developer issue can be a costly mistake for a software business.

## 5.2 Managerial contributions

This dissertation contributes to industry practice by introducing managerial conceptual tools and demonstrating their use in practice to gain better operational excellence and make tactical choices. The context of the established business is highly relevant in practice because the relevance of managing the organization increases compared with early startups. Therefore, for practicing managers, enriching the tactics toolkit with the following is suggested: analysis of revenue streams, comparison of software business models, selecting the focus of organizational development using lead time metrics, and boosting agile transformation by creating a quality-aware organizational culture.

Hence, the theme of improvement opportunities for organizational capability was explored in this dissertation. The recommendations that can be drawn from this research, or what seems to be essential, are six-fold. First, as noted by Fernandez and Passoth (2019), the fast pace of industry changes makes the validity of empirical research in software organizations limited. Thus, for the practitioner, it is not enough to rely on theoretical knowledge; rather, a deep understanding of the business model to find out where to focus is even more important than in other industries. Second, there are some easy universals, such as the importance of human aspects in the business model elements and general introduction of agile practices to improve the ability to adapt to changing VUCA environments. Third, understanding and clarifying revenue streams is the key to

finding value streams that can be optimized in established business. Business model conceptualization often promotes value proposition as the starting point, but one must understand that the goal of such conceptualization is to design a new business, not to understand or improve an established context. Fourth, agile methods are now the de facto tactic to improve the performance of value creation, but they do not necessarily go far enough. For the practicing manager that is planning an agile transformation, it is suggested that especially if bringing in external agile coaches, it is advisable to measure the impact on the desired outputs (such as defect rates). Fifth, to improve an established software-producing organization, a good tool is to optimize the value stream cycle times, which is achieved by choosing the right metrics based on the context. Sixth, ensuring a quality-focused culture is in place is key to further improve agile organizations. Overall, this dissertation has empirically demonstrated some concrete ways of improving the operational excellence of a software-producing organization.

There is a strong research interest in improving the dynamic capabilities of firms (Schilke et al., 2018) and providing advantages under conditions of change. It has been suggested that some corporations react to market changes by changing the organizational structure to improve agility (Lamberg et al., 2021). One contradictory result suggested by this dissertation was that at least on the team level, constant organizational structure changes can cause problems, preventing essential team bonding from happening and, thus, causing discontent among the people because they feel they are constantly “jumping around.” Good results were gained by stabilizing team structure toward domain knowledge based teams. The dangers of bringing a strategic-level concept of dynamic capabilities to scaled-agile-sized business units require more research.

### **5.3 Limitations**

Maxwell (1992) suggests that there are three main threats to the validity of qualitative research. Following this recommendation, the threats to the validity of this dissertation are discussed next in relation to descriptive validity, interpretive validity, and theoretical validity. Additionally, generalizability and possible bias of the research results are explored.

Descriptive validity relates to the possibility that the descriptions are inaccurate reflections of the real situation. Care was taken to recheck some of the descriptions from multiple data sources, such as contracts, and this supported the interview datasets and improved the validity of the descriptions. There were instances in which the informants misunderstood some of the concepts related to, for example, business model elements, but conclusions were not drawn from single data points. Having co-authors in the writing process of the papers also helped avoid misunderstandings.

Interpretive validity refers to the researcher’s sensitivity in capturing the meaning of the people involved in the studied phenomenon. The author had

prior experience in using semi-structured interviews and qualitative coding techniques in his master's thesis and had experience working in the investigated industry for over a decade before starting the current research. This made understanding the discussed phenomenon easier. The downside of practical experience is the possibility of introducing subjective bias to the interpretations. Having practical experience in multiple organizations and various roles mitigates this to a degree. Additionally, bias was reduced by using multiple data sources, which was especially important in the action research phase, where data collection rigor is the most important distinction from mere consulting (Baskerville and Wood-Harper, 1996).

Theoretical validity relates to the degree to which the theoretical explanations drawn from analysis are consistent with the data used. This especially relates to the interviews in which systematic coding methods were used. The validity of theoretical explanations in action research is also something to keep in mind. The fact that the chosen interventions were successful does not necessarily indicate that the theoretical explanation that led to those interventions was correct. It could be that some other explanation could have led to the selection of the same intervention. Hence, caution should be taken in extending the theoretical implications to other contexts.

Generally, drawing conclusions about causal relationships from qualitative research is not recommended. It was concluded with a high likelihood that the interventions lead to measurable results in defect reduction, but it cannot be claimed that the interventions were the only contributing factors. Mitigation to avoid this problem was carried out by providing a rich description of the context.

Qualitative research normally lacks the ability to generalize, but if such an attempt would be made by other researchers, it is advisable to note that the investigated context is limited geographically. Although there were international customers and non-native workers in the case companies, the local culture can have an impact on the relevancy of the findings outside Northern Europe.

An additional limitation in action research that needs to be considered is the participants' willingness to share or describe their experiences. Considering some of the emotional issues reported in the interview data, it is likely that this was not a large issue. It has been said that action research tends to overstate the importance of the intervention (Myers, 2019, p.77). To mitigate this, the background context description of Article IV is more in-depth than would have been necessary for just presenting the findings. This allows the reader to better evaluate the reasoning behind the interventions. The likelihood of causality of the resulting change relating to the interventions is strengthened with mixed-methods data and the quite substantial change in quantitative measure, but still, the author is aware that only replication of a similar context would warrant strong generalizability claims. The fact that a quantitative measure was employed will make it easier for other researchers to compare their results to those reported here.

## 5.4 Conclusion

In this dissertation, the idea was explored that the strategic differences of differently aged firms and parameters of a business model had an influence on the operational choices that a software delivering organization is most likely to benefit from. As the empirical explorations demonstrated, the context determines what should be optimized.

As the classic wisdom of software development states, there is no silver bullet (Brooks, 1995, p. 179). Based on this dissertation, the author has come to believe that there might be four “bullets” for serious consideration: ensuring team stability for learning, adequate communication structures for scaling, adopting cycle-time based metrics for effectiveness, and creating a culture that values quality to become a professional level organization.

## YHTEENVETO (SUMMARY IN FINNISH)

### **Monitahoisen ohjelmistoliiketoiminnan parantaminen: laatukulttuuri, läpimenoajat ja monitiimisen ketteryden johtaminen**

Ohjelmistoja tuottavat organisaatiot sykkivät uusien liiketoimintamallien ytimessä. Tällaiset organisaatiot voivat olla keskiössä tavoiteltaessa uudentyyppisiä kilpailuetuja kekseliäiden liiketoimintamallien avulla. Kuitenkin strategisen edun saavuttamisen jälkeenkin on usein mahdollista parantaa yrityksen operatiivista kyvykkyyttä. Tässä väitöskirjatyössä tutkittiin miten ohjelmistoja tuottavia organisaatioita (koodituotantotalot) kannattaa analysoida ja miten niiden operatiivista toimintaa voidaan parantaa tilannekohtainen toimintaympäristö huomioiden.

Yritykset etsivät keinoja toimia ns. VUCA toimintaympäristössä, jolle ominaista ovat nopeat muutokset, epävarmuus, monimutkaisuus ja moniselitteisyys (Whiteman, 1998). Tällaisen ympäristön haasteisiin ratkaisuja on etsitty toiminnan ketteryydestä ja joustavuudesta, tiedonkeruun ja uusien näkökulmien hankkimisesta, toimintojen uudelleenjärjestelystä vastaamaan ulkoista monimutkaisuutta ja aktiivisesta kokeilujen tekemisestä (Bennet ja Lemoine, 2014). Ohjelmistoteollisuus on yksi muutosten aiheuttaja, mutta on myös itse niiden vaikutuspiirissä. Jatkuvasti ulkoisiin muutoksiin reagoivien organisaatioiden kehittäminen paremmiksi on johtamisen haaste ajassamme. Tehtävä ei ole helppo, ja abstraktit strategiatason ideat eivät aina ole riittäviä johtuen tilannekohtaisesta vaihtelusta, johon pitää sopeutua.

Ohjelmistoliiketoiminta on siitä erityinen ala, että ohjelmoijien määrä on miltei tuplaantunut noin viiden vuoden välein. Kuten Bob Martin on suosituissa puheissaan huomauttanut, tämä tarkoittaa, että puolella alan ammattilaisista on alle viiden vuoden kokemus (Martin, 2019). Tällä on luonnollisesti hyviä ja huonoja seurauksia. Toisaalta uusien ihmisten virta alalle edistää innovaatioita, toisaalta historian puute johtaa helposti pyörien uudelleen keksimiseen ja tietämyksen paketoimiseen muodikkaisiin malleihin tieteellisesti tutkittujen käytäntöjen sijaan.

Tämä väitöskirja käsittelee ohjelmistoja tuottavia organisaatioita. Määritelmällisesti kyseessä on siis organisaatio, joka tuottaa ja ylläpitää ohjelmiston lähdekoodia ja omistaa ohjelmistotuotantolinjan olennaisena osana operatiivista toimintaansa.

Tärkeä teoreettinen taustakäsite tällaisten organisaatioiden tavoitteiden ymmärtämisessä on liiketoimintamalli. Se asemoituu käsitteenä strategian ja liiketoimintaprosessien välimaastoon. Tässä väitöskirjassa liiketoimintamallin ymmärretään olevan keskiarvoistettu poikkileikkaus tai tulevaisuuden tavoitetilä liiketoimintayksikön tavalle organisoitua sisäisesti ja suhteessa ulkoiseen maailmaan siten, että arvontuotanto asiakkaille ja sidosryhmille mahdollistuu, tulovirtoja synnyttäen.

Olennaisena teoreettisena käsitteenä on myös syytä tarkastella arvon käsitettä. Esimerkiksi uusyritysten ensisijaisena tavoitteena voi olla osoittaa

tarjooman kyky tuottaa käyttöarvoa asiakkaille. Vakiintuneemmissa organisaatioissa tavoite saattaa muuttua enemmän tuotetun arvon talteenoton optimoimiseen. Tällöin syntyy suurempi tarve ymmärtää tulovirtojen rakennetta. Käytännössä on havaittu, että 62% toimivan johdon edustajista ei kyennyt helposti kuvaamaan miten yritys teki rahaa (Shafer et al., 2005). Tämä osaltaan motivoi tarvetta liiketoimintamallin tulovirtoihin liittyvän käsitteistön selkiyttämiseen.

Liiketoiminnassa taktiikka voidaan määritellä tarkoittavan niitä jäljelle jääviä valintoja, jotka ovat yritykselle vielä avoimena liiketoimintamallin valinnan ja toimeenpanon jälkeen (Casadenus-Masanell ja Ricart, 2010). Tämä väitöskirja ammentaa kolmesta taktisiin valintoihin liittyvästä kehittämismenetelmästä. Ne ovat kokeilukulttuuripohjainen uusyrityttäjäisyys (lean startup), ketterät menetelmät (agile) ja laadun johtaminen (quality management). Kokeilukulttuuriin pohjautuvat uusyritykset pyrkivät arvioimaan hypoteesejaan asiakastarpeista siten että ne rakentavat nopeasti jotain valmista, mittaavat asiakkaiden hyväksyntää sille ja perustuen kokeilusta saatuihin oppeihin toistavat tätä silmukassa, kunnes löytävät todistettavasti asiakasarvoa tuottavan tarjooman, jolle alkaa syntyä kysyntää. Perusajatuksena on nopeuttaa oppimissykliä ja päästä näin nopeammin yli yrityksen selviytymiselle kriittisestä alkuvaiheesta. Ketterät menetelmät ovat tämänkin lähestymistavan taustavaikutin, ja niiden suosio siivitti kokeilukulttuurin painottamisen yleistymistä toimialalla.

Ketterät menetelmät ovat nykyisin de facto taktiikka useimmissa toimialan yrityksissä. Niille ei kuitenkaan ole kovin yksiselitteistä teoreettista pohjaa, mikä osaltaan myötävaikuttaa siihen, että ketteriin menetelmiin siirtymistä tavoittelevat muutoshankkeet eivät aina onnistu kovin hyvin. Näin on erityisesti, kun ketteryyttä yritetään skaalata monitiimisiin organisaatioihin. Ketteryys on alun perin tiimitason käsite, mutta sitä sovelletaan nykyisin myös laajemmin organisointumismallina ja jopa strategisen johtamisen piirissä.

Laadun johtaminen on teemana jäänyt toimialalla hieman vähemmälle kiinnostukselle ketterien menetelmien yleistyttyä. Yhtenä syynä voi olla se, että esimerkiksi Scrum-mallissa laadun ajatellaan olevan tiimin itsensä vastuulla. Joskus tämä tulkitaan virheellisesti niin että organisaation ei enää tarvitsisi erikseen varmistaa laadukkaan toiminnan toteutumista. Boehm et al. (1976) on vaikuttanut useiden laatustandardien syntyymiseen. He esittivät kattavan listauksen toimintatavoista, joilla voi parantaa laatua: eksplisiittisten laatutavoitteiden asettaminen, benchmarkkaus, tarkistuslistojen käyttö, laadunvarmistusaktiviteettien asettaminen, koneellisesti analysoitavien määrittelyjen käyttö, vaatimusten testattavuuden varmistaminen, vaatimus-ominaisuusmatriisi, strukturoidun koodin standardit, automaattinen koodistandardin tarkistaminen sekä suunnitelmien ja koodin katselmointi. Väitöskirjan neljäs julkaisu alleviivaa näistä erityisesti kahden tärkeyttä myös nykyaikana: tarkistuslistojen käyttö ja koodikatselmoinnit.

Laadun parantamisen kannalta on tärkeää ymmärtää organisaation vallitsevaa kulttuuria, jotta onnistumisen todennäköisyys paranee (Maull et al., 2001; Gambi et al., 2015). Laatukulttuurin luomisessa olennaista on kaksi asiaa: asioiden tekeminen oikein (=riittävän laadukkaasti) ensimmäisellä kerralla ja saada

ihmisille riittävä ammattiyhteisyys siitä, että työskentelevät organisaatiolle (Crosby, 1989).

Tässä väitöskirjassa pyritään vastaamaan seuraavaan kysymykseen: miten vakiintunutta ohjelmistoliiketoimintaa voidaan analysoida liiketoimintamallin valossa ja miten ohjelmisto-organisaation kehittämismenetelmiä voidaan soveltaa tämänkaltaisiin organisaatioihin. Vastauksia lähdettiin etsimään kahdessa tutkimusvaiheessa tarkentavilla kuudella kysymyksellä. Ensimmäisen vaiheen kysymykset liittyivät analysointiin ja ymmärtämiseen: 1) Mitkä ovat tulovirran käsitteen olennaiset rakennuspalaset yritystenväliseen markkinaan keskittyneessä ohjelmistopalveluyrityksessä? 2) Miten tulovirtaa osana liiketoimintamallia voidaan analysoida yrityksen sisäisesti? 3) Miten organisaation tai liiketoiminnan tyyppi heijastuu liiketoimintamallin elementtien painottamisessa ohjelmistoyrityksissä? Toisessa vaiheessa tutkimus pyrki kartoittamaan organisaation tuoksellisuutta parantavien taktiikoiden ja työkalujen käyttöä: 4) Mitkä mittaristot ohjaisivat läpimenoaikaperustaista ohjelmistotuotannon prosessikehitystä vakiintuneissa organisaatioissa? 5) Mitä rajoitteita on ketteriin menetelmiin siirryttäessä monitiimisisä organisaatioissa, jotka soveltavat monitahoista liiketoimintamallia ja miten nämä rajoitteet pitää huomioida? 6) Mikä vaikutus on laadun korostamisella?

Empiirinen tutkimus ohjelmistoja tuottavissa organisaatioissa on jonkin verran rajoittunut validiteettinsa suhteen, koska toimialan muutoksen ovat nopeita. Silti tällainen tutkimusote on ainoa tapa tuoda tutkimusta lähemmäs toimialan tosielämän toimintaympäristöä (Fernandez ja Passoth, 2019). Empiiriseen lähestymistapaan liittyen tässä väitöskirjassa käytettiin kolmea tutkimusmenetelmää: tapaustutkimusta, suunnittelutieteen menetelmää ja toimintatutkimusta.

Ensimmäisessä vaiheessa analysoitiin organisaatioita tapaustutkimuksen menetelmin. Tapaustutkimuksessa tavoitteena on rakentaa joukko toisiinsa suhtautuvia konstruktioita, jotka tukeutuvat teoreettiseen argumentaatioon, joka pyrkii selittämään tarkasteltavaa ilmiötä (Eisenhardt, 2021). Tapaustutkimus soveltuu lähestymistavaksi erityisesti silloin, kun etsitään vastauksia ”miten” ja ”miksi” kysymyksiin ja yritetään tarkastella tilannekohtaisia olosuhteita, jotka ovat ilmiölle olennaisia (Yin, 2003). Ääriesimerkit ovat asianmukaisia silloin kun pyritään laajentamaan ymmärrystä (Eisenhardt, 1989).

Suunnittelutieteen menetelmää (Hevner et al., 2004) käytettiin rakentamaan mittaristoviitekehityksen muodon saanut artefakti pohjautuen tutkimuskirjallisuuteen. Mittaristoviitekehityksen yleistettävyyttä tutkittiin soveltamalla sitä useisiin konteksteihin siten, että viitekehystä edustavia metriikoita käytettiin erilaisen prosessin kehittämistarpeiden tunnistamisessa. Metriikkaviitekehystä sovellettiin tapaukseen tarkastellen sen vaikutusta organisaation prosessiparannustoimenpiteiden päämäärien valintaan.

Toimintatutkimus (Coghlan ja Brannick, 2001; Lewin, 1947; Järvinen, 2021) oli käytössä väitöskirjan neljännessä tutkimuksessa. Siinä tavoitteena on muuttaa organisaatiota pelkän kuvaamisen sijaan. Muutossuositukset pohjautuvat teoriaan ja ne johtavat toimenpiteisiin, jotka muuttavat organisaatiota. Nämä toimenpiteet ja niiden vaikutukset raportoidaan ja oppimistuloksia arvioidaan.

Tiedonkeruussa hyödynnettiin puolistrukturoituja haastatteluita, joita oli ensimmäisessä tutkimuksessa 12 ja toisessa 23. Kolmannessa artikkelissa hyödynnettiin aikaleimoja ohjelmistokehitysprosessin virtauksen eri pisteistä. Näin saatiin tallennettua erilaisia metriikoita tapausyrityksestä. Neljännessä artikkelissa hyödynnettiin monimenetelmällistä tiedonkeruuta. Aluksi tehtiin 41 haastattelua ja lisäksi havainnointia ja kirjalliseen aineistoon perehtymistä. Haastattelujen muistiinpanoille tehtiin avoin ja 'in vivo' koodaus, jonka pohjalta voitiin tehdä induktiivinen kategorisointi haastateltavien kuvailemista haasteista, ongelmien diagnosointia varten. Lisäksi kerättiin raportoitujen bugien lukumääriä numeeriseksi tietoaineistoksi.

Ensimmäisen artikkelin löydöksenä todettiin, että vakiintuneiden ohjelmistoja tuottavien organisaatioiden implisiittiset liiketoimintamallit voivat olla todella monitahoisia (complex). Kun analysoidaan liiketoimintamallia, on tärkeää kirkastaa organisaation tulovirtojen rakenne kartoittamalla 1) tulovirtojen lähteet, 2) tulovirtojen olemassaolon syyt ja 3) tulovirtojen arvonkeruun menetelmät. Jos ollaan luomassa uutta liiketoimintaa, aloitetaan usein valitsemalla asiakkaat, miettimällä sen jälkeen arvolupaus, sitten muut liiketoimintamallin komponentit ja lopuksi tulovirrat. Ensimmäisessä artikkelissa tämän ehdotetaan kääntyvän pääläelleen silloin kun suunnittelun sijaan ollaan analysoimassa vakiintunutta liiketoimintaa. Sellaisessa tapauksessa olennaista on ensin ymmärtää tulovirtojen rakenne, seuraavaksi asiakkaat, kolmanneksi arvontuotanto ja lopuksi muut komponentit.

Toisessa artikkelissa tutkittiin erityyppisten ohjelmistoliiketoimintojen eroja. Löydöksenä oli, että liiketoimintamallin käsite soveltuu hyvin tuomaan esiin olennaisia eroavaisuuksia. Liiketoimintamallin luonne on erilainen erilaisissa yritys konteksteissa. Syvä ymmärrys liiketoiminnan kontekstista on tärkeää, kun suunnitellaan toiminnan parannuskohteita organisaatioissa.

Kolmannessa artikkelissa esiteltiin kolmen viitekehykseen perustuvan läpimenoaikamittarin konkreettista käyttöä; aika kehityksen aloittamisesta valmiiksi, viive tuotantoon eli valmiista asennetuksi ja kolmantena aika valmiista siihen, että tuotettu arvo saadaan kerättyä. Viitekehyksen mittari-instantaatiot paljastivat mielenkiintoisia parannusmahdollisuuksia tutkittavassa organisaatiossa. 1) Löydettiin joitain toiminnallisuuksia, joiden käyttö oli loppunut pian niiden valmistumisen jälkeen, mikä johti keskusteluun niiden parantamisesta tai poistamisesta tuotannosta. 2) Viive valmiista asennetuksi oli toisinaan pidempi kuin varsinainen ohjelmistokehityssykli minkä seurauksena toiminnan parantamista kohdennettiin kannustamaan nopeampaa uusien toiminnallisuuksien käyttöönottoa asiakkaalla. 3) Mittaamisen ulottaminen aina asiakaspalautteeseen asti mahdollisti aiemmin tiedostamattomien käyttäytymismallien havaitsemisen: joitain ominaisuuksia käytettiin enemmän tiettyinä viikonpäivinä ja vähemmän toisina, lomilla oli vaikutusta siihen, miten nopeasti asiakaspalautetta saatiin asennetuista uusista ominaisuuksista, seurauksena toimitusikkunat suunniteltiin uudelleen jotta saatiin vähennettyä kehitystiimille tulevan palautteen viiveen negatiivisia vaikutuksia. 4) Lisäksi ideoitiin muita esimerkkejä prosessikehittämisen fokusalueista: ohjelmistokehitysprosessin nopeus, jatkuvan tuotantoviennin



kyvyn arviointi, asiakasrajapinta- ja kehitystiimien yhteistoiminta, analytiikkakyvyn integroiminen edellisiin, toiminnallisuustason kannattavuus (ei aina mahdollista mitata), kehityksenjälkeiset prosessit arvon keräämiseksi, kyky löytää asiakastarpeita toiminnalliselta markkinalta (fuzzy front end eli tuotekehitysprosessin alkuvaihe).

Neljännessä artikkelissa tutkittiin monen tiimin ohjelmistotuotanto-organisaatiota, joka oli jo pidempään soveltanut Scrum-menetelmää saavuttamatta toivottuja tuloksia. Organisaatiota lähdettiin kehittämään toimintatutkimuksen avulla. Aluksi tilannetta diagnosoitiin haastatteluihin pohjaten. Haasteita kategorisoitiin seuraavasti: kulttuuriset asiat ja tiimien epästabiilius, vision ja kontekstin ymmärrys, prosessi ja laatu, tuote ja laatu, projektin hallinta, sekä tietämys ja asiakas. Näiden pohjalta lähdettiin rakentamaan sopivia interventioita toiminnan kehittämiseksi.

Ensimmäinen interventio lähti siitä, että haastattelut olivat paljastaneet ongelmia kokonaisuuden ja kontekstin ymmärtämisessä ja oli yleisesti tunne kyvyttömyydestä parantaa asioita. Suunnitelmaksi valittiin tiimien välisen kommunikaation parantaminen ja tiimien oppimiskyvyn kehittäminen. Toimenpiteenä mm. otettiin käyttöön kolmen kuukauden välein pidettävä suunnittelutapahtuma (skaalatun ketteryyden viitekehuksesta adaptoituna) ja muokattiin organisaatorakennetta siten että tiimien pysyvyys lisääntyi. Tämän jälkeen työntekijöiden tyytyväisyyden todettiin parantuneen ja myös henkilöstövaihtuvuus väheni. Oppina tästä oli organisaatiolle se, että tiimien pysyvyys ja tiimien ylitse tapahtuva kommunikaatio ja kontekstin luominen organisaation tavoitteista on tärkeää.

Toisen intervention motivaationa oli haastatteluissa nousseet kommentit laadun heikkoudesta ja ettei sitä aina pidetty kovin tärkeänä. Suunnitelmana tämän pohjalta oli parantaa organisaation laatukulttuuria ja korostaa yleisesti laadun merkitystä. Konkreettisine toimenpiteinä koulutettiin yleisiä laatuasioita ja kasvatettiin tietoisuutta niistä esimerkiksi kehityskeskusteluissa ja tiimipalaverissa, lisäksi ohjelmistotuotannon prosessiin lisättiin nimetty vaihe koodikatselmoinnille. Arvioiden mukaan laatu parani, bugien määrä väheni ja uudelleen-työstön määrä väheni. Tälle saatiin myös vahvistus numeerisen asiakkaiden raportoimien bugien määrän mittauksen kautta. Organisaatiolle oli tämän jälkeen selvää, että laadun korostaminen on tärkeää.

Kolmannen intervention tarve nousi havainnosta, että vaikka laatukäytäntöiden tärkeyttä korostettiin ei niitä aina noudatettu kaikissa tiimeissä esimerkiksi koodikatselmointien osalta. Toimenpidesuunnitelmana oli pyrkiä varmistamaan, että tuotettu laatu olisi johdonmukaisempaa riippumatta tiimistä tai kehittäjästä. Toteutettu toimenpide oli se, että vaadittiin koodikatselmointien tekeminen kaikelle lähdekoodille. Tämän toteutumista myös mitattiin ja organisaatiolle tiedotettiin säännöllisesti miten suuri prosentti-osuus uudesta lähdekoodista oli katselmoitu. Lopulta päädyttiin huomattavaan bugien määrän vähentymiseen ja kasvaneeseen ammattitilpeyden tunteeseen tiimeissä. Organisaation oppi tästä interventiosta oli, että koodikatselmointien toteutuminen on hyvä erikseen varmistaa myös ketterissä itseohjautuvissa tiimeissä.

Tämä väitöskirjatutkimus pyrki tuottamaan teoreettista lisätietoa ohjelmistoyritysten analysoimiseen ja parantamiseen. Ensiksi pyrittiin luomaan paranneltu käsitteellinen työkalu, tulovirtojen rakennemalli, jolla voi analysoida tulovirtojen muodostumista ohjelmistoja tuottavassa organisaatioissa. Väitöskirja antoi myös tukea liiketoimintamalli-käsitteen yleiselle hyödyllisyydelle työkaluna pyrittäessä hahmottamaan eroja uusien ja vakiintuneiden yritysten välillä. Liiketoimintamallin käsitteen käyttökelpoisuutta validoitiin useissa ohjelmistoja tuottavissa organisaatioissa. Lisäksi havaittiin, että joissain tapauksissa mallin käyttö sellaisenaan on hankalaa, ja sillä missä järjestyksessä sen komponentteja tarkastellaan voi olla merkitystä. Lisäksi eri komponenttien tärkeys riippuu kontekstista.

Toiminnan kehittämiseen liittyen esiteltiin mittarointiviitekehys, jonka avulla voidaan tukea jatkuviin kokeiluihin perustuvien käytänteiden kehittämistä vakiintuneissa yrityksissä. Lisäksi ketteryyden skaalaamiseen liittyen ja osin vastoin perinteistä organisaatiokulttuurijaottelua havaittiin, että sekä laatua että kommunikaatiota painottavat kulttuurilliset piirteet ja niiden johtaminen ovat tarpeen, kun puhutaan vakiintuneista monitiimistä organisaatioista, jotka toteuttavat monitahoista liiketoimintamallia. Laadun korostaminen on perinteisesti liitetty kontrollimekanismeihin, yhdessä taloudellisten ja tehokkuuteen liittyvien kontrollimekanismien kanssa, ja sama henkilö on tyypillisesti johtanut näitä kaikkia. Ketterien menetelmien myötä ulkoisen tehokkuuskontrollin tarve on vähentynyt ja kenties oletus on ollut, että laatukontrollista voidaan luopua ja laatu automaattisesti syntyy tiimeissä. Mahdollisesti virheellinen seuraus tästä on, että ainoastaan finanssikontrolli jää johtamiskohteeksi. Tämä on selkeä kohde jatkotutkimukselle.

Ketteriin menetelmiin liittyvä tutkimus suosittelee feature-tiimeiksi organisoitumista (Olsson et al., 2013; Larman ja Vodde, 2008). Tässä väitöstutkimuksessa havaittiin, että siirtyminen liiketoimintaosaamis pohjaisiin tiimeihin toimi hyvin. Syy saattaa liittyä siihen, että tutkitussa tapauksessa oli kyseessä hieman laajempi ja monimutkaisempi liiketoiminta-domain mikä vaatisi poikkeuksellisen laajaa toimialaosaamista, jotta feature tiimit olisivat mahdollisia. Jatkotutkimukset siitä miten liiketoiminta-domainin monimutkaisuus vaikuttaa optimaaliseen softatiimin rakenteeseen (esimerkiksi pohjautuen tuotteisiin, projekteihin, komponentteihin, featureihin, liiketoiminta-domainiin tai asiakkuuteen) olisi mielenkiintoista.

Yleisesti Scrum on käytetyin ketterä menetelmä. Ketteryyteen siirtymisessä on kuitenkin usein haasteita. Tässä väitöksessä ehdotettiin hyödylliseksi ottaa toimintatutkimuksesta ajatuksia parantamaan ketterän siirtymän onnistumista sen sijaan että ketteryys yritetään tuoda orjallisesti reseptiä noudattaen. Tilannekohtainen mukauttaminen on keskeistä.

Ketteryyden skaalaamisen viitekehys (SAFe) suosittelee yhteissuunnittelutapahtumia (program increment planning) joilla parannetaan tiimien välistä kommunikaatiota (Bajpai et al., 2020; Paasivaara, 2017). Tässä väitöskirjassa löydettiin tukea tällaisen strukturoidun tiedonjakotapahtuman hyödyllisyydelle paremman kontekstiymmärryksen synnyttämiseksi. Utta tässä oli se, että tämä

tuotelinjapohjaiseen toimintaan tarkoitettu menetelmä vaikutti toimivalta myös monitiimisessä organisaatiossa, jolla oli monitahoinen sekoitelma tuotteita, palveluita ja projekteja.

Väitöskirjan merkittävin organisaation toiminnan parannus havaittiin empiirisessä tutkimuksessa liittyen laatukulttuuriin panostamiseen tilanteessa, jossa ketterän siirtymän hyödyt eivät olleet täysin toteutuneet. Käytetyt taktiset työkalut tässä olivat koulutukset laatutietoisuuden parantamiseksi, erityisesti koodikatselmointikäytänteiden suosittelu ja henkilökohtaisten tarkistuslistojen käyttö. Nämä ovat tunnettuja tekniikoita, mutta niitä ei aina ole kuitenkaan käytössä organisaatioissa, jotka ryntäävät kohti viimeisintä ketterää menetelmää. Voi olla toimialan laajuinen ongelma, että kun keskitytään johtamaan nopeutta se aiheuttaa laadun muuttumisen vähäisemmäksi tavoitteeksi mikä lopulta johtaa nopeuden vähentymiseen. Laadun jättäminen pelkästään ohjelmistokehittäjien henkilökohtaiseksi asiaksi voi olla kallis virhe liiketoiminnalle.

Tämä väitöskirja tuottaa tietoa ohjelmistotuotannon käytännön johtamiseen esittelemällä käsitteellisiä työkaluja ja demonstroimalla niiden käyttöä operatiivisen tuloksellisuuden parantamisessa ja taktisten valintojen pohjana. Vakiintuneen liiketoiminnan konteksti on tärkeä siksi että organisaation johtamisen tärkeys korostuu verrattuna varhaisen vaiheen uusyrytykseen. Siksi ehdotetaan, että käytännön johtamistyön työkalupakkiin lisätään seuraavat työkalut: tulovirtojen rakenteen analysointi, ohjelmistoyritysten liiketoimintamallien vertailu, organisaation kehittämiskohteiden valinta perustuen läpimenoaikaperusteisiin mittareihin ja ketteryyteen siirtymisen tehostaminen luomalla laatutietoinen organisaatiokulttuuri.

Kuten perinteinen ohjelmistoalan sanonta kuuluu, ei ole olemassa hopealuotia (Brooks, 1995, s. 179). Perustuen tähän väitöstutkimukseen kirjoittaja on tullut siihen johtopäätökseen, että saattaa olla neljä "luotia" joita kannattaa kokeilla: tiimien pysyvyyden tukeminen oppimisen varmistamiseksi, riittävän kommunikaatiostruktuurin luominen skaalautumisen mahdollistamiseksi, läpimenoaikoihin perustuvan mittariston käyttö vaikuttavuuden varmistamiseksi ja laatua arvostavan kulttuurin luominen.

## REFERENCES

- Abrahamsson, P., Salo, O., Ronkainen, J., Warsta, J. (2002). Agile software development methods: review and analysis. VTT publication 478. Espoo.
- Al-Qutaish, R. E. (2010). Quality models in software engineering literature: an analytical and comparative study. *Journal of American Science*, 6(3), 166-175.
- Argyris, C. (1991). Teaching smart people how to learn. *Harvard business review*, 69(3).
- Auer, A., Karjalainen, J., Seppänen, V. (1996). Improving R&D processes by an ISO 9001-based quality management system. *Journal of systems architecture*. 42(8), 643-651.
- Bajpai, S., Joglekar, N., Eppinger, S. (2020). Enhancing Visibility in Agile Program Increment DSMs. In *DS 103: Proceedings of the 22nd International DSM Conference (DSM 2020)*, MIT, Cambridge, Massachusetts, October 13th-15th 2020 (pp. 1-10).
- Barney, J. (1991). Firm resources and sustained competitive advantage. *Journal of Management*, 17(1), 99.
- Baskerville, R. L., Wood-Harper, A. T. (1996). A Critical Perspective on Action Research as a Method for Information Systems Research," *Journal of Information Technology* (11), 1996, pp. 235-246.
- Beck, K. (2000). *Extreme programming explained: embrace the change*. Addison-Wesley. Boston. ISBN 0201616416.
- Benner, M. J., Tushman, M. L. (2003). Exploitation, exploration, and process management: The productivity dilemma revisited. *Academy of management review*, 28(2), 238-256.
- Bennett, N., Lemoine, G. J. (2014). What a difference a word makes: Understanding threats to performance in a VUCA world. *Business Horizons*, 57(3), 311-317.
- Boehm, B. W., Brown, J. R., Lipow, M. (1976, October). Quantitative evaluation of software quality. In *Proceedings of the 2nd international conference on Software engineering* (pp. 592-605).
- Brooks, F.P.Jr. (1995). *The mythical man-month: essays on software engineering*. Anniversary ed. ISBN 0-201-83595-9. Addison Wesley Longman. Reading.
- Casadesus-Masanell, R., Ricart, J. E. (2010). From strategy to business models and onto tactics. *Long range planning*, 43(2-3), 195-215.
- Chesbrough, H. (2007). Business model innovation: it's not just about technology anymore. *Strategy & leadership*, 35(6), 12-17 (2007).
- Chikoto, G. L., Neely, D. G. (2014). Building nonprofit financial capacity: The impact of revenue concentration and overhead costs. *Nonprofit and Voluntary Sector Quarterly*, 43(3), 570-588.
- Coghlan D, Brannick T. (2001). *Doing action research in your own organization*. 2001. London, Sage.

- Crosby, P. (1989). Crosby talks quality. *The TQM Magazine*, Vol. 1 No. 4. <https://doi.org/10.1108/eb059474>
- Cusumano, M. A. (2004). *The business of software: What every manager, programmer, and entrepreneur must know to thrive and survive in good times and bad*. Simon and Schuster.
- Dahlgard-Park, S. M., Reyes, L., Chen, C. K. (2018). The evolution and convergence of total quality management and management theories. *Total Quality Management & Business Excellence*, 29(9-10), 1108-1128.
- Denning, S. (2019). Lessons learned from mapping successful and unsuccessful Agile transformation journeys. *Strategy & Leadership*.
- Digital.ai (2021). 15th annual state of agile report. Available online: <https://info.digital.ai/rs/981-LQX-968/images/RE-SA-15th-Annual-State-Of-Agile-Report.pdf>. Accessed 2021-12-21.
- Eisenhardt, K. M. (2021). What is the Eisenhardt Method, really?. *Strategic Organization*, 19(1), 147-160.
- Eisenhardt, K. M., Martin, J. A. (2000). Dynamic capabilities: what are they?. *Strategic management journal*, 21(10 - 11), 1105-1121.
- Elo, S., Kyngäs, H. (2008). The qualitative content analysis process. In: *Journal of Advanced Nursing*, 62-1, 2008, 107-115.
- Falahat, M., Ramayah, T., Soto-Acosta, P., Lee, Y. Y. (2020). SMEs internationalization: The role of product innovation, market intelligence, pricing and marketing communication capabilities as drivers of SMEs' international performance. *Technological Forecasting and Social Change*, 152, 119908.
- Fernández, D. M., Passoth, J. H. (2019). Empirical software engineering: from discipline to interdiscipline. *Journal of Systems and Software*, 148, 170-179.
- Gambi, L. D. N., Boer, H., Gerolamo, M. C., Jørgensen, F., Carpinetti, L. C. R. (2015). The relationship between organizational culture and quality techniques, and its impact on operational performance. *International Journal of Operations & Production Management*.
- Goergen, M., Mallin, C., Eve, M-K., Al-Hawamdeh, A., Chiu, I, H-Y. (2010). *Corporate governance and complexity theory*. Edward Elgar Publishing, Cheltenham. 2010. ISBN: 978-1-84980-104-1.
- Grant, R. M. (1996). Toward a knowledge - based theory of the firm. *Strategic management journal*, 17(S2), 109-122.
- Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design science in information systems research. *MIS quarterly*, 75-105.
- Hoch, D. J., Roeding, C., Lindner, S. K., & Purkert, G. (2000). *Secrets of software success*. Boston: Harvard Business School Press.
- Hyvönen, E. (2003). *Ohjelmistoliiketoiminta*. ISBN 951-0-26996-4.
- Håkanson, L. (201). The firm as an epistemic community: the knowledge-based view revisited, *Industrial and Corporate Change*, Volume 19, Issue 6, December 2010, Pages 1801–1828, <https://doi.org/10.1093/icc/dtq052>

- Issa Mattos, D., Dakkak, A., Bosch, J., Olsson, HH. (2021). The HURRIER process for experimentation in business-to-business mission-critical systems. *J Softw Evol Proc.* 2021;e2390. doi:10.1002/smr.2390
- Johnson, Pontus & Ekstedt, Mathias & Jacobson, Ivar. (2012). Where's the Theory for Software Engineering?. *IEEE Software.* 10.1109/MS.2012.127.
- Johnson, Christensen, Kagermann (2011). In *On business model innovation.*
- Jussila, J., Sillanpää, V., Lehtonen, T., Helander, N. (2017). Value assessment of e-government service from municipality perspective.
- Järvinen, J., Huomo, T., Mikkonen, T., Tyrväinen, P. (2014). From Agile Software Development to Mercury Business. In C. Lassenius, & K. Smolander (Eds.), *Software Business. Towards Continuous Value Delivery : 5th International Conference, ICSOB 2014, Paphos, Cyprus, June 16-18, 2014. Proceedings* (pp. 58-71). Springer. *Lecture Notes in Business Information Processing*, 182. [https://doi.org/10.1007/978-3-319-08738-2\\_5](https://doi.org/10.1007/978-3-319-08738-2_5)
- Järvinen P. (2021). Improving guidelines and developing a taxonomy of methodologies for research in information systems. 2021. Jyväskylä. JYU dissertations.
- Karhapää, P., Behutiye, PR., Oivo, M., Costal, D., Franch, X., Aaramaa, S., Choras, M., Partanen, J., Abherve, A. (2021). Strategies to manage quality requirements in agile software development: a multiple case study. *Empirical Software Engineering*, 26(2), 1-59.
- Kirsh, D. Thinking with external representations. *AI & Soc* 25, 441–454 (2010). <https://doi-org/10.1007/s00146-010-0272-8>
- Klünder, J. A. C., Hohl, P., Prenner, N., Schneider, K. (2019). Transformation towards agile software product line engineering in large companies: A literature review. *Journal of Software: Evolution and Process*, 31(5), e2168.
- Kubiak, T. M., Benbow, D. W. (2016). *The certified six sigma black belt handbook.* 3rd edition. Quality Press.
- Kuhrmann M. et al., "What Makes Agile Software Development Agile," in *IEEE Transactions on Software Engineering*, doi: 10.1109/TSE.2021.3099532.
- Laatikainen, G., Ojala, A. (2021). The pricing capability lifecycle of digital innovations. *Technology Analysis and Strategic Management.*
- Lamberg, J-A., Lubinaitė, S., Ojala, J., Tikkanen, H. (2021). The curse of agility: The Nokia Corporation and the loss of market dominance in mobile phones, 2003–2013, *Business History*, 63:4, 574-605, DOI: 10.1080/00076791.2019.1593964
- Lewin K. (1947). Frontiers in group dynamics: II. Channels of group life; social planning and action research. *Human relations.* 1947;1,2:143-153.
- Luoma, E., Rönkkö, M., Tyrväinen, P. (2012). Current software-as-a-service business models: Evidence from Finland. In *International Conference of Software Business* (pp. 181-194). Springer, Berlin, Heidelberg.
- Martin, B., (2019). Bob Martin presents: The future of agile. Online: <https://www.youtube.com/watch?v=FedQ2NlgxMI>. Accessed on 2021-12-20.

- Massa, L., Tucci, C. L., & Afuah, A. (2017). A critical assessment of business model research. *Academy of Management Annals*, 11(1), 73-104.
- Masters, B., & Thiel, P. (2014). *Zero to one: notes on start ups, or how to build the future*. Random House.
- Maull, R., Brown, P., Cliffe, R. (2001). Organisational culture and quality improvement. *International Journal of Operations & Production Management*.
- McCall, J. A., Richards, P. K., Walters, G. F. (1977). *Factors in software quality. volume i. concepts and definitions of software quality*. General electric. Sunny Vale.
- Myers, M. D. (2019). *Qualitative Research in Business and Management*. Sage, 2019. ISBN 9781526418326
- Naslund, D., Kale, R. (2020). Is agile the latest management fad? A review of success factors of agile transformations. *International Journal of Quality and Service Sciences*.
- van Oosterhout, M. (2010). *Business agility and information technology in service organizations* (No. EPS-2010-198-LIS).
- Osterwalder, A. (2004). *The business model ontology: A proposition in a design science approach*. Institut d'Informatique et Organisation. Lausanne, Switzerland, University of Lausanne, Ecole des Hautes Etudes Commerciales HEC, 173, 2004.
- Osterwalder, A., Pigneur, Y. (2010). *Business model generation—a handbook for visionaries, game changers, and challengers*. Wiley, New York, 2010.
- Putta, A., Paasivaara, M., Lassenius, C. (2018, November). Benefits and challenges of adopting the scaled agile framework (SAFe): preliminary results from a multivocal literature review. In *International Conference on Product-Focused Software Process Improvement* (pp. 334-351). Springer, Cham.
- Rachinger, M., Rauter, R., Müller, C., Vorraber, W., Schirgi, E. (2019). Digitalization and its influence on business model innovation. *Journal of Manufacturing Technology Management*.
- Ries, E. (2011). *The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses*. Crown Publishing Group, 2011.
- Rönkkö, M., Frühwirth, C., Biffel, S. (2009). Integrating Value and Utility Concepts into a Value Decomposition Model for Value-Based Software Engineering. *PROFES 2009*. Springer-Verlag. LNBIP 32, 2009, pp. 362-374.
- Saltan, A., Smolander, K. (2021). Bridging the state-of-the-art and the state-of-the-practice of SaaS pricing: A multivocal literature review. *Information and Software Technology*, Vol. 133, 106510, ISSN 0950-5849.
- Schilke, O., Hu, S., Helfat, C. E. (2018). Quo vadis, dynamic capabilities? A content-analytic review of the current state of knowledge and recommendations for future research. *Academy of management annals*, 12(1), 390-439.

- Shafer, S.M., Smith, H.J., Linder, J.C.: The power of business models. *Business horizons*, 48(3), 199-207 (2005)
- Sjödin, D., Parida, V., Jovanovic, M., & Visnjic, I. (2020). Value creation and value capture alignment in business model innovation: A process view on outcome - based business models. *Journal of Product Innovation Management*, 37(2), 158-183.
- Strauss, A., Corbin, J. (1998). *Basics of qualitative research* (2nd ed.). Thousand Oaks, CA, Sage.
- Thomas, W.I., Thomas, D.S. (1928). *The Child in America: Behavior Problems and Programs*. New York: Knopf.
- Tushman, M. L., & O'Reilly III, C. A. (1996). Ambidextrous organizations: Managing evolutionary and revolutionary change. *California management review*, 38(4), 8-29.
- Uludag, Ö., Kleehaus, M., Caprano, C., Matthes, F. (2018). Identifying and Structuring Challenges in Large-Scale Agile Development Based on a Structured Literature Review. 2018 IEEE 22nd International Enterprise Distributed Object Computing Conference (EDOC). pp. 191-197. doi: 10.1109/EDOC.2018.00032.
- Vanhala, E., Kasurinen, J. (2014). The role of business model and its elements in computer game start-ups, presented at the ICSOB 2014 - The 5th International Conference on Software Business, Paphos, Cyprus, 2014.
- Veit, D., Clemons, E., Benlian, A., Buxmann, P., Hess, T., Kundisch, D., ... & Spann, M. (2014). Business models. *Business & Information Systems Engineering*, 6(1), 45-53.
- Whiteman, W. E. (1998). *Training and Educating Army Officers for the 21st Century: Implications for the United States Military Academy*. Army War Coll Carlisle Barracks PA.
- Wilson, M., Wnuk, K. (2018). Business modeling and flexibility in software-intensive product development – a systematic literature review. In *Conference on e-Business, e-Services and e-Society* (pp. 292-304). Springer, Cham.
- Wnuk, K., Mudduluru, P. (2018). Value-Based Requirements Engineering: Challenges and Opportunities. In *KKIO Software Engineering Conference* (pp. 20-33). Springer, Cham.
- Woodall, T. (2003). Conceptualising 'value for the customer': an attributional, structural and dispositional analysis. *Academy of marketing science review*, 12(1), 1-42.





## **ORIGINAL PAPERS**

### **I**

#### **FOLLOWING THE MONEY: REVENUE STREAM CONSTITUENTS IN CASE OF WITHIN-FIRM VARIATION**

by

Matti Saarikallio & Pasi Tyrväinen, 2014

International conference of software business 2014, June, pp. 88-99.  
Lecture Notes in Business Information Processing, vol 182. Springer, Cham.

DOI: 10.1007/978-3-319-08738-2\_7

Reproduced with kind permission by Springer Nature.

# Following the Money: Revenue Stream Constituents in Case of Within-firm Variation

Matti Saarikallio and Pasi Tyrväinen

matti@saarikallio.net, pasi.tyrvainen@jyu.fi

**Abstract.** The idea of this paper stems from the perception that the concept of revenue stream requires clarification and further division to be applicable to businesses with high internal variation in their methods of capturing revenue. Current study sets out to investigate the concept of revenue stream through an overview of previous literature and a case study to demonstrate how revenue streams of a b2b (business-to-business) software service firm can be analyzed by elaborating the concept further. The aim is to answer the following research questions: 1) What are the relevant constituents of the revenue stream concept within a b2b software services firm? 2) How revenue stream as part of the business model can be analyzed within a firm? This exploratory study contributes to the business model literature by investigating the concept of revenue stream and revenue stream type as managerial tools to better understand the business under investigation. The study further attempts to contribute to the decomposition of the revenue stream concept by exploring its constituents in the context of b2b software business. It is suggested that revenue streams in this context should be approached based on sub-component level analysis where the reason and source dimensions create a matrix of analysis cells from which revenue stream types emerge based on similarities in the method of the revenue streams. Based on previous literature and empirical study, it is further suggested that the revenue stream has three main constituents or sub-components: 1) the source of revenue, 2) the reason for revenue and 3) the method of revenue.

**Key words.** Business model, revenue stream type, software service company, b2b, source of revenue, reason for revenue, method of revenue

## 1 Introduction and Background

### 1.1 Business Model Research

Through experience, business practitioners have mental models about their business, but such mental model can only be communicated and modified once it has been made explicit as a business model [1]. Research about business models has been around for a long time in the domain of software firms. Still, research knowledge about business model is disjointed and unclear [2]. While there is not yet a common understanding, ontologically business model has been suggested to reside in the middle ground between business strategy and business processes [3].

There are various ways to conduct research relating to business models. Research sub-domains can be divided into definitions, components, taxonomies, representations, change methodologies, and evaluation models [4]. The goal of component research is to further decompose the business model concept into its fundamental constructs [4].

The business model concept and its sub-components are used often as a tool to plan and define the business model of new startups. For example Mahadevan [5] uses the term revenue stream to mean the plan for revenue generation. However, business model can also be used to analyze an existing established firm to gain understanding about the de facto business model in place. Such an approach has been taken for example by Rajala, Rossi & Tuunainen [6] in their software business evaluation framework. The idea for the current paper stems from the challenges in analyzing an existing firm's business model's revenue streams when the firm under investigation has multiple customers and offerings with high variability in revenue stream configurations.

### 1.2 Revenue Stream

Most business model conceptualizations include a financial aspect relating to the money that flows into the company. Business model literature is filled with various terms used for these aspect such as: revenue stream, revenue, sources of revenue, revenues, revenue model, revenue mix, revenue side of the business, revenue source, revenue logic, revenue earning logic, revenue mechanism, income model and earnings logic [5][7][8][30][9][10][6][18][11][23][12][13][14][15]. Table 1 summarizes the terms and what they are suggested to mean in the context of business model. The same unclarity that exists for the business model appears to be present for the revenue related sub-components as well. There seems to be a common theme, but not a clear agreement on the terminology.

Zott and Amit [16] have suggested that revenue model complements a business model design in similar way as pricing strategy complements product design. This can be a useful analogy but in the same way as business model is quite an abstract concept when compared to product design, revenue model is very much as abstract compared to pricing strategy. Revenue stream on the other hand seems to have potential to be

defined as a more tangible and measurable object of study as it can be reduced to the concrete idea of money flowing into the company. For this reason of seeking conceptual clarity, this paper focuses on the revenue stream as the main concept of business model and also adopts the approach used in the business model canvas concept suggested by Osterwalder and Pigneur [17].

**Table 1.** There is a multitude of partially overlapping revenue related business model concepts.

Author	Business model component	Description
Mahadevan (2000)	Revenue stream	The plan for revenue generation
Weill, Vitale (2001)	Sources of revenue	Description of source of revenue and how realistic they are.
Alt, Zimmermann (2001)	Revenues	The "bottom line" of a business model.
Stähler (2002)	Revenue model	From what sources in what ways is the revenue generated.
Stähler (2002)	Revenue mix	The sum of all the sources of revenue the firm has.
Magretta (2002)	Revenue side of the business	How is money made in this business.
Afuah, Tucci (2003)	Revenue source	Where is the income coming from, who pays when and for what value and also what are the margins and their drivers for each market.
Rajala et al. (2003)	Revenue logic	The way the software business generates its revenue and profit.
Osterwalder (2004)	Revenue model	The way company makes money through a variety of revenue flows.
Gordijn et al. (2005)	Revenue earning logic	Generating profitable and sustainable revenue streams.
Chesbrough (2007)	Revenue mechanism	How will the firm be paid for the offering.
Rédis (2009)	Income model	Sources of income generated by the company.
Nenonen, Storbacka (2010)	Earnings logic	How the firm yields a profit from its operations.
Schief, Bukmann (2012)	Revenue	Group revenue deals with the pricing model and financial flows.
Ojala, Tyrväinen (2012)	Revenue model	How a firm collects revenue through options that a firm may offer to customers.

### 1.3 Aims of the Paper

Thus, the current study aims to contribute to the research domain of business model components by investigating a sub-component referred to as revenue stream. For example Osterwalder [18] sees the revenue streams as one of the key parts of the business model. He uses the broader term revenue model to mean a collection of revenue streams within a company.

Osterwalder and Pigneur [17] have claimed that a business model can have two different types of revenue streams, namely transaction revenues and recurring revenues. While this is true in simple business models it is highly unlikely that such a simplification is enough to fully explain the revenue stream sub-component of the business model in the more complicated case.

Shafer, Smith, Linder [19] cite a study by Linder and Cantrell [20] which states that 62 % of executives had a difficult time describing how money is made in their company. This could indicate the complexity of the typical revenue models or that there is a lack of proper conceptualization. Either way this supports the relevance of the current paper's interest area.

This paper aims to clarify the revenue stream component by evaluating the revenue model of a case company which has multiple and variable revenue stream configurations and suggest an answer to the question: 1) *what are the relevant constituents of the revenue stream concept within a b2b software services company.* Further the study attempts to answer the question: 2) *how revenue stream as part of the business model can be analyzed within a firm.*

### 1.4 Revenue Stream Framework

Framework to analyze the case study data is suggested based on existing literature. It includes three key parts that must be addressed to explain a revenue stream. These constituents are the source of the revenue stream, the reason for the revenue stream

and the concrete description of the method of capturing the revenue which is called here the method of revenue. This framework builds upon Rajala, Rossi, Tuunainen and Vihinen [21] who suggest that approaches for capturing revenue can have differences in methods of pricing, sources of revenue and the products and services being sold. Similarly in context of business model innovation, revenue model innovations include as key parts offering reconfiguration and pricing models [22]. Chesbrough [23] uses the term revenue mechanism which is by definition comparable to method of revenue. Figure 1 illustrates the suggested model.

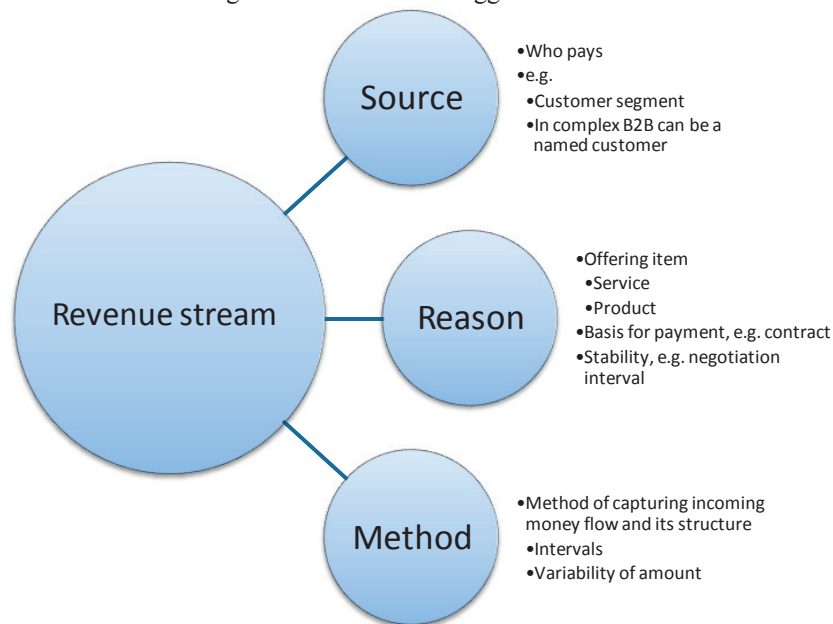


Fig. 1. Suggested decomposition of revenue stream

### 3 Methodology

#### 3.1 Exploratory Case Study

Yin [24] suggests using a case study design when trying to answer how or why questions and attempting to cover contextual conditions relevant to the phenomenon. In the current study attempt is made to understand how revenue stream as part of the business model can be analyzed in the context of a specific b2b software service firm.

When seeking to clarify the concept of revenue stream and related sub-components, it was necessary to analyze the patterns underlying them and it was required to gain an in-depth understanding. Qualitative research approach was chosen

to improve understanding of the investigated phenomenon [25]. The chosen research strategy was a single case study in a company that is considered a representative example, because it had enough complexity and variation in forms of multiple revenue stream combinations. Because a case study research strategy focuses on understanding the dynamics present in a single setting [26], it was a good approach in exploring the business model sub-component and how it can be analyzed in a real-life setting in a within-firm context. Thus, research strategy was that of a single case study. Eisenhardt [26] has suggested that instead of selecting cases at random, extreme examples are appropriate when seeking to extend theory, which is the goal in the exploratory research that this paper undertakes. Because a lot of the existing literature considers cases where there is one revenue model per business model, an extreme example deviating from the norm would be a case with multiple co-existing revenue models and high within-firm variation in the revenue streams. The selected case meets these criteria.

### **3.2 Case Firm**

The chosen case firm operates in the telecom operator software market. This market had only 196 companies offering software product or service offerings in 2006 with a volume just under \$30 billion [27]. Using the terminology from Luoma, Frank & Pulkkinen [27] the firm can be classified as a generic telco vendor. It can be predicted that this kind of firm would have a lot of variation in the revenue streams, because of the breadth of operations.

The analyzed case firm serves telecom operator customers by offering BSS (business support system) solutions. The solutions typically contain a service contract which is one side of the business and making continuous customer specific modifications is an additional way to generate revenue. New customers are a rare occasion and typically some sort of penetration pricing is used for initial deliveries. This is possible due to heavy vendor lock-in that is gained once the delivery is completed. The investigated firm has out of 150 people about 80 working in the investigated business unit. It was established in 1995 and has international customers. Relevant customer count is around ten, but three customers produce majority of the revenue. The firm is organized into customer serving teams with minor common functions. R&D, and marketing and sales departments are manned in ad-hoc manner and no organization exists for these functions. This has given rise to a very variable culture across customer serving teams and most interestingly to this paper it has given rise to a multitude of methods for revenue capture. The complexity of the case makes it a useful context to investigate revenue stream variation.

### **3.3 Data Collection**

The main portion of the data was gathered using semi-structured interviews. Twelve people in corporate and business unit management and account management positions were interviewed to find out the current revenue streams of different customer accounts and the various offerings and revenue capture methods for each. The

interviews lasted from one to three hours each and some were conducted in two separate sessions, because of scheduling challenges. In addition to revenue model specific questions, the understanding of the case was further widened by questions relating to general business model utilizing the business model canvas framework [17]. All interviews were recorded and transcribed. The interviews were scheduled close to each other during a period of one month. Close scheduling was done in order to avoid participants from influencing each others' answers. Some details were clarified by additional short discussions to avoid false interpretations.

In addition to the interviews, access was gained to written materials, mainly contracts and offers made by the case firm. This helped to solidify the actuality of contractual relations with case firm's customers in situations where the informants were unable to remember the details in full.

Data was analyzed using qualitative content analysis method with three analytical procedures of summary, explication and structuring as suggested by Kohlbacher [28]. The transcribed interview data was processed by summarizing the key themes to capture the main ideas from the informants. These themes were then used as a basis for further explication of the data. Dimensions of structuring became apparent from the data and the results are presented within those dimensions.

## **4 Results**

### **4.1 Source**

The collected data indicated that one explanation for the large revenue stream variation was the source of revenue, namely the customer or customer segment. As one informant put it: "Typically if they have an organization change then the desired invoicing [method] changes." Thus, a big factor affecting the revenue stream is the customer and their needs. This can be partially due to unbalanced negotiation power between the parties. The dynamic nature of the customer means that the revenue stream is also dynamic in nature. When the source of the stream is dynamic it is reflected in the revenue stream. Within the case firm five different sources of revenue were detected. Four of them were different medium to large companies. The fifth source was a group of small companies. The group was analyzed together as one revenue source, because there were no differences from revenue stream point of view.

All the revenue streams in the current case were negotiated separately on a customer by customer basis as a whole and in some customer accounts different parties were involved in negotiating the managed services and the software development agreements. Actually having to negotiate the pricing in each revenue stream added to the complexity of the sales process. The lack of a price list was mostly due to lack of product management efforts in general. The extent of customer specific negotiations suggests that the customer will have a great impact on the revenue stream making it a differentiating dimension. The customer negotiation intervals also have an effect on the predictability of revenue.

## 4.2 Reason

While source of revenue was a significant explanatory factor for the variation there were also differences in revenue streams originating from one source and it could be seen that the variation was dependent on the reason for revenue. Reason for revenue can be considered to be the offering item which is the product or service and has in most cases a contractual basis. In the current case 9 different reasons were identified from the interviews. They were: billing manager service, customer care system, order entry system, billing system, keeping the systems running, enterprise resource planning system, system development, consulting/analysis. Additionally the firm offers fixed price delivery projects for new customers before the relationship progresses into so-called operative mode. However, no such delivery was ongoing during the interviews and therefore this aspect was excluded from the study. Focus is on the current customer relationships.

Revenue streams based on different offerings varied in terms of packaging level. The revenue streams whose reason for revenue was system licensing or maintenance service offerings were sold as a complete package. On the other hand those streams whose reason for revenue was system development and customization activities contained various configurations based on customer specific needs.

As mentioned earlier, the source of revenue dimension had a somewhat dynamic nature meaning that the needs change over time. Similarly there was dynamism in the reason dimension. It was clear that the offerings were not static. There was also a preference towards generating one type of revenue over the other. An informant commented that they try to push more towards the model where the development is less and maintenance is more: "It's changing towards the direction where maintenance portion is growing; it also has the best upside, because tools are automated." This indicates that the reason for revenue -dimension is also dynamic in nature.

## 4.3 Method

The method of revenue dimension for each stream had differences across streams but similarities as well. Therefore the analysis within this dimension is more involved. In table two the method is described for each revenue stream that is considered unique. In the current case, each revenue source can be considered to originate unique streams compared to other sources, but multiple reasons can exist for the same stream and those reasons can have the same method, so they are combined here into cells depicted in table two.

## 4.4 Analysis Matrix

It proved useful to present the data in a matrix of reason vs. source where for each cell of the matrix the method of revenue stream was considered. If the reasons were contributing to the same revenue stream they were combined together. This way 11 revenue streams (separate money flows) were identified.



Further looking at these 11 revenue streams and their differences, they could be grouped into four revenue stream types that were considered as unique in the sense that they had a lot of similarities in the method dimension. The four different revenue stream types were given designations A, B, C and D (see table 2).

**Table 2.** Revenue stream types were grouped based on similar structure of revenue

Reason for revenue	Billing manager service	Customer care system	Order entry system	Billing system	Keeping the systems running	ERP system	System development	Consulting/A nalysis
							Stream type C	
Source of revenue	1	Revenue stream 1: Monthly service fee based on amount of subscriptions...				not offered	Revenue stream 6: Projects fee 8 times per year. Every half year a plan for 6 months of work, and after that invoice the extras. Analysis phase invoiced when leading to development. Unused reserved capacity partially invoiced.	
	2	Revenue stream 2: Monthly maintenance fee based on amount of customers with active subscriptions...				not offered	Revenue stream 7: Development fee 8 times per year. Based on hours but adjusted up or down based on the benefit that the customer would perceive they get, breakdown to analysis, development, etc.	Revenue stream 8: Analysis invoiced full-time and separately.
	3	not offered	not offered	not offered	Revenue stream 3: Fixed usage/license fee invoiced monthly.	not offered	Revenue stream 9: Variable development fee invoiced monthly based on worked hours.	Stream type D
	4	not offered	not offered	not offered	Stream type B	Revenue stream 4: Maintenance fee invoiced quarterly in advance. Fixed amount.	Revenue stream 10: Development fee monthly afterwards based on worked hours.	not offered
	5	not offered	not offered	not offered	Revenue stream 5: Maintenance fee invoiced quarterly. Fixed amount.	not offered	Revenue stream 11: Development fee monthly afterward based on worked hours.	not offered

### 4.5 Revenue Stream Types

Stream type A consists of similarly structured revenue streams one and two. For both of them the method of payment is a fixed fee and a per unit price. For the stream 1 the unit is per active subscription and for stream 2 the unit is per end-customer (customer's customer) who has an active subscription handled by the system. In the interviews it was suggested that due to foreseeable changes in the industry the preferred model from vendor perspective was seen to be per service per customer which would better reflect the cost structure and allow the provider to benefit from the new services they might need to support by the system. In general the benefit of the invoicing tied to the growth of subscriptions was seen in having a shared goal of helping the customer grow, because it means more money for the vendor as well. Informant number five commented that "this is the best model I know".

Revenue stream type B included streams where the name people used for the model was different but the formula was the same, so it can be considered one stream type. The terms were either usage fee, license fee or maintenance fee, but they were all basically a fixed amount invoiced at a regular interval, either monthly or by quarterly, in advance or afterwards. Stream types A and B are basically the same in terms of offering: system usage right, and maintenance service. The terminology is interestingly causing problems. Informant six noted: "Because we charge license fee

we have a lot of problems, because they see that they should get monthly development for free." Many people also felt that the future model should be more geared towards per unit based invoicing, because it offers a possibility to move toward value based invoicing away from cost based invoicing. Still, for new customers the downside is increased risks as informant 11 put it: "There is a challenge, because there are not that many of those and for us the cost of hardware doesn't go down. [In case of] minimum monthly payment, the volume can be too low, too much risk." This is one of the reasons why revenue stream type B exists alongside A. It was a safe choice at the initial selling stage.

Revenue stream type C is an interesting one, because it includes a guaranteed minimum purchase. Thus it could be called assured purchase volume and per unit invoicing. The way this is done in practice is that there is a planning session every half a year for the upcoming work which is partially guaranteed work. Informant 1: "Current agreement offers us safety, that we have half a year work at a time. We can invoice 80 percent even if they would order nothing". Otherwise work is invoiced on a per unit price rate where the unit is the amount of worked hours.

Revenue stream type D is a plain per unit invoicing. Compared to stream type C the vendor takes the bigger risk. Pure per unit invoicing was considered easier to sell. The benefits of having an assured purchase volume were seen mainly due to the low transferability of excess capacity between the teams producing the offerings that generate the revenue streams. In the current case this low transferability problem is interestingly solved not by developing the organization but rather creating a revenue stream method that allows it.

**Table 3.** Contributions of revenue reasons to total revenue from each source

Reason for revenue										
		Billing manager service	Customer care system	Order entry system	Billing system	Keeping the systems running	ERP system	System development	Consulting/Analyses	
Source of revenue	1	70 %					not offered	30 %		
	2	45 %					not offered	45 %	10 %	
	3	not offered	not offered	not offered	50 %		not offered	50 %		
	4	not offered	not offered	not offered	not offered	17 %		83 %	not offered	
	5	not offered	not offered	not offered	95 %		not offered	5 %	not offered	

#### 4.6 Revenue Contributions

There was variation between the percentage contributions of revenue reasons to total revenue from each source. Table 3 summarizes these percentages and shows the differences between how much each revenue reason group contributes to the total

income when comparing revenue sources to each other. The variation could be due to the lifecycle of the revenue source and one could guess that a new customer would require more development related activities whereas older customers would only need the service contract. There is initial support for such a conclusion, but the interviews indicated that other reasons like who made the original contract had more effect. Still, the interviews indicated that there was a goal to move away from stream types C and D towards stream types A and B. This was related to the fact that development work is dependent on doing more work: "In the development side the upside will not be very high. It always includes a lot of work." There was an element of unpredictability about future revenue. The fact that the buyer can decide upon buying something or not was seen bad and offering as a packaged service was preferred: "Rather predictability is better, so service fee [is preferred]." It could be said that revenue contributions overall are more likely to move towards the service oriented stream types A and B over time.

In sum, the undertaken analysis approach helped clarify the revenue stream variation within the case firm and gave support for the decision makers' business model understanding. During the interviews one of the informants had commented: "It's hard to tell which revenue stream contributes what. Because it seems the money goes into one bucket." Introducing the revenue stream type analysis can be the first step to alleviate the situation and help the firm in strategic decision making.

## 5 Discussion

The goal of this paper was to conduct exploratory research and answer the question about the constituents of revenue stream. The study suggests that a revenue stream has three main sub-components which are the source of revenue stream, the reason for the revenue stream and the method of revenue stream.

**Table 4.** Sub-components of revenue stream

Component	Definition	Examples
Source	The originating source of revenue flow from whom does the money come from.	Specific customer, customer segment, consumer segment.
Reason	The reason(s) why someone is paying the money.	Offering item, service or product, contractual relationship.
Method	The method of how the payment occurs and how it is structured.	Paid every month based on amount of worked hours with a minimum invoicing.

The second question to answer was how the revenue stream can be analyzed within a firm. It has been stated in previous literature that a business can produce one or more revenue streams from each source customer segment [17]. In this paper it has become evident that in a complex b2b setting, one revenue stream can be caused by several reasons of revenue each having different methods of revenue. In addition varying revenue streams can originate from similar sources. In the current b2b case the complexity was such that it was confusing to try to explain it without a clear structure or fit it into a too abstract model. It is suggested that revenue streams should be analyzed so that the method of getting paid is considered for each cell of a two dimensional matrix having two axis: source of the revenue stream and reason for the revenue stream. Only after this kind of analysis can the similarities in method of capturing revenue between the streams warrant a recombination into revenue stream types with similar attributes. Osterwalder [18] uses term stream type very broadly to mean type of economic activity used to generate income. Stream type is also often

reduced to just listing examples such as: selling, lending, licensing, transaction cut and advertising [29]. This paper suggests, however, that this simplification is not necessary or even applicable in the current case and a revenue stream type within a firm should be defined based on a comparison of methods of revenue viewed through a source by reason matrix. Thus, it is suggested that a revenue stream type describes the method of revenue for streams originating from a similar revenue source for a similar reason for revenue. Further a full explanation of a revenue model means describing all the revenue stream types used.

Based on the empirical analysis the following hypothesis is suggested for future testing: When analyzing the revenue streams of a business model, it is necessary to analyze them separately based on source (from whom does the revenue originate from?) and reason (on what offering is the invoicing based on?) dimensions. Further it is suggested that analyzing the method of revenue within these “source-reason” cells allows the detection of unique revenue stream types which define the nature of the business model in regards of revenue. This kind of matrix cell representation is suggested to describe the firm’s revenue mix much better than for example the revenue mix concept of Stähler [30] which is defined as the sum of all sources of revenue the firm has. It is suggested that revenue mix concept should rather be a description of revenue stream types in all three mentioned dimensions not just the one.

Because of the demonstrated incoherence of revenue aspects relating to business model in the existing literature, a decomposition of the revenue stream concept was attempted. This paper provides support to the usefulness of the concept revenue stream and suggests its applicability also to the analysis of b2b software service businesses. Because only one specific context of b2b software service business was considered, further study should be made in other contexts to compare the findings and investigate the suggested decomposition to enable more general theoretical propositions. Here the context was b2b, because of the case selection, but it might be possible to expand the findings towards b2c in the future.

This paper contributed to the business model research by defining three constituents of a revenue stream and introducing the concept of revenue stream type as a combination of revenue streams with similar method of revenue. For management practitioners a tool was presented for analyzing revenue aspects of the business model. The presented decomposition could be used when investigating for example the profitability of different revenue streams to gain a more fine grained analysis. Managers can use this systematic approach to better understand the business and describe and visualize the revenue streams involved.

The suggestion for future business model research is to promote the money flow i.e. revenue stream as the central concept around which an analysis of a business model should be built upon, because a business by definition has to generate revenue in order to be viable on the long term. Therefore, following the money is a good idea.

## **References**

1. Petrovic, O., Kittl, C., Teksten, R.: Developing business models for ebusiness. Available at SSRN 1658505 (2001)

2. Al-Debei, M.M., Avison, M.: Developing a unified framework of the business model concept. *European Journal of Information Systems*. 19(3), 359-376 (2010)
3. Osterwalder, A., Pigneur, Y.: An e-business model ontology for modeling e-business. In: 15th Bled Electronic Commerce Conference, pp. 17-19. Bled (2002)
4. Pateli, A., Giaglis, G.: A framework for understanding and analysing e-business models. In: Bled Electronic Commerce Conference. Bled (2003)
5. Mahadevan, B.: Business models for Internet-based e-commerce. *California management review*, 42(4), 55-69 (2000)
6. Rajala, R., Rossi, M., Tuunainen, V.K.: A framework for analyzing software business models. In: ECIS, pp. 1614-1627 (2003)
7. Weill, P., Vitale, M.R. *Place to space: Migrating to eBusiness Models*. Harvard Business Press (2001)
8. Alt, R., & Zimmermann, H. D.: Preface: introduction to special section—business models. *Electronic Markets*, 11(1), 3-9 (2001)
9. Magretta, J.: Why Business Models Matter. *Harvard Business Review*, 80(5), 86-92 (2002)
10. Afuah, A., Tucci, CL: *Internet Business Models and Strategies: Text and Cases*. McGraw-Hill, Boston (2001)
11. Gordjin, J.: Comparing two business model ontologies for designing e-business models and value constellations. Technical report, 18th Bled eConference eIntegration in action (2005)
12. Rédis, J.: The impact of business model characteristics on IT firms' performance. *International journal of business* 14.4: 291-307 (2009)
13. Nenonen, S., Storbacka, K.: Business model design: conceptualizing networked value co-creation. *International Journal of Quality and Service Sciences*, 2(1), 43-59 (2010)
14. Schief, M., Buxmann, P.: Business models in the software industry. In *System Science (HICSS), 2012 45th Hawaii International Conference on* (pp. 3328-3337). IEEE (2012)
15. Ojala, A., Tyrväinen, P.: Revenue models in cloud computing. In E. Prakash(Ed.), *Proceedings of 5th Computer Games, Multimedia & Allied Technology Conference (CGAT 2012)* (pp. 114-119). Singapore: GSTF (2012)
16. Zott, C., Amit, R.H.: Designing your future business model: an activity system perspective. *Social Science Research Network Working Paper Series* (2009)
17. Osterwalder, A., Pigneur, Y. : *Business model generation—a handbook for visionaries, game changers, and challengers*. Wiley, New York (2010)
18. Osterwalder, A.: *The business model ontology: A proposition in a design science approach*. Institut d'Informatique et Organisation. Lausanne, Switzerland, University of Lausanne, Ecole des Hautes Etudes Commerciales HEC, 173 (2004)
19. Shafer, S.M., Smith, H.J., Linder, J.C.: The power of business models. *Business horizons*, 48(3), 199-207 (2005)
20. Linder, J., Cantrell, S.: *Carved in water: Changing business models fluidly*. Accenture Institute for strategic change (2000).
21. Rajala, R., Rossi, M., Tuunainen, V.K., Vihinen, J.: Revenue Logics of Mobile Entertainment Software-Observations from Companies Producing Mobile Games. *JTAER*, 2(2), 34-47 (2007)
22. Giesen, E., Berman, S.J., Bell, R., Blitz, A.: Three ways to successfully innovate your business model. *Strategy & Leadership*, 35(6), 27-33 (2007)
23. Chesbrough, H.: Business model innovation: it's not just about technology anymore. *Strategy & leadership*, 35(6), 12-17 (2007)
24. Yin, R.K.: *Case study research: Design and methods* (Vol. 5) Sage (2003)
25. Yin, R.K.: *Case study research: Design and methods*. 2nd edition. Applied social research method series, vol 5. Sage publications (1994)
26. Eisenhardt, K.M.: Building theories from case study research. *Academy of management review*, 14(4), 532-550 (1989)

27. Luoma, E., Frank, L., Pulkkinen, M.: Overview of telecom operator software market. In Vertical Software Industry Evolution (pp. 35-42). Physica-Verlag HD (2009)
28. Kohlbacher, F.: The use of qualitative content analysis in case study research. In Forum Qualitative Sozialforschung/Forum: Qualitative Social Research (Vol. 7, No. 1) (2006)
29. Scheithauer, G., Wirtz, G.: Business modeling for service descriptions: a meta model and a UML profile. In Proceedings of the Seventh Asia-Pacific Conference on Conceptual Modelling-Volume 110 (pp. 79-88). Australian Computer Society, Inc. (2010)
30. Stähler, P.: Business models as a unit of analysis for strategizing. Online: <http://www.scribd.com/doc/34770740/Business-Models-as-a-unit-of-Analysis-for-Strategizing>. Referenced on 7.1.2013 (2002)



## II

### **BUSINESS MODEL ELEMENTS IN DIFFERENT TYPES OF ORGANIZATION IN SOFTWARE BUSINESS**

by

Erno Vanhala & Matti Saarikallio, 2015

International Journal of Computer Information Systems and Industrial  
Management Applications vol 7, 139-150

Reproduced with kind permission by Machine Intelligence Research Labs.

# Business model elements in different types of organization in software business

Erno Vanhala<sup>1\*</sup> and Matti Saarikallio<sup>2</sup>

<sup>1</sup> Software Engineering and Information Management, Lappeenranta University of Technology.  
P.O. Box 20, FI-53851, Lappeenranta, Finland  
erno.vanhala@lut.fi

<sup>2</sup> School of Computer Science and Engineering, University of Jyväskylä  
P.O.Box 35, FI-40014, Jyväskylä, Finland  
matti.saarikallio@umbergames.com

**Abstract:** The business model concept has been discussed widely during the current millennium. On one hand most of the discussion is not academic in nature and on the other hand the industry practitioners have been rarely included nor have their voice been heard in the academic studies. This has led to differences in definitions of the business model concept between academic studies and the thinking of industry practitioners. In this study we dive into practitioners' views and investigate how they fit to business model canvas, a tool that is now popular in business practice. We also investigate how different types (in terms of age and size) of organizations working on different software business fields utilize the business model concept in their own ways. The findings showed variation in how different organizations promote different elements of the business model as more important and how the elements included different content even within the software business domain. We also demonstrate some of the similarities that prevail in software business, such as people being the key resource, regardless of the field or type of business.

**Keywords:** Business model, software business organizations, established vs. entrepreneurial, startups, B2B vs. B2C, business model canvas

## I. INTRODUCTION

The business model concept has gained more and more popularity in the scientific literature in this millennium. It can be used in multiple scenarios, like designing a new venture or analyzing and developing an existing business further. The goal of this study is to investigate business model as a useful theoretical construct in both cases and look into the differences and similarities that arise from the different viewpoints on the concept.

The existing literature has been discussing the definition of business model [1], [2] and its usefulness for the software industry [3], [4]. The role of business model concept has been

studied in various studies in various industry fields (e.g. [5]–[7]). The current research lacks the organizational point of view discussing how the business model concept is understood by the industry practitioners and how the organization can utilize the business model to help their business [8], [9].

In this study, we concentrate on focal firms, which are working in the software business domain. We compare the role of business model for startup organizations – the ones in their early stages moving from idea to product and improving operations and securing financing [10], [11] – to organizations that have been in the field for years. Besides the organization age, we also compare the size of organizations from micro entities to small entities being part of medium sized organization. As the third comparison unit we used the business type and field. The aim of this study is to identify how people consider the concept of business model in different sized and aged organizations doing different type of business in different software industry fields.

The current scientific literature has been arguing over the definition of business model concept [1], [3] and the uniform definition is yet to be found, although competent frameworks and models have been developed. In this study we dive into an investigation about how industrial practitioners experience these models.

Based on these ideas we build our research question as following: *How is the organization or business type reflected in the emphasis of the business model elements in software firms?*

In this study we select a new perspective where we compare the business models of organizations having differences in size and age of organization, and field and type of business. This study is combination and extension of earlier studies by the authors. See [12], [13] for reports on the individual



findings. This paper focuses on reanalyzing the data and comparing the findings from this new perspective.

## II. RELATED RESEARCH

Information technology is still a special industry due to the speed of technological development. Baden-Fuller and Haefliger [14] have pointed out that technology development facilitates new business models. Therefore it is particularly interesting to consider software business as the environment in which business model research can provide new insights.

Business model can be considered as a combination of three streams, the value stream, the logistical stream and the revenue stream. This viewpoint presented by Mahadevan [15] considers the value stream as identifying the value proposition, the logistical stream identifying the choices made about the supply chain, and the revenue stream identifying the plan of how the business generates revenues. Business models also reflect the operational and output systems of the company and they capture the way the firm operates and creates and delivers value to customers and mutually converts received payments to profit [16]–[19]. The overall definition of business model could be described: to define who is offering what to whom, how the offering is produced and what is expected in return.

Especially in the fields of information technology and software business the concept of business model has given a powerful and much used tool for analyzing, developing and understanding businesses more thoroughly. Business model has been suggested to reside in the middle ground between business strategy and business processes [3], [17]. The concept of business strategy is identified as a more abstract way of positioning an organization in the business field and business process is categorized as a more operational level with its detailed descriptions of operations. This segmentation is also supported for example by [6], [20], [21]. The concept of business model should not be thought of as a process, but merely description of the steps and key items [22], [23].

Some scientific studies use the term component [1], [6], [23], while some talk about elements [7], [24] when they refer to building blocks of a business model. They are still talking about the same thing: parts that form the unique business model as the concept of a business model is more of an umbrella term to these various sub-parts. In this article we have chosen the term element to describe what combines to a business model.

Shafer et al. [25] have suggested that business model elements should be classified into four primary categories: strategic choices, the value network, creating value, and capturing value. In this paper we take the stand that strategic choices do not belong as part of the business model concept, but should be discussed as part of strategy instead. Thus, we do not include it as an element.

Numerous studies defining the concept of business model identify elements that are characteristic to this concept [6], [7],

[23], [24]. There exists variety in both number and definition of elements, but the most commonly used ones include for example value production, customers and the revenue model.

Table 1 summarizes the existing literature of business models and lists the different elements found in various studies. While there is a difference in the wordings and which parts are considered more important to include in the business model, there is still an emerging consensus that similar elements are included in the concept of business model.

Study	Elements
Timmers [26]	an architecture for the product, service and information flows, potential benefits, sources of revenues + marketing strategy
Alt and Zimmerman [27]	mission, structure, processes, revenues, legal issues, technology
Rajala et al. [28]	product strategy, revenue logic, distribution model, service and implementation model
Shafer et al. [25]	strategic choices, the value network, creating value, capturing value
Chesbrough [29]	value proposition, target market, value chain, revenue mechanism(s), value network or ecosystem, competitive strategy
Al-Debei and Avison [3]	value proposition, value architecture, value network, value finance
Osterwalder and Pigneur [30]	customer segments, value propositions, channels, customer relationships, revenue streams, key resources, key activities, key partnerships, cost structure
Weiner and Weisbecker [24]	value approach, market interface, products & services, value creation & capabilities, financial aspects
Schief and Buxmann [7]	main categories: strategy, revenue, upstream, downstream, usage

Table 1. Elements of business model in different studies

Recently a summary of business model elements presented by Osterwalder et al. [31] has gained popularity. Their business model canvas (BMC) offers a summing-up of most of the elements that are discussed in the literature as essential parts of the business model theory. This paper takes the BMC element division as the main theoretical framework under investigation, because it is now quite well known in industry in Europe. A quick Google trends search with the term "business model canvas" reveals the trend continues and is gaining popularity.

There are various ways to conduct research relating to business models. Research sub-domains can be divided into definitions, components, taxonomies, representations, change methodologies, and evaluation models [32]. This paper focuses on contributing to the elements (components) area of

research as the goal is to compare how the elements are recognized in differing types of software business organizations.

The business model concept has been studied in various business areas – like health-care [5], airline business [6] and software business [7]. Software business has its own peculiarities not found from other fields of business as it builds intangible products and services that a user cannot experience directly but only through user interfaces [33]. In a systematic literature study conducted by Vanhala and Smolander [9] it was concluded that there were several articles available describing particular areas of the software business, for example, revenue and pricing issues, how the software-as-a-service paradigm is changing the business, what open source and mixed source mean to the business model and what are the difficulties when a IT company is expanding to overseas. The study conducted by Vanhala and Kasurinen [12] shined a light on how startups recognize the business model concept, but their study was limited only to this area and no comparison of startups and established organization doing business was found.

As stated earlier, the current paper agrees with the BMC [30] understanding of the business model concept and considers the following elements: value proposition, customer segment, customer relationship, channel, revenue stream, cost structure, key resources, key activities, and key partners. Conceptually we argue that a business model is described through a description of these sub-concepts and their interactions.

### III. RESEARCH PROCESS

This study follows an adapted version of the multiple case study research method [34], [35] and the framework developed in Gable [34]. In the framework six steps are presented: defining the strategy, reviewing the literature, developing the case study protocol, conducting a pilot case study, conducting a multiple case study, and developing a conceptual model. The strategy is determined by our research question presented earlier. The literature was reviewed in the Related research section, besides the original articles [12], [13], and the computer game business model literature has been systematically reviewed by Vanhala and Smolander [9]. As this study relied on existing interview data, the case study protocol was build on over the idea that interview themes from two individual study match each other. The analysis produced

a conceptual model, presented in the Findings section. To guarantee the validity of the results, we followed principles derived from Gable et al. [34]–[36]. This included for example choosing the data collection procedures (we used thematic interviews), data analysis methods (we used coding) and avoiding being biased (we had more than one researcher discussing the interviews and conducting the analysis of the collected data).

Data was analyzed using qualitative content analysis method with three analytical procedures of summary, explication and structuring as suggested by Kohlbacher [37]. The transcribed interview data was summarized to key themes in order to capture the main ideas from the interviews. Themes were grouped based on the theoretical framework and described in the light of the framework. Structuring of the data was based on comparing the results across the different organization and business types.

#### A. Data Collection

The data was gathered through semi-structured interviews totaling twenty-three people in business unit, account management or technical management positions as well as CEOs and owner managers. All interviews were recorded and transcribed. Some details were clarified by additional short discussions to avoid false interpretations. On some cases there were more than one interviewer present and they could discuss the interview topics later on, in order to avoid any misunderstandings.

We wanted to compare different types of organizations and this lead us to choose the firms so that they included both startup and established organizations, medium-sized and micro-sized ones, and organizations with different business types and field. This enabled us to compare them and find differences that could lead to interesting findings. We chose micro-sized companies and small organization units being part of a medium-sized organization, because they are quite close to each other but distinct enough to improve the likelihood of finding differences. It is easier to study the business model in a more manageable sized organization. In large organizations things like processes, organization structure and competitive strategy are likely to become more relevant and we therefore consider business model in that context a less interesting target of research.

The current multiple case study takes a new analysis viewpoint and is therefore original research although the data

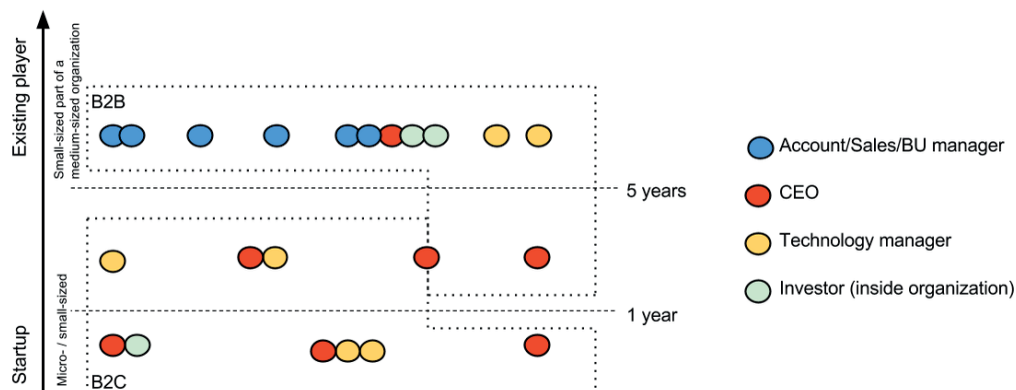


Figure 1. Positions of interviewed persons reflected to age of the organization they are working in. Markers side by side imply they are part of the same organization.

is based on two distinct data sets gathered in 2012 and 2013, and which were partially reported in previously published studies by the authors Saarikallio and Tyrväinen [13] and Vanhala and Kasurinen [12]. The report by Saarikallio and Tyrväinen [13] only utilized the revenue related interview material from the data set, and rest of the data was unpublished. The multiple case study conducted by Vanhala and Kasurinen [12] focused only on startup organizations thus lacking the more general approach to business models in other types of software businesses. As this shortcoming was already noted by Vanhala and Kasurinen [12], the study required an extension. For the current research we combined the two original studies with the case organizations found in both. The themes in both interviews were identical. Both interview sets utilized the same theoretical framework as the basis of data collection thus enabling the use of a combined data set. One additional interview was conducted in autumn 2014 to further enrich the data. The role of the informants and the age of their organizations are presented in Figure 1.

Table 2 presents the case organizations and their key statistics. Organization can be defined as a carefully constructed system, that has the task to reach the goals it has been set [38].

	Size* of an organization	Years in businesses	Field of business	Released products / Finished projects	Type of business
Case A	3	<1	Mobile games	The first one being developed at the moment	B2C
Case B	4	<1	Mobile and browser games	The first two being developed at the moment	B2C
Case C	4	<1	Mobile games	The first one being developed at the moment	B2C
Case D	3+1 half-time	<2	Browser and mobile games	1	B2C
Case E	8	<3	Mobile and PC games	2	B2C
Case F	4	<2	Serious games for health-care purposes	2 projects being developed at the moment	B2C / B2B
Case G	2	<3	Browser-based software	More than 15	B2B
Case H	25	>5	Telco vendor		B2B
Case I	5	>5	Telco vendor		B2B
Case J	8	>5	Telco vendor		B2B
Case K	2	>5	ERP vendor		B2B

Case L	3	>5	Telco vendor	B2B
Case M	5	>5	Telco vendor	B2B

Table 2. Description of case organization.

### B. The selected elements for comparison

The themes of the semi-structured interviews were on both original research projects based on elements presented in business model canvas (BMC) presented by Osterwalder and Pigneur [30]. The business model canvas is therefore the underlying framework for this study. Based on the interview forms we compared the questions and found out that the answers provided data to be utilized in this study. Table 3 illustrates the interview questions compared to BMC elements. Some of the questions differ, because of the domain of the interview. The questions were selected to elicit the thematic discussion only and are the starting point of discussion about the theme not the only thing asked.

Comparison criteria	Specific data collection questions in established business model study	Specific data collection questions in startup business model study
Channels	How are we reaching our customer segments now?	What are the ways/platforms used in delivering games to customers?
Cost structure	What are the costs in the business model? Which resources or activities cost the most?	How would you describe your cost structure?
Customer relationships	What type of relationships do our customer segment expect us to establish and maintain with them?	How do you maintain customer relationship?
Customer segments	Who are your most important customers?	Who is your customer?
Key activities	What key activities does the business model require?	What are the most important key activities you do in your company?
Key partners	Who are your key partners? Who are your key suppliers? What resources or activities do you get from partners?	What/who are your key partners?
Key resources	What key resources do your value propositions require?	What are your key resources?
Revenue streams	For what value do the customers pay? How do they pay? How much does revenue stream contribute to overall revenue?	What do you base your revenue stream on? Which party is the main source for income?

Value proposition	What value do you produce to the customer? What problems do you help to solve? What customer needs do you fulfill?	What is that what you give to the customer?
-------------------	---	---

Table 3. The comparison criteria used in this study to map differences and similarities of business model viewpoints.

#### IV. FINDINGS

We went through the data and utilized business model canvas [30] as our framework when comparing the organizations. Table 4 presents the themes that were identified from individual studies. The identification was conducted by coding the transcribed interviews and interpreting the outcome of the coding by each researcher by himself. After that the differences were discussed between conductors of the original studies and differences are presented in Table 5.

Business model theme	Themes emerging in established organization interviews	Themes emerging in startup organization interviews
Key partners	Data center services, software tools, hardware, licenses, IT support, service level agreements chaining, software development skills, monitoring and maintenance	Publisher, outsourcing (both ways) partners (art studios, musicians, marketing), companies in the same building, B2B contacts
Key activities	Customer relationship management, managing the people, contract and financial activities, software development, software delivery, system maintenance, support activities	Innovating game design, programming, testing, graphical designing, 3D modeling, music and sound producing, getting funding, communicating with customer
Key resources	People (developers, analysts, administrators, account managers), money, contracts, customers to provide requirements, code, hardware	People, intellectual property (brand), office, computers, special software
Value propositions	Flexibility, outsourcing, reducing capex, staff reduction, time-to-market, securing revenue, customer tailored products and services, tools for customer service, sales, invoicing	Providing entertaining experience, on serious gaming: to improve the healing process/ to provide reduce cost to health-care organizations; to provide services that improve the business of a client (B2B)
Customer relationships	Dedicated personal relationship handled by account manager, business analyst, project manager, and service manager. Relationship stage dependent cost of maintaining the relationship. Sales cost is very low. Disagreement on the depth of relationship, some considered it a partnership, some supplier relationship.	Getting direct and indirect feedback from customers (discussing with customers), discussing with physically present testers

Customer segments	European greenfield retail mobile virtual network operators, e-invoicing and staffing companies	geographical segmentation, translations
Channels	Personal relationships and contacts, customer specific ad hoc channels, good customer reference	Towards customers: app stores, word-of-mouth; from customers: social media, app store feedback fields, discussions with a client
Cost structure	Salaries of development people, service people, general people; servers, license costs, data center rent; office space rent, tools, computers, software	Salaries, rent, computers, special software
Revenue streams	Service/maintenance/hosting fee, licenses, deliveries, consulting, development	Income from selling products (pay-to-play), income from advertising and in-app-purchasing (free-to-play), grants, outsourcing work

Table 4. Themes emerging from interviews

Business model theme	Differences and themes
Key partners	In the larger organization other organization units were sometimes categorized as partners, information sharing was more open in small ones, backend server hotel was identified as partner in the older organization and as a resource in the startup
Key activities	The startup had the acquisition of finance as a special feature, managing the people did not come up in any of the startups, neither did they have any metrics, there was the difference of business analyst vs. innovation (&feedback), artistic tendency vs. efficiency & functionality, calendar invites vs. morning coffee
Key resources	Human capital is the most important resources for all the organizations. Older organizations view people more as role-based, whereas startups have generalists. The computer game startups focus on building a brand, and IPR are important. In B2B customer is the product owner and contracts are mentioned as key resources. IPR is less important for B2B, because the main portion of money comes from changes, not an existing product.
Value propositions	It seems a new organization tends to take the general value proposition in the field of business and that is it. Inside older organizations the thinking is sourced from a much larger palette when considering what is our value proposition. There is an interesting difference, because the goal of a game is to get customers to spend as much time as possible, and in comparison other software is targeted to speed up or automate a process, which is especially true in B2B vendors.
Customer relationships	The customer relationship was done almost completely online in computer games business and indirect data was collected from games. B2B organization had more intimate relationship with their customer as they used phone and had physical discussion. As organization grows older it can have dedicated person to handle customers.
Customer segments	In the case of a young organization the customers were considered as a whole. Segmentation was based mostly on a technical platform. In the established organization there was

	a tendency to think about positioning in the industry and how to find their own segment. For the startups segmentation was more as a cost due to translation expenses and such.
Channels	Here the bigger contributing factor is most likely the B2C vs. B2B difference instead of organization age.
Cost structure	Although salaries were the biggest cost in all cases, there was more division into e.g. development team vs. service/maintenance team in larger organizations. Managerial structure was bigger in older organizations.
Revenue streams	None of the new organizations had innovated their own revenue model.

*Table 5.* Identification of emerged differences and themes from different organization groups.

In the cases under investigation, the source of the differences in the business model elements can be identified to arise from three dyads of differences within the cases. Those are the established vs. startup difference, business to business vs. business to consumer orientation and micro vs. small to medium sized organization. In addition there are differences which arise from the field of business (games, telco vendor, serious games for health-care, web software product).

Key partners were recognized differently between our startups and established organizations. Within the established organizations other organization divisions were identified as partners whereas the startups mentioned other companies in the same building as partners. People in established organization felt that the in-house knowledge is a major player but in the startup employing only few persons the partnership is bond with other companies sharing the same office space and coffee table. Among the bigger organizations data center service providers and software tools and platform vendors (to some degree also open source community) were sometimes considered as partners while startups mentioned back-end solutions as resources.

Key activities was one of the most distinct elements. The founder groups of startups were concentrating on developing their products, supporting funding and communicating with customers. They had issues of getting funding, which had already solved out in the established organizations. The processes of established organizations were more formal and for example managing the people was listed as one key activity. This was not necessary in startups atmosphere as the whole workforce shared the same coffee table and all the management could be done there. Startup computer game organization did not either utilize any metrics to measure productivity or presence of people while established organizations were relying at least attempting to measure the worked hours. The same happened also with Case G, which was doing B2B. Although they did not have any systematic process to measure hours they needed to charge something from the customer and thus they had some hour accounting.

The established organizations included a business analyst role, whose task is to gather software requirements from the

customer. Startups did not have these kind of people but they relied on innovative game designers and feedback from customers. Regarding the key activities element, the startups operated in a more artistic way which they considered effective operation mode. Still, the startups did have innovation days, so there was an initial innovation process forming. The clear difference is that with established organization there was a bigger role for a specific person discussing with the customer, and that person received customer needs directly. Within the game companies their artistic tendency is visible in activities. Requirements are not gathered from customers, but all features are more or less the product of a creative process. Thus the difference boils down to innovation vs. requirements gathering as a key activity.

Informants in both – startups and established organizations – valued their people as the most important key resource. The difference was that in established organizations people had more specific and defined roles whereas startups only had people that did certain required tasks. Basically in the startups the whole workforce was capable of developing the product and no special analysts, administrators or account managers existed. People in the established B2B organizations argued that contract negotiations with their customers are important as the contract is identified key resource that enables income. For B2C startups contracts were not made with customers instead intellectual property rights were identified important and the brand building was started as soon as the company was founded.

The value proposition in computer game startup was straightforward: to provide an entertaining experience. Organization managers did not consider any other value they were providing but concentrated on providing entertaining games. The startup (Case F) working with health-care organization was aiming to provide products that would cut the time it takes patients to get back in fit after physical injury. This leads to reduced costs for their customers – health-care organizations. The B2B startup (Case G) aimed to give simple and fast service so that customer needs to spend as little as possible to bureaucracy. For the established organizations value propositions varied. The B2B model is clearly identified as customer tailored products are mentioned as one of the main value the organizations produce. Also several different ways to provide value were mentioned. In our cases startups take the general value proposition of the industry instead of developing their own whereas established organizations have pondered the value proposition more and want to stand out from competitors through it.

The field of business affects how the value proposition is constructed. Typically software is built with a goal to minimize the time user needs to spend with the task but as pointed out by Vanhala and Kasurinen [12] computer games try to do the opposite; to maximize the time spent and still keep it entertaining.

In the established B2B organizations customer



relationships were handled through dedicated personnel in different levels of business collaboration. Some of their customers identified them as a partner and some identified them as a supplier. B2C startups handled their customer relationship through getting direct feedback via online services and indirect data collected from customer sessions. The only physical form of communication was when they had the opportunity to give a test device with their game to some random potential user.

In the established organizations customer segmentation is valued and it played a role in their business model while computer game startups did only geographical segmentation through translations of games. In the established organization deep discussion were held about how the organization finds best segment for its products. It was part of their business model as an improvement element, whereas computer game startups mentioned segmentation – translation – as a cost. They were developing products for global markets.

As established organizations were working with B2B projects their main channels to reach customers were personal and customer specific while computer game startups, with B2C model, were mainly reaching customers through online media like app stores and social media. The role of customer references was recognized important when doing B2B projects also with the B2B startups. In the B2C organizations the feedback in the app stores plays a role as it shows the value of the product.

The cost structure element of business model mostly consists of labor cost which is often the case as software industry produces intangible products. Besides labor cost, companies had also costs from hardware, software licenses and office rent. The difference between established organizations and startups was the diffusion of cost between different human resource groups. The established and larger organizations divided the costs into several groups whereas startups had just general labor cost. The amount of organizational structure increases when organization grows and creates different levels of management and supporting services.

Various revenue streams were found from business model of established organizations. Revenues were based on service, maintenance and hosting fees and also licenses, deliveries, consulting and development. Computer game startups based their revenue streams on generally used models: selling games, selling in-game material and advertising. CEOs of startups also mentioned that they have build their products with money gained from grants and had done some work for other companies too. The difference found was the fact that none of the startups – neither B2C nor B2B – had innovated new revenue models whereas the established organizations had built several individual revenue models and linked them as they would fit best. The B2B model gained revenue from maintenance and changes to software while B2C earned income directly or indirectly from the products.

### A. Mapping Organizational Differences

In addition to Table 5, Table 6 presents the mapping of the organizational differences based on the different elements presented in BMC. We found differences emerging from the business type and field, from the age of organizations and from the size of organizations.

	Business type and field	Age of the organization	Size of the organization
<b>Channels</b>			
Existing reseller channels to reach customers	B2C, Games		
<b>Cost structure</b>			
Increased amount of organizational structure		Established	Medium
<b>Customer relationship</b>			
Dedicated personal assistance	Telco vendor		
Self-service (sometimes community)	Games		
Anonymous data is collected from games to respond the gamers' problems	Games		
<b>Customer segments</b>			
The importance of segmentation	B2B Telco vendor		
The cost of a segmentation (translations)	B2C, Games		
<b>Key activities</b>			
Personal communication to reach customers	B2B Telco vendor		
Measurements on productivity	B2B Telco vendor	Established	Medium
Innovation and artistic way of doing things	Games		
Analytical way of building business	B2B Telco vendor, games		
IPR is important	B2C, Games		
Building/marketing brand from the beginning	B2C, Games		
<b>Key partners</b>			
Other organizations are identified as partners	Games	Startup	Micro
Other division are identified as partners		Established	Medium
<b>Key resources</b>			
People as role-based		Established	Medium
People are generalists		Startup	

Different levels of management		Medium
Management done in a coffee table	Startup	Micro
<b>Revenue streams</b>		
Dependency on external funding (grants, venture capital, loan)	Startup	
Revenue is earned with maintenance	B2B Telco vendor	
<b>Value propositions</b>		
Industry level general value proposition	Games	Startup
To provide entertaining experience (to increase the time spend with the product)	Games	
Software aims to automate processes	B2B Telco vendor	

*Table 6.* Concepts emphasized within business model framework categories mapped to type and field of business, age of organization and/or size of an organization.

Business type and field that the analyzed case organizations were involved in gave rise to multiple differences. In case of the B2B Telco industry part of the revenue was earned through maintenance, the goal of the developed software was to automate business processes or operative processes, customer segmentation was considered very important, personal communications was the main way to reach customers and measurements on people's productivity was collected.

People in organizations, involved in the games industry, were identified as more artistic and the aim of games was the opposite to B2B products that were aimed to improve efficiency and shorten the time spent in specific task. The aim of games was to provide entertaining experience thus increase the time spent with the game.

The games industry organizations had some tendencies that are considered to arise from their involvement in the B2C product business. As the games industry organizations typically target global markets, they tended to view segmentation as a cost, not a customer strategy. The cost is mostly related to translation expenses. The interviewed organization representatives also stressed the importance of IPR which was not considered as important in service business. The product based business and consumer market combination also lead the organization to focus on marketing and building a brand from the start.

It seems that the age of the organization tends to have an effect on differences in business model as well. We noticed that the startup organizations tend to create partnerships with other small organizations, there is a dependency on external funding, people who work there tend to be more of a generalist in their work roles, and the little specific management there

exists is generally conducted informally around the coffee table. On the other hand in established organizations the people in the investigated organizations identified other divisions of the organizations as partners, the amount of organizational structure was larger, productivity statistics were measured, and people identified themselves with specific roles in their work. For example a role could be a tester, programmer, agile coach, project manager, service manager, and such.

Size of the organization was also one culprit for the observed differences. Small organizations tended to view other organizations as potential partners, not so much rivals. Management was conducted very informally and in an unstructured way. Informants in medium sized organization viewed other parts of the organization as partners. This is related to the fact that the amount of structure in the organization was larger in comparison. There were more levels of management and productivity metrics were measured. Size also increased the likelihood of people having a self image of themselves where their organizational role mattered more than in smaller businesses.

#### *B. Summary of Findings*

In the beginning we set out to answer research question *how is the organization or business type reflected in the emphasis of the business model elements in software firms?* We found answers to this question. The software business industry relies on human capital, which was also noted in this study. The human capital is the single most important key resource that enable the success of business. Our study fortifies the idea that startups work with more ad-hoc method [39] and the level of systematic working and bureaucracy increases when organization gain years and grow that also increases the cost structure. On the other hand for example Davis et al. [40] have modeled optimal organization structure amount and suggested that for established organizations in unpredictable environments, such as software business arguably tends to be, it is beneficial to decrease structure of the organization to gain flexibility, and for new organizations with little structure the need for building some structure is essential. According to our findings computer games are mostly being sold through existing channels and no new investments are needed. On the other hand B2B seems to require more specific channels and personal contacts. The brand building and IPR were present at the beginning when game organizations developed their products. The role of segmentation is also different between B2C and B2B type of business. Whereas B2B business is focused on certain industry domain or sector, the B2C business tries to gain as much customers as possible. To gain larger customer base it requires translations, localization and marketing, which increase initial costs. One interesting finding is the reliance of external funding of computer game startup. The external funding was not present in B2B startups but with computer

games external funding was a major player in the beginning. The Figure 2 summaries the findings in business model canvas and illustrates how the findings center around key activities and key resources.

This probably exists also in other domains than just software industry, but as software industry is building intangible products it also has different cost structure. For example, Saarikallio and Tyrväinen [13] have suggested a refined

<b>Key Partners</b> Startups identified other organizations as partners while established medium sized organization recognized other division as partners.	<b>Key Activities</b> B2B organization used <b>personal communication</b> to reach customers. Established ones <b>measured productivity</b> and had analytical way of building business while computer game organizations focused on <b>innovation and artistic way</b> of doing things and built <b>brand and IPR</b> from the beginning  <b>Key Resources</b> Established organizations identified people as <b>role-based</b> whereas in game organization people were <b>generalists</b> . In medium sized organization the level of management was increased while startups used only <b>coffee table management</b>	<b>Value Proposition</b> Game startups chose a <b>general industry value proposition</b> to provide entertaining experience and aim to maximize time spent with the product. B2B vendor aim to minimize time required by <b>automating processes</b> .	<b>Customer Relationship</b> B2B is often offering dedicated <b>personal assistance</b> . Games try to go for <b>self service</b> , community building, and collect anonymous user data.	<b>Customer Segment</b> While B2B <b>Telco</b> vendor considers segmentation as important, B2C game vendor sees the <b>segmentation as a cost</b> .
<b>Cost Structure</b> Established and medium sized organization had <b>larger organizational structure</b> , which produced costs.		<b>Revenue Streams</b> Most of the startups were trying to get <b>external funding</b> whereas B2B <b>Telco</b> organizations revenue was earned with <b>maintenance</b> .		

Figure

2. The findings presented in business model canvas. Two most special fields are highlighted with different color.

## V. DISCUSSION

### A. 5.1 BMC for software organizations – improvement ideas

Although the original Business Model Generation book [30] describe also software companies like Skype, we argue that utilizing BMC should include the idea that different elements have different weights in different industries. BMC could be the starting point, but it cannot be considered as a perfect tool for modeling software business. It could be speculated that BMC reflects better fields of business creating concrete products where for example concrete channels and logistics need to be built when delivering products to customers.

We also found out that there exists concepts that are hard to put under one element. The organizations in our study discussed that for example a venture capitalist can be identified as a revenue stream as it provides money. It can also be identified as a resource as it is used in a process to develop a product. Finally it can also be recognized as a partner when the relationship is close and in addition to money also other form of collaboration is done. Similarly some gamers in the customer segment are also part of the key resources when they spend time on giving feedback, improving ideas and even developing content to a game.

The BMC model [30] does not account for external funding as part of the revenue stream element. The findings in this study suggests that this is a very important element of business model that comes up in practice especially in the startup case.

model of revenue stream, where it is divided into three sub-elements which are the source, reason and method of revenue. Venture capitalists fit within this model as they are the source of the money stream. The reason is not a product or service, but a stake in the whole company due to belief in the success potential of the firm. The method is an equity or sometimes debt investment. This shows that in some cases a refined model is more applicable than the more general revenue stream construct and demonstrates the need for re-evaluating parts of the business model construct further.

All investigated cases considered people as the most important resource. This is most likely a common phenomenon in the information technology industry. As the software industry is manufacturing intangible products [33] the human resource component raises to be one of the most important elements in the business model regardless of the size or age of the organization or type or field of business. The same conclusion can be indirectly derived from the software engineering research, where it has been noted that the quality of people is the largest success factor [41]. Thus, because software development is an essential part of a business in IT industry, it can be inferred that the business model reflects this same phenomenon. Our empirical findings are in line with this conclusion.

Based on earlier research by Vanhala and Kasurinen [12] the human capital stands out as the most important element and for example in the case of computer games the channel and customer segment elements were not seen that important.



Because the current study also indicates human capital as a very important area, it could suggest that human capital could be promoted to a main element in business model instead of being sub-element of key resources when we are discussing the software industry.

We found out that customer references were important to both B2B and B2C, but in B2C organizations it wasn't possible to choose the references so easily for marketing purposes, because the app stores allow both positive and negative feedback. Thus, it could be argued that B2B references are easier to control, whereas B2C requires more quality assurance and marketing efforts.

The findings indicate that organizations involved in established B2B field measured worker productivity, but the startup B2C game organizations did not. It could be argued that the need to collect statistics on people's productivity is more natural in service business, because increasing productivity would translate directly to more profit and the extra capacity can be used to sell more to existing customers. Also as organization grows the processes improve and thus the measuring is introduced to organization work-flow. On the product side the link between profit and productivity of people is not as direct. Other things like quality of the product and marketing effectiveness can be said to have more impact than how productive people are. This would be an interesting avenue on which to conduct further research.

Our contribution to scientific community is the research of business model concept and business model canvas in software industry domain. We argue that the current research has decreased the ambiguity of the business model concept. There is still more work to do as for example startups are not discussed thoroughly. We also argue that BMC is a suitable tool to analyse business model, yet it has its own flaws especially with the human capital driven business manufacturing intangible products or services. This requires further research and maybe even some improvement to BMC framework.

#### *B. Managerial Implications*

This study has presented multiple differences in business model usage and understanding in the context of B2B vs. B2C, established vs. startup organization, as well as micro vs. medium-sized organization. It can give a practicing manager a good understanding of the new business model learnings she might be facing when focusing on new kinds of businesses. One example could be changing position from running a startup to leading an established business unit. Another example is doing the reverse when a manager wants to leave the corporate side and become an entrepreneur. When facing these issues we recommend that the manager thinks thoroughly how the ad-hoc working versus measuring productivity is handled. With right choices the motivation of workers can be significantly higher than when selecting incorrect methods to lead the organization. Managers should

also note how different customer relationship is between B2B and B2C business and how different organizations utilize different channels when reaching customers.

The environment change requires new kind of business model understanding and this paper gives insight on the differences and gives help in adapting management style for the new situation. The management tasks and ways vary because one correct way to do management and leadership was not identified.

#### *C. Limitations of The Study*

When discussing and analyzing qualitative data there are some threats to validity and generalization of the study. For example Robson (2002) classifies these threats: observational bias, researcher bias and reactivity. We had three different interviewers to avoid interviewer bias, two people to conduct the data analysis to avoid observational bias and this study has been discussed extensively with four people familiar with the topic and the data to avoid personal bias. As we combined two individual studies, it produces issues when the original aim of both studies has not been exactly the same. But as we utilized BMC in the both studies we argue that they are comparable and thus this study is valid in the sense of interview data. As this is a qualitative study it is only valid in this context and it should be considered as suggestions or practice-based recommendations beyond this scope.

This study has a couple of limitations. First of all it addresses only software business organizations based in Finland. All of them were targeting the global business, thus we can argue they present wider aspects of industry than just Finland. Still, we realize that broader view would provide improved results. In this sense we are merely opening discussion for the topic. Secondly in our study we are comparing B2B to B2C, startups to established organizations and micro/small organizations to medium sized one. We understand it diffuses the data widely, but we argue that the key findings are relevant and we merely miss some other issues rather than find non-existing ones. Thus, the presented results could be a subset of the results available through a sequel study with a larger sample.

## **VI. Conclusion and Future Work**

In this study we reported differences in business models in different information technology organizations. We noted several differences emerging when organizations are different aged or sized and their business type or field varies. Our study noted the importance of human capital as a key element of a business model and how people in older and larger organizations work role-based while in startups they are more generalists. Brand building and external funding are important among computer game startups whereas personal contacts to customers were seen important in our B2B cases. Also different management needed in different contextual

businesses. The software startup manages its daily life around coffee table and utilizes ad-hoc methods while established organizations had more formal processes and increased bureaucracy; key processes of software organization varies during the life cycle of the organization. This leads us to argue that the weight of business model elements vary between different software organizations so that especially managers need to understand the issue when switching from organization to another.

This study has also pointed out the difficulties of applying one concept of business model into varying organizations: we had difficulties to interpret where should be put venture capitalist as they were key partners while giving advices, revenue stream when providing money and they can also be categorized as key resource when they provide input to the development process. If we want the business model theory to become a generic conceptual tool that it has the potential to become, it is very important to consider the environment in which the concept is applied and notice the varying details arising from those environments. We have pointed out some areas where it is not unambiguous how to categorize the data into the traditional business model elements. This suggests there still is a need for further clarification and refinement of the business model concept.

This study concentrated only on software business related organizations. We would be interested in comparing our findings with findings from other fields of business.

### Acknowledgment

This study was partially funded by the European Union Regional Development Grant number A32139 "Game Cluster" administered by the Council of Päijät-Häme, Finland, and the organizations funding the related research project.

### References

- [1] J. Hedman and T. Kalling, "The business model concept: theoretical underpinnings and empirical illustrations," *Eur. J. Inf. Syst.*, vol. 12, no. 1, pp. 49–59, Mar. 2003.
- [2] E. Luoma, M. Rönkkö, and P. Tyrväinen, "Current Software-as-a-Service Business Models: Evidence from Finland," in *Software Business*, vol. 114, M. Cusumano, B. Iyer, and N. Venkatraman, Eds. Springer Berlin Heidelberg, 2012, pp. 181–194.
- [3] M. M. Al-Debei and D. Avison, "Developing a unified framework of the business model concept," *Eur. J. Inf. Syst.*, vol. 19, no. 3, pp. 359–376, Jun. 2010.
- [4] A. Ojala and P. Tyrväinen, "Business models and market entry mode choice of small software firms," *J. Int. Entrep.*, vol. 4, no. 2, pp. 69–81, 2006.
- [5] J. Hwang and C. M. Christensen, "Disruptive Innovation In Health Care Delivery: A Framework For Business-Model Innovation," *Health Aff. (Millwood)*, vol. 27, no. 5, pp. 1329–1335, Sep. 2008.
- [6] M. Morris, M. Schindehutte, and J. Allen, "The entrepreneur's business model: toward a unified perspective," *J. Bus. Res.*, vol. 58, no. 6, pp. 726–735, Jun. 2005.
- [7] M. Schief and P. Buxmann, "Business Models in the Software Industry," presented at the 2012 45th Hawaii International Conference on System Sciences, Hawaii, USA, 2012, pp. 3328–3337.
- [8] A. Valtakoski and M. Rönkkö, "Diversity of Business Models in Software Industry," in *Software Business*, vol. 51, Springer Berlin Heidelberg, 2010, pp. 1–12.
- [9] E. Vanhala and K. Smolander, "What Do We Know About Business Models In Software Companies? - Systematic Mapping Study," *IADIS Int. J. WWWInternet*, vol. 11, no. 3, pp. 89–102, 2013.
- [10] N. Paternoster, C. Giardino, M. Unterkalmsteiner, T. Gorschek, and P. Abrahamsson, "Software development in startup companies: A systematic mapping study," *Inf. Softw. Technol.*, vol. 56, no. 10, pp. 1200–1218, Oct. 2014.
- [11] S. M. Sutton, "The role of process in software start-up," *IEEE Softw.*, vol. 17, no. 4, pp. 33–39, Aug. 2000.
- [12] E. Vanhala and J. Kasurinen, "The role of business model and its elements in computer game start-ups," presented at the ICSOB 2014 - The 5th International Conference on Software Business, Paphos, Cyprus, 2014.
- [13] M. Saarikallio and P. Tyrväinen, "Following the Money: Revenue Stream Constituents in Case of Within-firm Variation," in *Software Business. Towards Continuous Value Delivery*, Springer, 2014, pp. 88–99.
- [14] C. Baden-Fuller and S. Haefliger, "Business Models and Technological Innovation," *Long Range Plann.*, vol. 46, no. 6, pp. 419–426, Dec. 2013.
- [15] B. Mahadevan, "Business models for internet-based e-commerce: an anatomy," *Harv. Bus. Rev.*, vol. 42, no. 4, pp. 55–69, 2000.
- [16] R. Casadesus-Masanell and J. E. Ricart, "From Strategy to Business Models and onto Tactics," *Long Range Plann.*, vol. 43, no. 2–3, pp. 195–215, Apr. 2010.
- [17] A. Osterwalder and Y. Pigneur, "An e-Business Model Ontology for Modeling e-Business," *EconWPA, Industrial Organization 0202004*, Feb. 2002.
- [18] D. J. Teece, "Business Models, Business Strategy and Innovation," *Long Range Plann.*, vol. 43, no. 2–3, pp. 172–194, Apr. 2010.
- [19] B. W. Wirtz, O. Schilke, and S. Ullrich, "Strategic Development of Business Models: Implications of the Web 2.0 for Creating Value on the Internet," *Long Range Plann.*, vol. 43, no. 2–3, pp. 272 – 290, 2010.
- [20] J. Magretta, "Why business models matter," *Harv. Bus. Rev.*, vol. 80, no. 5, pp. 86–92, 2002.

- [21] L.-M. Sainio and E. Marjakoski, "The logic of revenue logic? Strategic and operational levels of pricing in the context of software business," *Technovation*, vol. 29, no. 5, pp. 368–378, May 2009.
- [22] R. Amit and C. Zott, "Value creation in E-business," *Strateg. Manag. J.*, vol. 22, no. 6–7, pp. 493–520, Jun. 2001.
- [23] C. Zott, R. Amit, and L. Massa, "The Business Model: Recent Developments and Future Research," *J. Manag.*, vol. 37, no. 4, pp. 1019–1042, May 2011.
- [24] N. Weiner and A. Weisbecker, "A Business Model Framework for the Design and Evaluation of Business Models in the Internet of Services," in *Proceedings of the 2011 Annual SRII Global Conference*, 2011, pp. 21–33.
- [25] S. M. Shafer, H. J. Smith, and J. C. Linder, "The power of business models," *Bus. Horiz.*, vol. 48, no. 3, pp. 199–207, 2005.
- [26] P. Timmers, "Business Models for Electronic Markets," *Electron. Mark.*, vol. 8, no. 2, pp. 3–8, 1998.
- [27] R. Alt and H.-D. Zimmermann, "Introduction to Special Section - Business Models," *Electron. Mark. - Int. J.*, vol. 11, no. 1, 2001.
- [28] R. Rajala, M. Rossi, and V. K. Tuunainen, "A framework for analyzing software business models," in *ECIS*, 2003, pp. 1614–1627.
- [29] H. Chesbrough, "Business model innovation: it's not just about technology anymore," *Strategy Leadersh.*, vol. 35, no. 6, pp. 12–17, 2007.
- [30] A. Osterwalder and Y. Pigneur, *Business model generation - a handbook for visionaries, game changers, and challengers*. John Wiley & Sons, Inc., 2010.
- [31] A. Osterwalder, Y. Pigneur, and C. L. Tucci, "Clarifying Business Models: Origins, Present, and Future of the Concept," *Commun. Assoc. Inf. Syst. Vol.*, vol. 15, pp. 1–25, 2005.
- [32] A. Pateli and G. Giaglis, "A framework for understanding and analysing e-business models," presented at the Bled Electronic Commerce Conference, Bled, 2003.
- [33] H. Chesbrough and J. Spohrer, "A research manifesto for services science," *Commun. ACM*, vol. 49, no. 7, p. 35, Jul. 2006.
- [34] G. G. Gable, "Integrating case study and survey research methods: an example in information systems," *Eur. J. Inf. Syst.*, vol. 3, pp. 112–126, 1994.
- [35] C. B. Meyer, "A Case in Case Study Methodology," *Field Methods*, vol. 13, no. 4, pp. 329–352, Nov. 2001.
- [36] H. K. Klein and M. D. Myers, "A Set of Principles for Conducting and Evaluating Interpretive Field Studies in Information Systems," *MIS Q.*, vol. 23, no. 1, p. 67, Mar. 1999.
- [37] F. Kohlbacher, "The use of qualitative content analysis in case study research," in *Forum Qualitative Sozialforschung/Forum: Qualitative Social Research*, 2006, vol. 7.

### Author Biographies

**First Author** Erno Vanhala is a PhD student at the Lappeenranta University of Technology, Finland. He graduated as Master of Science in engineering and has since published international research articles on topics such as computer game business, innovation in game development and requirement engineering aspects. His current interest in the academic field include computer game start-ups and their business models. Besides business issues he is also mesmerized by the open source phenomenon and web-based software. He is also merited teacher, having received several awards on teaching different software engineering topics.

**Second Author** Matti Saarikallio has graduated as Masters of Science in engineering from Aalto University, Finland, and started his PhD in University of Jyväskylä, Finland. He is heavily involved in the software industry and has found for example a computer game company.



### III

## **METRICS FRAMEWORK FOR CYCLE-TIME REDUCTION IN SOFTWARE VALUE CREATION**

by

Pasi Tyrväinen, Matti Saarikallio, Timo Aho, Timo Lehtonen & Rauno  
Paukkeri, 2015

ICSEA 2015: The tenth international conference on software engineering  
advances

Reproduced with kind permission by IARIA.

# Metrics Framework for Cycle-Time Reduction in Software Value Creation

## Adapting Lean Startup for Established SaaS Feature Developers

Pasi Tyrväinen, Matti Saarikallio  
Agora Center, Department of CS and IS  
University of Jyväskylä, Finland  
pasi.tyrvainen@jyu.fi, matti.saarikallio@gmail.com

Timo Aho, Timo Lehtonen, Rauno Paukeri  
Solita plc  
Tampere, Finland  
{timo.aho, timo.lehtonen, rauno.paukeri}@solita.fi

**Abstract**— Agile software development methodologies driving cycle-time reduction have been shown to improve efficiency, enable shorter lead times and place a stronger focus on customer needs. They are also moving the process development focus from cost-reduction towards value creation. Optimizing software development based on lean and agile principles requires tools and metrics to optimize against. We need a new set of metrics that measure the process up to the point of customer use and feedback. With these we can drive cycle time reduction and improve value focus. Recently the lean startup methodology has been promoting a similar approach within the startup context. In this paper, we develop and validate a cycle-time-based metric framework in the context of the software feature development process and provide the basis for fast feedback from customers. We report results on applying three metrics from the framework to improve the cycle-time of the development of features for a SaaS service.

**Keywords**— metrics framework; cycle-time; agile; software engineering process; lean startup; feedback; SaaS.

### I. INTRODUCTION

The software engineering (SWE) process has traditionally been managed on a cost basis by measuring programmer effort spent per lines of code, function point or requirement. These metrics have also been used to guide software process improvement. In order to align more with business strategy and value production the focus has shifted more towards value creation instead of cost reduction. For example, value-based SWE [1], software value-map [2] and a special issue on return on investment (ROI) in IEEE Software [3] have explored value in software development. As a reaction to move away from a cost-reduction focus, the recent goal of lean thinking has been to optimize for perceived customer value [4]. Thus, we can say that leadership approach for the software development process is moving from a cost focus to a value focus.

Measuring the value of application software and cloud services is difficult to do before it is in use, as you need to consider the value of the software for the potential users, the business value for the firm developing it and the value for other stakeholders [1][5][6]. The current theories of value do not present a simple way of assessing customer value [7]. Although companies put a great amount of effort into increasing customers' perceived value in the product development process, determining how and when value is

added is still a challenge even in marketing and management sciences. [7] Further, the software engineering metrics are measuring attributes of the software development process (e.g., cost, effort, quality) while these metrics remain disconnected from the attributes and metrics developed for measuring value (see Table I). Various approaches have been developed to overcome this gap [1][5][6][8][9][10][11][12][13][14][15][16] without any major break-through.

The software engineering community has adopted an iterative approach to software development in form of Scrum [17], XP [18] and other agile [19] methods. These promote fast cycle user interaction and development process to keep the effort focused on customer needs based on fast customer feedback either interactively or through analysis of service use behavior. The startup community has adopted a similar approach and commonly uses the lean startup cycle [20] to evaluate the hypothesis of customer needs using the build-measure-learn cycle, which is repeated to improve customer acceptability of the offering and the business value of the startup. The common theme of these approaches is that instead of trying to estimate or predict the value in advance, try to shorten the cycle time from development to actual customer feedback, which indicates the value of the software in use. That is, from the SWE perspective, the speed of feedback received from users is the best indicator of the value of the newly created software. This indicates that shortening the feedback cycle would drive the SWE process towards faster reaction on customer value and higher value creation.

Although there exists a common understanding about the key role of a fast customer feedback cycle in linking the SWE process to value creation, the measurement methods and metrics available in literature are positioned either as cost-based SWE methods or as value-oriented metrics with little connection to the engineering process providing little guidance for managing and developing the SWE process (see Table I). Thus, the research question of this paper is, what metrics would guide cycle-time-driven software engineering process development in established organizations?

As the answer is context-dependent, a set of metrics will be needed. This paper aims at filling this gap by proposing a metrics framework enabling adoption of such metrics in a variety of contexts where new features are incrementally added to software.



TABLE I. POSITION OF THIS RESEARCH TO BRIDGE COST-ORIENTED SOFTWARE ENGINEERING (SWE) METRICS AND VALUE-ORIENTED BUSINESS METRICS

	Measurement Domains		
	<i>SWE Metrics</i>	<i>Research Gap Addressed Here</i>	<i>Value Metrics</i>
Scope (measurement target)	SWE Process	Value Creation Cycle	Customer Value of Offering, Value of Startup
Measured Attribute	Cost, Effort, Quality	Cycle Time	Value for Customer, Value for Enterprise
Examples	Function Points per month, Faults per lines of code		Value in Use, ROI, Lean Analytics

Applying the guidelines of the design science method [21], this research has been initiated based on company needs presented in interviews of Software as a Service (SaaS) development firms in a large industry-driven research program [22], to target an issues with business relevance in firms.

In Section II, we construct the metrics framework artifact based on the analysis and synthesis of previous research literature selected from the perspective of the research question. Following the design science research guidelines, we also demonstrate generalizability of the framework artifact to several contexts by choosing from a variety of metrics to target the specific process development needs. We also propose a simple diagrammatic representation for visualizing some of the metrics values in operational use to pinpoint development tracks requiring attention in an organization with multiple parallel feature-development teams.

In Section III, we evaluate the metrics framework by applying it to the case of a firm developing new features for an existing SaaS service and discuss the impact of the findings on revising the target of the next process improvement actions. In Section IV, we summarize the results, draw the conclusions and propose directions for further research.

## II. THE CYCLE-TIME METRICS FRAMEWORK

### A. Developing the Framework

The flow of new features through a SWE process can be measured at various points in time with an aim to reduce delay between points to reduce cycle time. The scope of the process measured will impact the attention of the software developing organization. In the narrowest scope, the cycle time measured includes the basic software development cycle while the widest cycle takes into account the customer needs and experience and, thus, matches and even expands the lean startup cycle [20].

In the proposed framework (see right side of Figure 1), the feature life-cycle begins with three planning phase events: 1) a need emerges, 2) a software development

organization recognizes the need, and 3) the decision is made to develop the feature. In large established organizations, the identification of feature needs has been excluded from the responsibility of the SWE organization to responsibility of the product marketing organization, while the entrepreneurship-oriented startup community has emphasized the value of including the need identification step as an inherent part in the fast business development cycle of the organization developing the software. Sometimes there is an intentional lag between events 2 and 3 as the decision may be to wait for the right time window (cf. real options [23][24]), or features with higher priority are consuming all resources available.

Continuing from the 3 events that form the beginning of the feature life cycle (above) and for the purposes of measuring the value creation cycle, the main development events included in this framework are 4) development starts, 5) development done, and 6) feature deployed. Use of XP, Scrum and other iterative and incremental development (IID) processes has aimed at reducing the time between events 4 and 5 (or fixing that to 2–4-week cycles). The cycle-time from 4 to 5 is here referred to as the Development cycle (see Figure 1). Moving from packaged software to cloud delivery and SaaS development along with moving from an annual or a six-month software release cycle to continuous integration (CI [25]) and continuous delivery (CD [26]) in development operations (devops [27]) has reduced the interval between 4 and 6.

After the event 6, the traditional software engineering process is often thought to be completed, while many entrepreneurship-oriented approaches, such as Lean Startup [20], go further, starting from building a product to measuring the use of it, which produces data used for learning and for producing ideas for the next development cycle (see left side of Figure 1). That is, building the product based on current ideas is only one of the three main events needed for value creation: build–measure–learn [20]. For considering the business and customer perspectives in this metrics framework for the value creation cycle, we need to expand beyond step 6 to include the use, measuring and learning phases: 7) when the feature gets used, 8) when feedback data is collected to support learning, and 9) when a decision is made based on the feedback. Note that events 8 and 9 resemble events 2 and 3 while not all information from customer needs is collected through measuring the use of the current product. It is also commonly assumed that the time from feature deployed (6) to first use (7) is short, while without measured data this can be an incorrect assumption. There have been cases where almost half of software features were never used [28]. Further, if software quality is high, it can take some time to get feedback, and it may require many uses of the feature before customer sends feedback about problems. Additionally, it can take time for a feature to get sufficient number of uses to allow for a reliable analysis of customer behavior (8). Also, the deployment process of the company can delay the decision to act on the feedback (9).

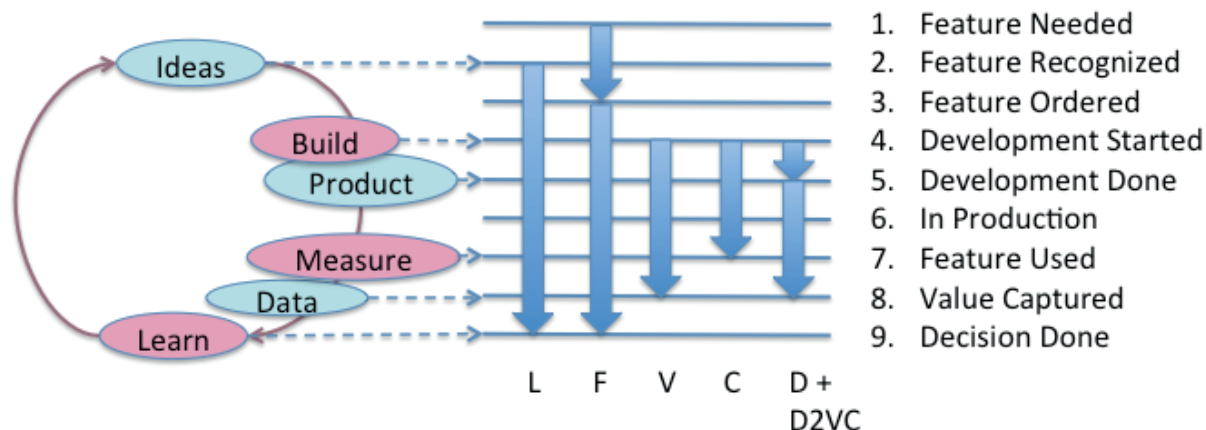


Figure 1. The value-driven metrics framework for driving software engineering cycle-time reduction (on the right), the Lean Startup cycle (on the left) and example cycles, for which cycle time can be used as the metrics driving cycle-time reduction (in the middle).

Figure 1 depicts the proposed framework. On the right side we have the sequence of events identified. On the left side, we have the Lean Startup cycle with horizontal arrows pointing from the phases to related events of the framework. The vertical arrows in the middle represent examples of cycle times that can be used as a target metric for developing SWE process. The cycles in the center are labeled as follows: L = Lean Startup cycle, F = Full cycle including fuzzy front end and full feature development cycle, V = Value cycle from starting the development to value capture, C = Core cycle from development start to first feature use, and finally D+D2VC, where D = Development cycle from start of development to production readiness and D2VC = time from development done to value captured. We emphasize that this list of cycles is not exclusive and new cycle time metrics can be created with this framework on demand for each context.

#### B. Changing Process Development Focus through Metrics

The various cycle-time metrics available in the framework can be used for focusing process development activity to specific process areas based on the need (see Table II). For example, if the basic software development process has been well developed and if some incremental development process, automated testing and continuous integration are applied, it may be useful to shift the attention to continuous deployment. In that case, the metric to be followed can be changed from Development cycle to cycle time between events 4 and 6, from start of development to start of production (see the second line in Table II). Changing the metric will also change the focus of attention and can often result in adjusting the processes, resource allocations or tools used.

TABLE II. EXAMPLE PROCESS DEVELOPMENT TARGETS WHEN USING ALTERNATIVE CYCLE-TIME METRICS

Cycle	Start Event	End Event	Addressed Capabilities	Process Development Focus
D, Development	4: Development Started	5: Development Done	XP, Scrum and other IID processes, automated testing and continuous integration (CI)	Using this cycle-time metrics addresses cycle-time of the basic SW development process
Time to production	4: Development Started	6: In Production	Same as in D, adding continuous deployment (CD) to the measurement scope	Using metrics for this cycle time focuses attention to CD capability
C, Core cycle	4: Development Started	7: Feature Used	Same as previous adding communication (diffusion) to customer base to the scope	Here the focus shifts to integrating customer facing team with development
V, Value cycle	4: Development Started	8: Value Captured	Adding customer analytics and customer feedback capabilities to the previous scope	Shifts focus to integrating analytics capability to IID+CI+CD capability
Time to Value	4: Development Started	*: Break Even	As Value cycle, but using this metrics assumes that value produced can be evaluated.	As in Value cycle
D2VC	5: Development Done	8: Value Captured	Post-development processes needed to deliver the created value and to get the feedback	Focusing on value cycle capabilities after the basic SW development process.
Fuzzy Front End	1: Feature Needed	3: Feature Ordered	Deep customer understanding (between events 1 and 2) and market understanding (2 to 3)	Measuring capability to find customer needs close to actionable market
...	...	...	...	...

In large organizations, where the product-marketing department is responsible for collecting market requirements and for product launches, the processes crossing product development and product-marketing departments may be problematic. In these cases, choosing the Value cycle, Time to Value or Design done to value captured (D2VC) as a common metric for both of the departments will enforce collaboration between the departments and will likely improve the total value creation capability of the organization, while local metrics within the departments are likely to lead to local optimization leading to non-optimal organizational behavior. It should be noted, that this issue appears mainly in large established organizations rather than in small startup firms, the needs of which the lean startup approach has been developed.

The time to value cycle in Table II ends with the event of reaching the breakeven point, which is marked with an asterisk “\*” rather than a number representing a specific ordering in the framework. In some cases a pay-per-use business model provides a basis to determine the break-even point for a feature, while in some cases the break-even point is estimated by qualitative means. A new feature may produce enough value to reach the break-even point when it is published (event 6) or when it is used for the first time (event 7). However, in many contexts this event occurs close to event 8, Value captured, that is, the feature use count is high enough, and sufficient feedback has been received, to ascertain whether the feature was worth the development effort. Based on these examples and the other examples in Table II we can observe, that the choice of applicable metrics is context dependent. Thus there is a need for a framework for metrics, which supports choosing the metric applicable for a specific situation.

### C. Depicting Cycle-Time Elements

Depicting the proposed cycle-time metrics makes it easier to decide whether to further develop or even to drop a existing feature and will also help in communicating the cycle-time reduction agenda to software engineers and other parties involved. For this purpose we devised a simple diagrammatic representation presented in Figure 2. In this example, the development starts at point 4 and ends at point 5. The y-coordinate represents the cumulative development time, in line with the cumulative cost for the organization. This linear curve is intentionally simplistic as the focus is on the form of the curve after event 5. In contrast, software engineering oriented representations, such as the Kanban Cumulative Flow Diagram [29], focus on analysis of the development cycle from 4 to 5 and ignore activity after production readiness.

From event 5 on, the horizontal line represents the duration of the waiting time from ready-to-deploy through deployment to first use. The feature is used for the first time in production at event 7. After that the dropping logarithmic curve represents the speed at which feedback has been received. After the second use the curve comes down to half, after the third use to one third of the original, and so on.

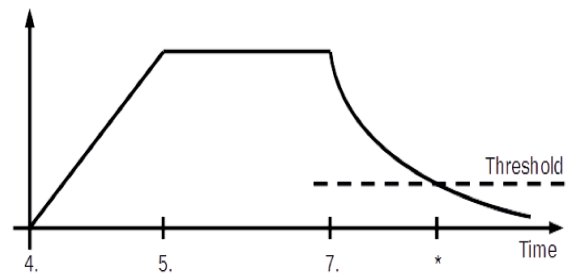


Figure 2. Depiction of the cycle times for feature analysis and process development. The numbers refer to the event number in the framework.

That is, the curve represents development time divided by number of times used. A context-specific target threshold for development time per times used is presented as a dotted line and the time when the curve reaches the threshold is marked with an asterisk “\*”.

In line with our approach to focus on the cycle times, this graphical representation aims at depicting the cumulative effort invested to the feature during development. There is a risk embedded in this development effort as it has not received feedback from the customers. Thus it is potential waste if customers do not accept the feature. This risk is mitigated along the narrowing gap of the asymptotic curve and the horizontal axis and reaching the threshold indicates that enough customer feedback has been received to ascertain whether it has been worth developing the feature. Event 8, Value captured, is serving this purpose as the event when sufficient user feedback is gained to evaluate the value of the newly developed feature and for adjusting the development plans accordingly, to further develop the feature or to drop it. In addition to guiding value creation, fast feedback from event 8 makes it easier for software developers to fix errors and modify the feature as long as they can still recall the implementation of the feature and have not moved on to new assignments.

Although measuring value is difficult, we would also like to identify the time-to-value cycle, that is the time from starting the development to break even, to the point at which its value to the customer exceeds the development costs. Now we face the challenge that while the cost can be easily measured in terms of time or money, value as a concept is not clearly defined and even if it were it would be hard to measure. We can speculate that the break-even point could be reached already on deployment (for features whose existence provides value even if they are not used, e.g., emergency-situation feature), on first use (when customer finds it), after a certain amount of uses (some use value derived from each), or sometimes a feature can fail to become profitable. Thus, the location of this measurement point cannot, in general, be identified in the sequence of events in the proposed framework, rather it is context dependent.

If we want to measure value, we need to define value. Historically three forms of economic value are the use value, exchange value and price [30]. There are many theoretical divisions of value to support decision making about which



software feature to work on next [2][5][7][8][9][10][11][12][13][14][15][16][24][30][31], but most theories consider the use value to the customer as essential. For the purposes of metrics development the focus will be on customer use value. It is important to note that due to market mechanism, exchange value is less or equal to use value [30]. This means that we could calculate a monetary estimate for the upper bound of the value captured by the software developer, that can be compared with cost. Still, the issue is problematic.

If we assume that there is use value for a feature, and in some cases the use value can be estimated as equal for each use, we would like to measure directly the cost versus benefits ratio:  $\frac{\text{development costs}}{\text{benefits}}$ . However, as discussed the benefits are challenging to measure and, at worst, we might need a new metric for each feature. This leads us to suggest that we isolate the hard-to-measure part, benefits, by instead measuring the precisely calculable cost per use  $\beta = \frac{\text{development costs}}{\text{times used}}$  and only if possible compare it to the estimated value for the user, based on a case-by-base estimation method. Next, we will show, using a case study, that reaching events “\*” and 8 produce very similar value for process development and feature decision making and that they can be used interchangeably. Thus, time to receive enough feedback is also a good, practical proxy for value produced.

### III. METRICS VALIDATION CASE STUDY

#### A. Target Organization and Service

We evaluated the metrics framework in a mid-sized Finnish software company, Solita Ltd. The case software development team develops a publicly available SaaS (lupapiste.fi) used by citizen applying for a construction permit related to real estate and other structures. This privately operated intermediary service provides a digital alternative to avoid the time-consuming paper-based process of dealing directly with the public authority. This service is used by employees of the licensing authority in the municipalities (about 100 users), the applicants (citizens and companies, about 100 per month), and architects and other consultants (1-2 per application). The software development process metrics were evaluated with the usage data collected from the process flow of five new features of this SaaS service deployed during the observation period, in mid-2014.

The service has a single page front-end that connects through a RESTful API to its back-end. Each call to the API is recorded on the production log files with a time stamp. We mined and analyzed the log entries together with the development data captured by the version control system. In this case, we chose features that introduced a new service to the API and were thus possible to trace automatically with a simple script that queried the monitoring system automatically. Some manual work was needed to find the features that introduced a new API, but automation of this work is also possible.

#### B. Results from Applying the Metrics to Sample Features

From the recorded event time stamps we calculated three metrics values for the case features. Development cycle (D) from start (4) to done (5) in working days. Lag to production from done (5) to deployed (6) in calendar days. And finally, D2VC, time from development done (5) to value captured (8). In this context the target company estimated that enough feedback data was collected for learning when the feature was used four times per each day spent on development, which gave the context-specific definition for the value capture event (8). Table III presents the data that is depicted in Figure 3. To enable comparison, all the features are shifted in the time axis to have event 4 (start of development) at day 0. In a daily use, an alternative depiction can show the timeline representing the history of all features to current point of time from which it is easy to identify development peaks and, more specifically, to notice the curves that remain high after the peak which indicates a demand for action. Either a feature has not been deployed and promoted well for the users or there is no user need for the feature.

#### C. Case Analysis and Discussion

From Table III we can see that for these five features the average of development effort needed to implement and test the features was about eight working days. When the development was done, on an average 12 calendar days was spent on waiting for deployment of the feature to the production environment. We can also observe that the features with lower priority (F1647 Unsubscribe and F1332 Note) have almost double the lag to production compared to the other features.

TABLE III. DESCRIPTION OF THE SAMPLE FEATURES, THEIR PRIORITY, DEVELOPMENT TIME (IN WORKING DAYS), LAG TO DEPLOYMENT (IN CALENDAR DAYS) AND DEVELOPMENT TO VALUE CAPTURED (D2VC; IN CALENDAR DAYS)

Feature id	Priority	Development (days)	Lag to deployment (days)	D2VC (days)	Description of the feature
F1332 Note	2	10	24	24	Authority user can add a textual note that other users cannot see.
F1498 Attachment	4	9	10	N/A	Applicant user can set the target of an uploaded attachment.
F1507 Validate	4	10	1	49	System validates the form prior the user sends the application.
F1537 Sign	4	7	11	15	Authority user can require an applicant to sign a verdict.
F1647 Unsubscribe	3	2	15	28	Authority user can unsubscribe emails.
<b>Average</b>		<b>8</b>	<b>12</b>	<b>29</b>	

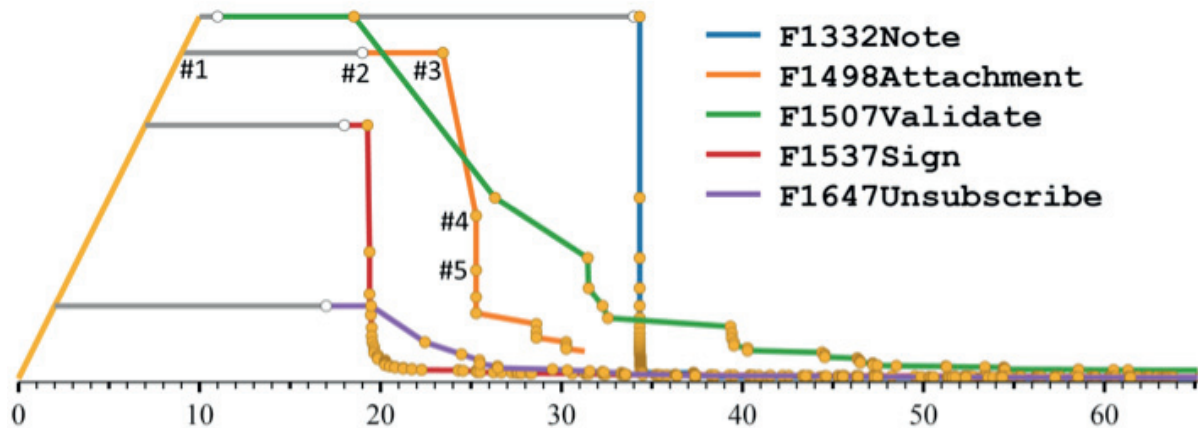


Figure 3. Depiction of the cycle-times of the five features. Development working days share the rising line starting from (event 5) and end in event 6 (start of the gray horizontal line), deployment (7, white dots in the right end of the gray part of the horizontal lines) and usage (yellow dots). To enable comparison, all the features are synced to have event 5 (start of development) at day 0.

The average time from completion of development to value capture is 29 days (this does not include feature F1498 Attachment, which did not reach the number of uses needed for the threshold). From the depiction in Figure 3, we can also see that this feature is no longer used. This feedback triggers the discussion on the reasons for the discontinuation of use of the feature to determine if there is a need to improve it or remove it from the service. When the target is to minimize the cycle times, minimizing the lag from production readiness to deployment (from event 5 to 6) and the means to increase the use of new features are clearly the places where major improvement can be reached much easier than from reducing average development time. By plotting the events in this way, it is easy to identify the places where changes can be made as well as to communicate the need with the development teams.

The results triggered also a discussion on the release practices of the firm. From the service use statistics it is possible to see that the service is heavily used from Monday to Thursday, less on Friday and very little during weekends. Thus it is likely that features released on Mondays will get used sooner than the ones released on Fridays, which provides the additional benefit that the feedback from users (8) would reach the developers when they still recall the software they were working on. Even more profound than the weekly cycle is a similar variation related to the vacation seasons. Deploying new features just prior to vacations will have negative impact on the Value cycle, as described above.

#### IV. SUMMARY, CONCLUSIONS AND FURTHER RESEARCH

The feedback from practitioners suggests that the current literature lacks metrics that could be used for directing a software development organization from the business perspective to enhance effective value creation and value capture. Although the Lean Startup Methodology proposes to develop the software via the build-measure-learn cycle, we

seem to lack the means to measure the value that the delivered software creates. Also the researchers have observed this problem and conclude [7], that the current theories of value do not present a simple way of assessing customer-perceived value. Although companies put a great amount of effort into increasing customers' perceived value in the product development process, determining how and when value is added is still a challenge even in marketing and management sciences [7]. Previous literature on XP, Scrum, lean startup and related approaches has indicated that in the context of SaaS services, delivering new versions of a service to the customer, collecting the usage data and making further decisions based on the data provides the most promising path for the software vendor to understand customer-perceived value. Agile methods have been shown to enable shorter lead times and a stronger focus on customer needs [32].

Shortening the cycle times provides increased flexibility maintaining options to change development direction with speed [20][22] as well as other business benefits for software service firms. This encouraged us to search for metrics that help software firms in the process development towards shorter cycles. On this basis, we formulated the research question as, what metrics would guide the cycle-time-driven software engineering process development in established organizations?

As the proposed solution, we adopted and extended the lean startup [20] value creation cycle and constructed a framework for metrics based on the times between main observable events within the cycle, all the way through to receiving and analyzing user feedback. This focuses attention on fast execution of the value creation within the user feedback cycle. That is, we are not trying to measure value of the results of the cycle, such as the value of the product produced or the value of the startup or progress of the startup in creating the offering, as in lean analytics [12].

By finding the measurable values from within the value creation cycle, the cycle-time metrics framework aims at bridging the gap between cost-oriented SWE metrics and value-oriented business metrics. Cycle-time reduction serves as the intermediary of increased value creation guiding software feature development and software process development. The metrics measure the calendar time between the key events. The first three events are related to feature need identification and the business decision to implement the specific feature (event 3). The core events following this decision are start of the development (4), the feature is ready for deployment (5), the feature is deployed (released, 6), and first use of the feature by a customer (7). These events are followed by feedback related events, the feature feedback data has been collected and analyzed (8) and a decision is made based on the feedback (9). The time intervals between the core events (4-7) are of most interest for the engineering while the other events (1-3 and 8-9) relate to the customer-perceived value analysis of the feature. We also provided examples on how changing the measurement cycle directs the process development to new process areas.

Our empirical focus was at the level of features being added to an existing SaaS offering. In the empirical part, the times between the events in the core cycle were measured for five new features in the development processes of an independent software vendor's SaaS service. The results showed that the core metrics were able to capture and bring up useful characteristics of the business process that triggered both a "drop vs. develop feature" discussion (for feature F1498) and a number of process development discussions. In these five feature development cases the average development time was shorter than the waiting time for the feature to be released. This has negative impact to the efficiency of fixing potential problems emerging during the first uses of the feature by first users, as the developers have already oriented towards another assignment. The detection of the delay of feature releases lead to a further analysis of the vendor's release practices in general and prompted quick improvements to their process.

Although the results from the empirical part showed that the metrics are useful in practice, there are still several avenues of further research that we wish to explore. The empirical part used data from the engineering system and customer feedback data to identify the core events. This seems to be a useful starting point and the firm in our case study would like to extend the collection of data to cover as many of the nine events as possible and as automatically as possible. The time from release readiness to analyzed customer feedback seems to be a particularly useful measurement of deployment performance.

In general, collecting the data can and should be automated using engineering information systems to the extent possible (events 1 and 8 cannot be detected automatically). For the other events, we propose collecting and depicting the data graphically in real-time status displays providing an overview of the development activities for business and engineering management. As we can observe from the empirical case, the results are useful both for

focusing process development activities and for making business decisions regarding which features will be developed further, which will be used as they are, and which features will be removed from the service. This way the simplified depiction can provide transparency between the business and the development organization. Thus we encourage further empirical work on the automation of data collection and its depiction based on events identified in the framework.

In startups the result of value creation cycle can be analyzed in the context of the evolution of the enterprise [12]. In context of established feature development processes, this framework adopted the approach of using only cycle times between events as the metrics within the value creation cycle. This is due to limited applicability of suitable previous research results for real-time customer-perceived value analysis beyond A/B testing and similar tools that can be used between events 7 and 8. Although cycle time metrics seems to provide high added value to focus process development in connecting software development with customer value, investigating the value capture events 8 and "\*" further is needed. Finding an easy to apply means for estimating the perceived user benefits would enable various new developments supporting the operative business development of a software engineering team.

#### ACKNOWLEDGMENT

This work was supported by TEKES as part of the Need for Speed (N4S) Program of DIGILE (Finnish Strategic Centre for Science, Technology and Innovation in the field of ICT and digital business).

#### REFERENCES

- [1] B. W. Boehm, "Value-based software engineering: Overview and agenda," in *Value-based software engineering*, Springer Berlin Heidelberg, 2006, pp. 3-14
- [2] M. Khurum, T. Gorschek, M. Wilson, "The software value map - an exhaustive collection of value aspects for the development of software intensive products," *Journal of software: evolution and process*, Wiley, 2012, 711-741.
- [3] H. Erdogmus, J. Favaro, W. Strigel, "Return on investment," *IEEE Software* 3(21), 2004, pp. 18-22.
- [4] K. Conboy, "Agility from first principles: reconstructing the concept of agility in information systems development," *Information Systems Research*, 20(3), 2009, pp. 329-354.
- [5] S. Barney, A. Aurum, C. Wohlin, "A product management challenge: Creating software product value through requirements selection," *Journal of Systems Architecture*, 54(6), 2008, pp. 576-593.
- [6] A. Fabijan, H. Holström Olsson, J. Bosh, "Customer Feedback and Data Collection Techniques in Software R&D: A Literature Review," in *Software Business*. Springer International Publishing, 2015, pp. 139-153.
- [7] J. Gordijn, and J.M. Akkermans, "Value-based requirements engineering: exploring innovative e-commerce ideas," *Requirements Engineering* 8(2), 2003, pp. 114-134.
- [8] M. Rönkkö, C. Frühwirth, S. Biffl, "Integrating Value and Utility Concepts into a Value Decomposition Model for

- Value-Based Software Engineering,” PROFES 2009, Springer-Verlag, LNBP 32, 2009, pp. 362–374.
- [9] R.B. Woodruff, and F.S. Gardial, Know your customer: New approaches to customer value and satisfaction. Cambridge, MA, Blackwell, 1996.
- [10] C. Grönroos, “Value-driven relational marketing: from products to resources and competencies,” *Journal of Marketing Management* 13(5), 1997, pp. 407–419.
- [11] T. Woodall, “Conceptualising ‘value for the customer’: An attributional, structural, and dispositional analysis,” *Academy of Marketing Science Review*, no. 12, 2003, pp. 1526–1749.
- [12] A. Croll, and B. Yoskovitz, *Lean Analytics: Use Data to Build a Better Startup Faste.*, O’Reilly Media, Inc. 2013.
- [13] P. Tyrväinen, and J. Selin, “How to sell SaaS: a model for main factors of marketing and selling software-as-a-service,” in: *Software Business*, Springer, Berlin Heidelberg, 2011, pp. 2-16.
- [14] V. Mandić, V. Basili, L. Harjumaa, M. Oivo, J. Markkula, “Utilizing GQM+ Strategies for business value analysis: An approach for evaluating business goals,” *The 2010 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*, ACM, 2010.
- [15] M. Saarikallio, and P. Tyrväinen, “Following the Money: Revenue Stream Constituents in Case of Within-firm Variation,” in: *Software Business*. Springer International Publishing, 2014, pp. 88-99.
- [16] J. Bosch, “Building products as innovation experiment systems,” in: *Software Business*, Springer, Berlin Heidelberg, 2012, pp. 27-39.
- [17] K. Schwaber, and M. Beedle, *Agile Software Development with SCRUM*, Prentice Hall, 2002.
- [18] K. Beck, *Extreme Programming Explained: Embrace Change*. Addison-Wesley, 1999.
- [19] A. Cockburn, *Agile Software Development*, 1st edition, 256 p. Addison-Wesley Professional, December 2001.
- [20] E. Ries, *The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses*. Crown Publishing Group, 2011.
- [21] A.R. Hevner, S.T. March, J. Park, S. Rami, “Design Science in Information Systems Research,” *MIS Quarterly*, Vol. 28, No. 1, 2004, pp. 75-105.
- [22] J. Järvinen, T. Huomo, T. Mikkonen, P. Tyrväinen, “From Agile Software Development to Mercury Business,” in: *Software Business. Towards Continuous Value Delivery*, Springer Berlin Heidelberg, LNIB, vol. 182, 2014, pp 58-71.
- [23] H. Erdogmus, and J. Favaro, “Keep your options open: Extreme programming and the economics of flexibility,” in Giancarlo Succi, James Donovan Wells and Laurie Williams, “Extreme Programming Perspectives”, Addison Wesley, 2002.
- [24] M. Brydon, “Evaluating strategic options using decision-theoretic planning,” *Information Technology and Management* 7, 2006, pp. 35–49.
- [25] M. Fowler, *Continuous Integration*, 2006. <http://martinfowler.com/articles/continuousIntegration.html> retrieved: Septmeber, 2015.
- [26] J. Humble, and D. Farley, *Continuous delivery: reliable software releases through build, test, and deployment automation*, Pearson Education, Jul 27, 2010.
- [27] P. Debois, “Devops: A software revolution in the making,” *Cutter IT Journal*, vol. 24, no. 8, August, 2011.
- [28] J. Johanson, Standish Group Study, presentation at XP2002.
- [29] K. Petersen, and C. Wohlin. "Measuring the flow in lean software development." *Software: Practice and experience*, vol. 41, no. 9, 2011, pp. 975-996.
- [30] J.S.Mill, *Principles of political economy*, 1848, abr. ed., J.L.Laughlin, 1885.
- [31] M. Cohn, *Agile estimating and planning*. Pearson Education Inc. 2006.
- [32] M. Poppendieck and M.A. Cusumano, “Lean software development: A tutorial,” *Software*, IEEE, vol. 29, no. 5, 2012, pp. 26–32.



## IV

# **QUALITY CULTURE BOOSTS AGILE TRANSFORMATION - ACTION RESEARCH IN CONTEXT OF B2B SOFTWARE BUSINESS**

by

Matti Saarikallio & Pasi Tyrväinen, 2022

In review for Journal of Software: Evolution and Process

Reproduced with kind permission by JSME.

# Quality culture boosts agile transformation – action research in a B2B software business

## Authors

Matti Saarikallio

[matti@saarikallio.net](mailto:matti@saarikallio.net)

University of Jyväskylä

Pasi Tyrväinen

[pasi.tyrvainen@jyu.fi](mailto:pasi.tyrvainen@jyu.fi)

University of Jyväskylä

## Abstract

Agile methodologies are sometimes adopted in existing software businesses, with the assumption that benefits can be attained by only using a set of best practices, which can sometimes work to a degree. In this paper, a case is discussed where a software producing organization of seven teams achieved significant improvements. The goal of the research was to answer two questions: how an already agile organization could improve its performance further and what is the impact of promoting quality aspects. The questions were answered by implementing interventions based on prior literature and data emerging from semi-structured interviews. The context was an established business with a complex revenue stream structure, meaning the mix of various project/service/product based work rendered the adoption of agile methods a challenge. Action research comprising three rounds of interventions was conducted to improve the organization and its quality culture while enforcing code review practices. Interventions resulted in a significant

improvement in quality, as measured by reported defects. Therefore, it is suggested that agile methods are not sufficient on their own to take software business forward unless a quality-focused culture is simultaneously achieved through a mindset change and/or organizational structures to enforce quality practices. The paper contributes to research on the managerial practices of software business and agile transformation by providing empirical support to introducing formal quality improvement to the agile mix as a method for practitioners to improve organizations with complex business models and multiple teams.

### **KEYWORDS**

quality, mixed business model, agile adoption, empirical, scaled agile, increment planning event, hybrid development methods, team coordination, B2B, revenue stream

### **Highlights**

1. Agile transformations tend to ignore accompanying implementation of software quality practices.
2. Significant software defect reduction is possible with small cultural and procedural changes.
3. Existing agile organizations with complex business models can enhance performance through quality improvements.

## **INTRODUCTION AND BACKGROUND**

### **Introduction**

Research on agile transformation often involves analyzing the context of a simple business model and a small team, with the assumption that quality improvements will appear. In this paper, this approach is problematized in three ways. The first is related to difficulties faced by multi-team organizations in fully obtaining the expected benefits<sup>1</sup>. The second is companies with complex business models attempting to use these tactics, which originate from simpler contexts<sup>2</sup>. The third (and most interesting) is the suspicion that the quality of software produced after the introduction of agile practices does not necessarily improve as much as anticipated. The goal of this paper is to investigate these three themes and report on action research that resulted in measurable improvements.

In this introductory section, relevant background research is summarized to analyze the context, and a question about current agile approaches is raised. This forms the theoretical background, rationale, and aim for the selected action research interventions.

### **Software business and business model**

The context of the current paper is software business. One way to understand this context more clearly is through the lens of business models. Business models facilitate gaining a structured understanding of how firms conduct business and create value<sup>3</sup>. Established B2B (business-to-business) firms differ in their business models from recently established or B2C (business-to-consumer) firms. For example, established B2B firms tend to have personal communications with customers, have ways of measuring worker productivity, organize people in a role-based structure, focus mostly on process automation, have a larger organizational structure, and have more complex

3



revenue capturing structures (such as through service contracts)<sup>4</sup>. The business model of an established B2B software business can be further investigated starting from its revenue stream structure<sup>8</sup>. This means that the source of revenue streams, reasons for the existence of those streams, and the method of converting value to revenue in each stream should be considered. This type of analysis will be employed to determine the details of the contextual landscape of the case firm in this paper. Further, software businesses vary in terms of customer expectations or value drivers. Ulaga<sup>5</sup> lists these as product quality, service support, delivery, supplier know-how, time-to-market, personal interaction, price, and process costs. Value drivers are also influenced by business logic<sup>6</sup>. Therefore, in the context investigated in the current paper, it is important to examine specifically what B2B customers value. This paper focuses on an organization that operates within an established B2B market segment. Therefore, the value drivers empirically determined by Parry et al.<sup>7</sup> are chosen as a frame, and it is assumed that customers value the following as the most important factors: 1) software quality, 2) professionalism and 3) understanding the customer.

The history of agile methods originates from many sources. For example, scrum was influenced by the thinking of Takeuchi and Nonaka on how to speed up product development<sup>9,10</sup>. Due to this history, in traditional agile methodology practice, example businesses are suspected to have an over-representation of a single team producing work related to a clearly defined single product line for either a singular business customer or consumer market. While optimal for simplifying an ideal model, this can result in ignoring some aspects of the real world, which consists of a multiplicity of different types of business models, motivating further research into other types of software organizations.

### **Agile methods and quality practices**

Since the start of the current millennium, agile development methods have become the de facto tactics for software producing organizations. Agile methods are typically described as incremental,

co-operative, straightforward, and adaptive<sup>11</sup>. Core to these methods are agile manifesto principles that highlight the importance of interactions, working software, customer collaboration, and responding to change<sup>12</sup>. While quality is one of the key value drivers of software industry, as previously mentioned, the role of quality is rarely explicated in agile transformations. Quality may be implicitly expected to result from agile methods, for instance from shorter iterations, the practices of pair programming or developing the personal craftsmanship of the developers, but there is no explicit focus on developing quality practices.

When the size of an agile organization increases, issues related to the scaling of agility become relevant. While there is no single definition of what constitutes a large-scale team, typical suggestions are six teams or more<sup>13, 14</sup>, rendering the current research (with seven teams) a borderline case. Napoleão et. al.<sup>15</sup> analyzed the knowledge sharing activities in agile development, noting that sharing information inside the team is the knowledge management step that is strongly present in agile organizations. However, this would require team stability and does not consider cross-team concerns.

It has also been stated that agile methods are bespoke in origin<sup>16, 17</sup>, meaning that the focus is on small projects. Helander and Ulkuniemi<sup>6</sup> indicated that customer perceived value differs in the sense that customers of project firms tend to value deep customer relationships, while those in product businesses place greater value on assurances of quality. As there is an increasing interest in software-as-a-service, it should be noted that investment in software quality is often necessary in such business models<sup>18</sup>.

Sfetsos and Stamelos<sup>19</sup> conducted a systematic review of quality, specifically in agile practices. They highlighted test-driven development and pair programming as agile practices that contribute to

code quality. More traditional quality practices were not mentioned. For example, research on the impact of inspecting software (currently called code reviews) existed long before agile practices emerged. Software inspection and reviews have a long research history<sup>20</sup>. This stream appears to live its own life, and a recent review of modern code review practices<sup>21</sup> only mentions agile once, which is perceived as the reason why modern code review practices have replaced traditional formal reviews.

The importance of quality is undisputed if the goal is to produce professional software. Moreover, it has been suggested that the amount of reworking required for software development is as high as 44% of the total work effort, with the amount of defect insertion being highest during the early phases<sup>22</sup>. It is also quite easy to infer that defects found early in the process are easier to fix than those found later. Further, there is evidence that the more code is reviewed, the higher the quality of software<sup>23</sup>. While there is a common consensus in the software industry that code review practices are important, the actual implementation of such practices in organizations reveals problems. According to recent survey<sup>24</sup> 76% of people conducting code reviews do so less than once a week.

In other industries, the quality of production processes is managed by controlling and measuring aspects of safety, quality, delivery, inventory levels, and production<sup>25</sup>. For example, the lean and Six Sigma methodologies (and the so-called seven basic tools of quality) have been considered essential tools in many industries<sup>26</sup>. Adopting lean practices in software development has also been perceived as useful<sup>27</sup>, attempts have been made to adapt tools such as quality function deployment to the software context<sup>28</sup>. While agile practices have adapted ideas from these fields, some aspects may have been overlooked. Moreover, it would appear that current literature on agile practices does not consider inspection and reviewing as essential ingredients for creating quality software.

Another simple method for organizations to improve quality is through the use of checklists. These are used in various fields, from flying bomber aircraft to medical surgery<sup>29,30</sup>. Agile methods promote the use of coding standards written down in form of definition of done. However, it is quite rare to find an actual checklist in software teams. Moreover, while the idea of teams self-organizing towards better coding standards is good, this might be a little idealistic.

As discussed previously, while contemporary agile research somewhat neglects a focus on quality, the question is whether this is true in industry practice. Based on a consulting agency report<sup>31</sup> on how agile adoptions typically fail in industry, it is striking that the only mentions about quality relate to test automation, automated static analysis as part of continuous integration, or valuing craftsmanship. Accordingly, it is suspected that quality in the context of industry agile transformations sometimes tends to be reduced to suggesting the use of automation tools. This could lead to poor managerial decisions and the omission of best practices from general software research when transforming to agile practices.

### **Rationale and aim**

Based on the literature, the lack of focus on quality in agile transformations seems to require investigation. The case under study in this paper was identified as representing a software business with quite a complex business model, which could be affected by this. The business had adopted agile practices approximately 7 years before the current research and was still struggling to gain hoped benefits. Thus, the aim of the current study is to contribute to the research domain of software business quality improvement practices in the context of agile methodology, especially in the context of a specific less-researched type of software business—the established B2B mixed-business model. There seems to be a reduced interest in agile transformations pertaining to quality improvement aspects. It is suspected that in some cases it is assumed that agile transformations somehow fix quality automatically. Therefore, the goal is to investigate whether agile methods are

sufficient and if improvement opportunities exist through the introduction of additional quality checks.

Based on the literature and the highlighted question about quality focus in agile adoption in current industry practice, two research questions were considered important: 1) how can an already agile organization improve its performance further and 2) what is the impact of promoting quality aspects.

## **METHODOLOGY**

### **Case study and action research**

A case study strategy focuses on understanding the dynamics present in a single setting<sup>33</sup>. Although multiple teams are present in this study, they all share a similar context, rendering this more of a single setting. The level of analysis is not on a single team; it is on the organizational behavior and effectiveness of the whole business unit.

Information systems science uses both case studies and action research<sup>32</sup>. The research reported in the current paper is not purely observational, because a process improvement was actively initiated, guided, and followed-up by the author. Therefore, it is better described as a combination of a case study and action research.

Software development is an undertaking that resides in the complex domain. Accordingly, software-producing organizations tend to be complex and variable. As a methodology, the aim of action research is to understand a social situation and its change processes. The goal of action research is to analyze a complex social-organizational model (or complex human process) in a real environment<sup>34</sup>. Participation of the researcher in planning the changes and intervening in the social system to

improve the organization's functioning was considered a suitable method to gain a deeper understanding of this case. The goal was to gain knowledge and explore the applicability of interventions aimed at improving the organization's effectiveness.

Action research has the goal of actually changing (rather than simply describing and explaining) the organization<sup>35</sup>. Following this methodology, the initial recommendations are founded upon a theoretical frame. The recommendations lead to actions that produce experimental changes in the organization, and the actions and their impacts are reported and discussed. Lewin<sup>36</sup> formulated the cycle of action research as follows: early mapping of the situation, planning for changes, taking action, and making observations. Further, contextual analysis of behavior in a social setting is also emphasized. In his taxonomy of research methods, Järvinen<sup>37</sup> recently recommended action research if the goal was to achieve utility and build a better system. This renders it a relevant approach for this study, because the goal is to improve the output of a software-producing organization.

The case organization is described in the next chapter. Thereafter, the mixed method data collection techniques are described as integral parts of the conducted action research process. Mixed methods were employed because there is a need to understand the context and meaning in a social setting. In addition, there is a need to provide quantitative evidence about the impact of any changes, making the study easier to replicate and possibly generalize in the future<sup>38,39</sup>.

## **Case**

The focus of investigation is the business unit of a publicly traded Northern European software company. The investigated unit provides services and products for the financial services industry. Although the mother corporation has over 1,000 employees, the focus of this analysis is the unit serving the financial sector (approximately 100 employees).

Revenue sources of the chosen business unit emanate from its operations in the financial services software market, providing products and services for banking and loans, wealth management, fund management, asset management, and custody and settlement domains. It is noteworthy that the financial sector was one of the first industries to adopt computers on a large scale to optimize business processes, with up to 50% of banks' fixed capital already spent on computers 40 years ago<sup>40</sup>. Therefore, it is unsurprising that a large amount of source code and fairly comprehensive product offerings can be found in established players serving this sector.

The method of revenue capture varied substantially. There were maintenance fees from existing installations, license fees, hourly-based fees on customer-specific small features, and larger developments delivered as various forms of projects. Thus, there are service contracts, product license contracts, hourly-based work, and SaaS models used in parallel.

The reasons why revenue can be captured are from activating product features or creating new functionalities for customers, promising maintenance with agreed service levels. Some customers are offered specific people as capacity, while others are sold work through (sometimes capped) hourly contracts. It is also fairly common that a product feature required by multiple customers has a type of kick-back discount if other customers choose to use it, instead of providing exclusivity of features. Typically, a monthly service fee is increased by a certain percentage when more features are added to that customer's service. While customers vary substantially in terms of their product configurations and customer-specific tailoring, it has been possible to maintain a common source code core and database model (to a degree), making products more maintainable.

Based on a short revenue stream analysis, the revenue stream structure of the company is quite complex. Moreover, it is easy to understand that the complexity of operations is high in our case,

which has caused large variations across the operative teams. Hence, the case business unit is a typical small B2B multi-offering organization with a complex mix of services and products.

The business unit has offices in two countries and three cities, meaning it has a more national (rather than global) focus specializing in a deep understanding of a country's specific needs. The focal business unit's employee headcount is approximately 100, of which approximately 50 work in software development teams. Additional people work in service, product, and project management functions, as well as for sales and consulting. The operational model was based on separating the functions of sales, service management, project management, and other administrative roles from the software delivery organization.

### **Procedure and data collection**

Action research followed the phases of diagnosing, action planning, action taking, evaluating, and specifying learning. The action research procedure was cyclical (Figure 1), and three interlinked interventions were performed. During each intervention, the organizational situation was diagnosed by collecting data in the form of interviews, supporting documents, and observations (1). After diagnosis, a plan was formulated based on theoretical predictions emerging from data and supporting literature (2). During action taking (3), further observations were made and informal interviews were conducted to understand the impact. Evaluation of the outcomes of actions (4) was conducted based on qualitative and quantitative data, and then reflected. Finally, the learnings were specified (5).

PLACE FIGURE 1 ABOUT HERE

There were several data collection steps during the time window of this study, in accordance with the steps of action research. Data collection started at the end of Q1 2018 with interviews in the organization. The first goal was to quickly identify the most important problems (close timing also helped with avoiding cross-contamination of the interviews). Several changes were then introduced



in the three action research iterations. Most of the initial data was relevant during all interventions, as the interventions targeted different aspects of the diagnosed issues. It should be noted that different from classical action research, the interventions were not fully sequential. Instead, they had an overlap in time with respect to the first and second interventions, because it was possible to drive some actions in parallel. Decisions needed for the actions (and learnings) were discussed in the business unit's management team.

### **Diagnosing**

As a preliminary diagnosis of the situation, 41 people (Table 1) in the organization were interviewed to gather insights into the current state of affairs in the business unit. The questions pertained to the biggest challenges faced by the organization, the reasons for these challenges, improvement possibilities, required actions to realize the improvements, and requesting advice on the most important focus areas (adapted from Watkins<sup>41</sup>). The collected interview data was analyzed by conducting inductive content analysis<sup>42</sup>. The interview notes were summarized and translated into English and then coded using open and 'in vivo' coding<sup>43</sup>. The list of similar text items included 354 lines that were grouped based on similarity to create categories. The number of occurrences of similar comments per category were then counted to elicit a sense of importance for each one. These categories were further organized into main categories, allowing the most important themes that emerged from the data to be identified and analyzed. Some emotionally charged or sensitive details were removed from the reported table due to ethical and confidentiality considerations, although the data items were still counted in the analysis. Additionally, some of the organizational rituals, e.g. scrum events, were observed. Further, secondary materials (such as documentation, and observation of meetings) were examined.

PLACE TABLE 1 ABOUT HERE

### **Actions planned and taken**

Action planning and action taking were implemented after the initial diagnosis, which were also based on observed early improvements. The first of the three interventions was initiated shortly after the interviews. Several changes were planned and implemented during the research time-span. The plans of action were based on the diagnosis of the situation and theoretical predictions. It was essential to gain enough organizational and management support to drive the changes through to ensure they would stick. Therefore, not all changes were attempted at once. This small-steps approach also helped in building the organization's change capability. Building general change capability was important to allow appropriate pacing and sequencing to give time for adaptation to the new situation<sup>44</sup>. |

### **Evaluating**

Evaluation of the intervention outcomes was initially based on qualitative data such as observations and interviews. In addition, secondary documentation (such as management team meeting memos, architect meeting memos, and retrospective notes) was cross-checked.

While qualitative measures are good for in-depth understanding, a quantitative measure for the organization's ability to produce quality software was chosen to increase the reliability of the results. A good measure for this was the number of defects reported. Thus, evaluating the impact gained by actions taken was mainly based on measuring the number of defects reported during each quarter (3 calendar months). Data for the number of defects were obtained from the issue tracking system, which was the same system used for new functionality, defects, and tasks. The whole business unit was chosen as the unit of analysis; hence, the sum of all reported defects was obtained for each time interval. During the evaluation period, the effective head count remained almost flat (standard deviation of 2.05) with no trend. Accordingly, the data were not adjusted according to head count.

There were two defect count measures available from the issue tracking system: defects opened as internal (originating from anyone) and defects originating from customer's acceptance testing or other user reports (originating from customers). The conclusions were drawn from the customer-reported defects, because this measures the quality of the produced output more clearly. Similar measures have been adopted in previous research on product quality<sup>45</sup>. It should be noted that a defect here only included those issues that were suspected to require code changes. Typically, this was non-invoiceable work, meaning the practical relevance was high for the business. The measure did not include general support work (such as customer guidance or training).

Quantitative data were analyzed to ensure that observed changes in quality were relevant. This was evaluated by calculating the likelihood that results of changes could be explained by chance. This calculation was based on the assumption that the defect count followed a Poisson distribution, which is typical for integer count data. Null hypothesis: Changes in quarterly defects can be explained with random variation, meaning the process output quality did not improve and the before and after data points emanated from the same underlying distribution. Alternative hypothesis: quarterly defects reduced, meaning the process output has improved and before and after data points emanated from different distributions.

### **Specifying learnings**

As the interventions had the predicted impact, they were further solidified by making them enforced practices and the culturally expected ways of working. This was mainly accomplished by promoting good initial results in team presentations and explaining how this would constitute expected behavior in the future. The learnings were specified, communicated to participants, and maintained by updating documents. The common ways of working were also written down as living documents in the business unit's wiki.

## Results

### Initial results leading to interventions

The short interpretative literature that was summarized earlier formed the main theoretical assumptions and motivation for the action research interventions and was complemented by themes emerging from the data. In this section, the qualitative interview data common for all interventions are listed, then each intervention is described sequentially. The quantitative results are shown in the evaluation of Intervention 2, although they are also relevant for the diagnosis and evaluation of Intervention 3. Next, the categories and sub-categories emerging from the interviews are briefly listed. The grouping of the 354 coded sentences in the text data resulted in 7 main categories and 31 sub-categories (Table 2).

PLACE TABLE 2 ABOUT HERE

The first (1) main category was **cultural issues & instability** (103 comments were categorized here), which included the sub-categories of hurry, turnover, negative cultural issues, instability, team spirit, resourcing, interruptions, role clarity, and positive cultural issues. The second (2) main category was labeled **vision** (79 comments), which included the sub-categories of product vision/management, lack of leadership, context, and pricing/selling. The third (3) main category included comments relating to **process and quality** (67 comments ) and sub-categories relating to process, testing, documentation, code reviews, scrum issues, refactoring, maintenance, and definition of done. The fourth (4) main category related to **product and quality** (40 comments), including sub-categories relating to technical complexity, continuous integration, technical debt, and product quality. The fifth (5) main level theme was related to **project management** (28 comments), including sub-categories of administrative project management, bureaucracy, and project vs. product. The sixth (6) main category was about **knowledge and customer** (35 comments) related

comments, while the seventh (7) was related to other comments (such as personal advice given to the researcher about what to focus on and how to drive change).

After categorization, the issues were discussed within the organization and business unit's management team. Then, the highest priority improvement possibilities were selected based on business needs and the theoretical likelihood of success. It was decided that the most important were cultural issues and stability (including communication issues about vision) and quality issues (process and product). The other issues were not forgotten, although they were not the focus of improvements. A short summary of the interventions and motivations leading to them is provided in Table 3. Next, each step of the ensuing interventions is described in more detail.

PLACE TABLE 3 ABOUT HERE

### **Diagnosis (Intervention 1)**

The interviews revealed that people had problems in understanding the context of their work. There was also a lack of product vision; it seemed that separation of teams prevented communication across boundaries, and there was a lack of communication across projects and teams. There was also a feeling of people "jumping around" and a lack of team stability. Apparently, some people had left the company due to organizational problems, communication issues, and rushed projects, creating a sense of hurry.

Reflecting on previous research literature, it was theorized that one of the reasons for the problems was that the scaling of agile methods was failing, as the size was 7 teams. Additionally, it was suspected that the identified problem of moving people from team to team prevented effective team learning.

### **Action plan (Intervention 1)**

Action planning was conducted in collaboration with the organization. Partially based on the theories of agile scaling in the literature, it was predicted that improving communication, adding more visual workflow management, and increasing team stability would have positive impacts on the identified issues. The action plans were discussed within the organization's management team before taking action to ascertain support.

### **Taking action (Intervention 1)**

The first change was the introduction of a quarterly planning event that was adapted from the Scaled Agile Framework. This is suggested as a good choice when teams are struggling with information sharing and synchronization across teams<sup>46</sup>. The team structure was not changed for the first 3-month increment. Starting from the second increment, the team structure was changed toward business domain competence -based teams. Further, visual management was improved by unifying the visualization of issue workflows (kanban-board), which were shown in daily scrum meetings. This constituted fine-tuning, since varying related practices were already in place in most teams.

### **Evaluation (Intervention 1)**

After the first intervention, the turnover problems reduced and remained less than the corporate average for the length of the study. The interviews revealed that the general levels of satisfaction increased and team stability remained quite good, which allowed the teams to bond more effectively. In addition, complaints about not understanding the context were reduced.

### **Learnings (Intervention 1)**

The first intervention's results were as predicted, although they happened much slower than anticipated. The main learning was that in the context of this type of mixed business model in a borderline scaled agile organization, it is important to keep the teams fairly stable and ensure that

there is a visualization of the workflow. Further, it is important to provide a mechanism for sharing information between teams via additional processes, such as increment planning events.

### **Diagnosis (Intervention 2)**

The diagnosis of the second intervention was based on the finding that both process and product quality were considered problems, with the theory suggesting that quality is not necessarily the main concern in scrum. In fact, an informant mentioned agile method as a reason (or excuse) for not focusing on quality. While interview data is subjective, this resulted in examining this issue further, revealing that quality was not afforded much importance. It was discussed further that there was a tendency to think quality was the concern of an individual developer's skill rather than something that could be improved at an organizational level. Interview data also included the comment "there is no definition of done". However, checking this fact revealed that there was one, although it was hard to find and was followed quite randomly. Moreover, the process did not contain a point where this could be enforced by the team. In the interviews, there were only a few discussions on code reviews, with most informants commenting that code reviews were not conducted. However, after checking, it was revealed that some people conducted these occasionally and that there was already a tool for this purpose.

The contract structure (ref. revenue stream analysis in case description) can worsen the situation with quality problems, as the fixes are often free. Unfortunately, there was no easy quantification of this issue due to the lack of consistent project reporting data; hence, the evidence is only based on the interviews. It was speculated that a long history of using a more traditional project-based plan-driven operational model had contributed to some of these quality issues, as the interview results about feelings of hurry might lead to a lowering of quality standards in some cases.

### **Action plan (Intervention 2)**

The plan of action for the second intervention was formed based on the following concept: increasing quality awareness in the organizational culture and highlighting the importance of quality practices would lead to fewer defects, improved morale, and increased profitability due to reduced warranty work.

### **Taking action (Intervention 2)**

Two forms of action were taken to implement the quality improvements. First, a cultural change was driven through informal team meetings and discussions. Lean thinking, prioritizing work in the order of safety, quality, inventory, productivity (+qip model<sup>25</sup>), and the idea that quality and efficiency are not opposites were also touched upon in these team discussions. Especially for junior team members, the idea of personal review checklists was recommended. The quarterly introduction to the increment planning event was also a good tool for vocalizing leadership support on the importance of quality. Further, one-on-one discussions with each individual developer and team meeting discussions were used to drive the initiative. The second form of improvement was related to the structure of workflow visualization. The kanban boards that were unified earlier for each team were redesigned to include an additional step between the development and testing phases. This step was named “in review,” with the instruction that it was to be used for indicating that the work item was in code review. Automatically, this created peer pressure towards actually conducting code reviews. Hence, it was no longer possible to skip the code review step without the developer making a conscious decision about moving the work item forward. Moreover, the issue management system would retain a record of this decision. At this point, the code review step was still only promoted as recommended practice, not one that should be enforced. The rationale for first promoting and not making it mandatory was to tackle possible resistance to change and drive the change through early adopters and thought leaders, instead of eliciting vocal opposition from laggards.



## **Evaluation (Intervention 2)**

Following the second intervention, people felt that quality was improving and that there were fewer defects. Informal interviews (especially with quality managers, testers, and architects) revealed that the amount of reworking (iterative multiple rounds of bug fixing) had reduced. The comments reflected that this was possibly due to stronger feelings of professional pride from the developers.

PLACE FIGURE 2 ABOUT HERE

To improve the reliability of this finding and evaluate the impact of the intervention, a simple quantitative measure of reported software defects was employed. This revealed a good impact from the first intervention. Further, by simply training and promoting a quality culture, the defect count started to reduce (see Figure 2 and further results in the evaluation of Intervention 3). This started a cultural change in the organizations, and awareness of the importance of quality started to increase.

## **Learnings (Intervention 2)**

The main takeaway from Intervention 2 was that the organization needed to promote quality in addition to agile work management tactics. It was also understood that through quality promotion, a reduction of rework was starting to happen, which could have implications for profitability. It should be noted that without the stronger team workflow visualization being in place, it would not have been possible to increase the adoption of code review practice so easily.

## **Diagnosis (Intervention 3)**

The starting point for another intervention related to the realization from observations that quality practices were not always followed. Thus, further diagnosis was undertaken. At the end of 2018, a crude estimation of the amount of code reviews was conducted by checking the number of requests for reviews and how many were actually completed. It transpired that 63% were requested and only 48% were completed. It was also noted anecdotally that “it could take about a month to get your

code reviewed”. Thus, it seemed that quality practices (especially code reviews) were not always followed.

### **Action plan (Intervention 3)**

Although average quality had started to improve, variations in quality can lead to many problems. Accordingly, the next plan of action was to reduce variation across teams and developers and make quality more consistent.

### **Taking action (Intervention 3)**

The goal of the third intervention was to improve the likelihood of code reviews happening, which was made possible by controlling that code reviews were always conducted. Two mechanisms were used to make this possible. First, it was communicated to the teams that the goal for 2019 was that “all code is reviewed”. Second, the percentage of changes that were actually reviewed was calculated and communicated monthly. This metric of code review coverage increased further from 75% to 88% within a year from the start of the intervention. The remaining code that was not being reviewed mainly involved small fixes, which were excluded due to diminishing returns.

### **Evaluation (Intervention 3)**

Looking at the whole period of the three interventions, the third maintained quality improvements and improved the situation further. The number of defects reduced from an average of almost 300 in 2018 to an average of under 100 in 2020 (Figure 2). Assuming that defect data follows a Poisson distribution ( $\lambda = 164$ ), the likelihood that this reduction could be explained by chance was 0.0000. Hence, the null hypothesis could be safely rejected and the alternative hypothesis could be considered true, which supports the conclusion that quarterly defect amounts reduced. Additionally, there was a clear practical significance, which was observed from the reduction of non-invoiceable hours. A follow-up on the case business further showed that the new practices were still in place at the end of 2020, and the final months of the year set a record for the business unit’s profitability.

While this might have been partially attributable to circumstances (such as less traveling due to Covid-19), it could be speculated that focusing on quality had a positive impact on performance in the form of profitability improvements.

During 2020, an additional metric was introduced to the organization that measured the average cycle time from the start of development to the end of testing. This was possible after the stabilization of team workflows had been achieved. Based on this data, a reduction of cycle time was observed (all team averages per quarter in workdays: 34.6, 20.4, 23.5, 14.7). Thus, it could be speculated that the increased quality also started to reduce cycle time. Since other interventions were conducted (such as highlighting slow-moving work in retrospectives), it is not absolutely clear to what degree cycle-time reduction was caused by quality improvement or other circumstances.

Some additional circumstantial evaluation was possible through yearly employee satisfaction surveys, and a positive trend of improving results continued throughout the research period. While the results could be attributed to many things, based on the free comments in the yearly employee survey data, it would appear that complaints about not understanding the context and feelings of hurry had reduced. There were also direct mentions about the increment planning event being a contributing factor. However, it can be speculated that increased quality and reduced bug fixing also had a positive effect on developer morale. This was a good result, because it has been suggested that increased employee happiness generally results in higher productivity<sup>47</sup>.

### **Learnings (Intervention 3)**

The main learning from the third intervention was that it is not sufficient to simply let teams self-organize and stumble upon quality practices. Further, code reviews need to be enforced, even for agile teams. Controlling for quality practices happening had a clear impact on the quality of the produced software in the investigated organization.

## Discussion of findings

1) The first question that was investigated in this paper was “How can an already agile organization improve its performance further?” Action research was conducted to improve team stability, cross-team communication, and the quality focus of the organizational culture. Further, code review practices were added into an organization that was already following agile practices. As a result, it was demonstrated that an organization that had earlier adopted agile practices was able to improve its performance. There was a significant reduction in reported defects, and secondary results included improvements in employee satisfaction and profitability, which could be partially attributed to the introduced changes.

The current research supports earlier research on the problems of scaling agile development. Typical problems when scaling agile practices to multiple teams include testing and coordination issues<sup>48, 49</sup>, which were found to be present. The interventions reported in this paper were able to address both of these through creating more stable teams and by adapting the Scaled Agile Framework’s increment planning events. Thus, empirical support is given for the applicability of the planning event, even to business contexts where a clear product line or “value stream” is not easy to identify. Motivational benefits and increased shared contexts appeared to be the main wins.

2) The second question for investigation was “What is the impact of promoting quality aspects?” The main portion of the action research focused on testing the impact of quality-promoting interventions. As a result, a substantial reduction in the amount of defects and an overall increase in business profitability, professional pride, and morale were observed. Therefore, this paper adds some empirical evidence that the promotion of quality aspects (especially enforcing code reviews) in addition to normal agile adoption is important for improving the performance of established software businesses. This latter result is in line with a finding from an Asian organization’s review

introduction reported in the SPI manifesto report<sup>50</sup>, although it is reported as an example of a management-driven cultural change towards CMMI maturity. Nevertheless, the code review introduction created similar results.

There has been an unfortunate trend in agile adoption of promoting a kind of revolution against processes, instead of truly integrating real industry needs for such things as product liability and governance requirements<sup>51</sup>. While a revolutionary approach has value in being a working marketing gambit for promoting agility in general, it could be better to go beyond fighting the straw-man called waterfall and start integrating factors such as business model understanding and quality assurance as integral parts of agile best practices. This paper provides a small critique of the agile methods movement and its ability to drive software business performance improvement all the way. The suggested fix hinted at by the reported evidence is to reintroduce a focus on quality as an integral part of agile best practice. The managerial implication is the recommendation that when attempting to gain the benefits of agile transformation, make sure that quality improvement practices (especially code reviews) are a key part of the agenda. It is a bit odd that the Agile Manifesto does not include code reviews as a suggested practice, because it seems to be an easy practice to implement and has real benefits. This omission could be because pair programming seemed like a better (similar) alternative at the time. Still, the practices have been suggested to have complementary benefits from the perspective of team knowledge creation<sup>52</sup>.

### **Limitations and future research**

The main limitation of this study was that the context was quite specific—a well-established business with a strong local market position. In this kind of context, the pursuit of improving quality might not always be the best idea for optimizing profitability. Although high quality can increase sales in the long term, if the business is a startup (or slowly declining), it could be argued that investing in quality is not the most important tactic. This is especially true if their revenue model is

based on charging hourly costs instead of value-based revenue. However, at least for growing business lines, high-quality software is increasingly perceived as a prerequisite to sustain any business over time.

An additional limitation could be that the investigated business unit did not follow agile practices sufficiently, rendering it a less representative sample. It could be argued that some of the problems identified were related to not following scrum guidelines fully. The counterpoint to this is that when compared to the so-called scrum anti-patterns<sup>53</sup>, only a few of them were present in the diagnosed situation, and not all teams had the same issues. Moreover, it could be argued that the case of a poorly followed scrum is more typical in real life compared to an ideal model.

There was also a limitation related to construct validity, which is inherent in action research. Accordingly, it is not entirely clear which additional factors have an impact on the results. Although mixed method data were used, repeating the interventions in another company with similar characteristics of having adopted agile practices (although still having quality and profitability problems) would be interesting. This research has presented an empirical example of how to improve a software-producing organization. While the applicability to general software organizations is limited, there are many mixed-business-model organizations that might be less researched than purer business models. Further, it is not clear if the origin of the identified problems was more related to the complexity of the business model or the scaled agile size. Therefore, further studies in similar contexts are required.

Zorzetti et al.<sup>54</sup> investigated combining user-centered design and lean startup into the agile palette, which can be a good idea for product-oriented new businesses (as was their case). It might be a better idea to start from the quality improvement angle when the context of the business is an

established business with a large existing offering with complex revenue logic. Another avenue of research would be to investigate the contextual analysis as the basis for deciding the improvement approaches with the biggest benefit. It is likely that decisions will be dependent on the business model.

Recently, pair programming has been seen as valuable in creating shared mental models and backup behavior, in addition to improving performance in novel tasks in the context of large enterprises<sup>55</sup>. Incorporating pair programming is not the easiest change, especially with the increase of remote working. Accordingly, it would be useful to evaluate whether similar benefits can be obtained by code review practices. Large companies have to contend with massive code bases. For example, Facebook seems to have adopted a compulsory code review practice as a necessity to enable continuous deployment<sup>56</sup>. Within really large companies, an interesting avenue for future research could be the use of AI to partially automate code review processes. This would require a substantial amount of training data, which can only be collected if the code reviews are conducted and documented appropriately. Thus, it is the opinion of the author that rigorous code reviews remain an essential part of any professional-level software organization.

## **Conclusion**

While the agile movement has fundamentally affected the way software is designed today, many organizations are still struggling to achieve the promised benefits. This paper has outlined some issues that need to be considered. It was highlighted that (re)introducing the practice of code reviews is an essential ingredient of success in software business, which should not be forgotten in the rush to move towards more agile organization structures. This might be especially true in organizations that have either a strong tradition of being driven by financial numbers or value capture-focused management, particularly compared to value creation-focused newer organizations (such as startups or other new product development organizations).

The nature of qualitative research is to explore the investigated context, and definitive answers are not claimed. Regardless, the likelihood of the following conclusion seems higher: even for organizations that have used agile practices for a while, if the organization has a complex business model (B2B, mix of services/products) and is larger than approximately six teams, scaling agile problems are likely. These problems can be improved by introducing more cross-team communication (e.g., increment planning events and stabilizing team structures) to allow for learning to happen and by promoting quality awareness. Further, if the code base is large and complex, benefits from agile testing practices are harder to implement. Therefore, introducing compulsory code reviews can be a recommended course of action to improve organizational performance.

## REFERENCES

1. Kalenda M, Hyna P, Rossi B. Scaling agile in large organizations: Practices, challenges, and success factors. *Journal of Software: Evolution and Process* 2018; 30,10:e1954.
2. Kruchten P. Contextualizing agile software development. *Journal of Software: Evolution and Process* 2013;25,4:351-361.
3. Zott C, Amit R, Massa L. The business model: recent developments and future research. *Journal of Management* 2011;374:1019-1042.
4. Vanhala E, Saarikallio M. Business model elements in different types of organization in software business. *International Journal of Computer Information Systems and Industrial Management Applications* 2015;7:139-150.
5. Ulaga W. Capturing value creation in business relationships: A customer perspective. *Industrial marketing management* 2003;32,8:677-693.



6. Helander N, Ulkuniemi P. Customer perceived value in the software business. *The journal of high technology management research* 2012;23,1:26-35.
7. Parry S, Rowley J, Jones R, Kupiec-Teahan B. Customer-perceived value in business-to-business relationships: A study of software customers. *Journal of Marketing Management* 2012;28,7–8:887–911.
8. Takeuchi H, Nonaka I. The new product development game. *Harvard Business Review*, 1986;64,1:137-146.
9. Abrahamsson P, Warsta J, Siponen MT, Ronkainen J. New directions on agile methods: A comparative analysis. In *25th International Conference on Software Engineering*, 2003. *Proceedings*. 2003;244-254 IEEE.
10. Saarikallio M, Tyrväinen P. Following the Money: Revenue Stream Constituents in Case of Within-firm Variation. In *International Conference of Software Business*. 2004:88-99 Springer, Cham.
11. Abrahamsson P, Salo O, Ronkainen J, Warsta J. Agile software development methods: Review and analysis, VTT publication 478 2002. Espoo, Finland, 107.
12. Beck K, Beedle M, Van Bennekum A, Cockburn A, Cunningham W, Fowler M, ... Thomas D. *Manifesto for agile software development*. 2001.
13. Dikert K, Paasivaara M, Lassenius C. Challenges and success factors for large-scale agile transformations: A systematic literature review. *Journal of Systems and Software*. 2016;119:87-108.
14. Abrar MF, Khan MS, Ali S, Ali U, Majeed MF, Ali A, ... Rasheed N. Motivators for large-scale agile adoption from management perspective: A systematic literature review. 2019. *IEEE Access*, 7, 22660-22674.

15. Napoleão BM, de Souza ÉF, Ruiz GA, Felizardo KR, Meinerz GV, Vijaykumar NL. Synthesizing research on Knowledge Management and Agile Software Development using the Meta-ethnography method. *Journal of Systems and Software*. 2021;178:110973.
16. Karlstrom D, Runeson P. Combining agile methods with stage-gate project management. *IEEE software*. 2005;22,3:43-49.
17. Dzamashvili Fogelström N, Gorschek T, Svahnberg M, Olsson P. The impact of agile principles on market-driven software product development. *Journal of software maintenance and evolution: Research and practice*. 2010;22,1:53-80.
18. Choudhary V. Comparison of software quality under perpetual licensing and software as a service. *Journal of management information systems*. 2007;24,2:141-165.
19. Sfetsos P, Stamelos I. Empirical studies on quality in agile practices: A systematic literature review. In 2010 Seventh International Conference on the Quality of Information and Communications Technology. 2010:44-53. IEEE.
20. Fagan ME. Design and code inspections to reduce errors in programs. *IBM Systems Journal*. 1979;15,3:219-248.
21. Davila N, Nunes I. A systematic literature review and taxonomy of modern code review. *Journal of Systems and Software*. 2021;110951.
22. Brykczynski B, Meeson R, Wheeler DA. Software inspection: eliminating software defects. Institute for Defense Analyses. 1994. Alexandria, Virginia.
23. McIntosh S, Kamei Y, Adams B, Hassan AE. An empirical study of the impact of modern code review practices on software quality. *Empirical software engineering*. 2015; 21,5:2146-2189.
24. Dosea M, Sant'Anna C, Oliveira Y, Colaco Junior M. A Survey of Software Code Review Practices in Brazil. 2020. arXiv e-prints, arXiv-2007.

25. Protzman C, Whiton F, Kerpchar J, Lewandowski CR, Stenberg S, Grounds P, Bond, J. Getting Ready to Implement Lean System. In *The Lean Practitioner's Field Book*. 2018. Productivity press.
26. Magar VM, Shinde VB. Application of 7 quality control (7 QC) tools for continuous improvement of manufacturing processes. *International Journal of Engineering Research and General Science*. 2014;2,4:364-371.
27. Poppendieck Mary. *Implementing lean software development: From concept to cash*. 2006. ISBN 0-321-43738-1. Pearson. Boston.
28. Jamkhaneh HB, Shahin R, Shahin A, ArabYarmohammadi M. CMMS software quality function deployment based on maintenance objectives: a framework for software selection process. *International Journal of Productivity and Quality Management*. 2021;32,4:413-439.
29. Emerton M, Panesar SS, Forrest K. Safer surgery: how a checklist can make orthopaedic surgery safer, *Orthopaedics and Trauma*. 2009;23,5:377-380, ISSN 1877-1327.
30. Bohm R, NOT FLYING BY THE BOOK: SLOW ADOPTION OF CHECKLISTS AND PROCEDURES IN WW2 AVIATION. 2013. San Diego.
31. Hyde P. *10 Ways Your Agile Adoption Will Fail*. 2019. Gartner.
32. Runeson P, Höst M. Guidelines for conducting and reporting case study research in software engineering. *Empirical software engineering*. 2009;14,2:131-164.
33. Eisenhardt KM. Building theories from case study research. *Academy of management review*. 1989;14,4:532-550.
34. Baskerville RL. INVESTIGATING INFORMATION SYSTEMS WITH ACTION RESEARCH. *Communications of the association for information systems*. 1999;2,1:19.
35. Coghlan D, Brannick T. *Doing action research in your own organization*. 2001. London, Sage.

36. Lewin K, Frontiers in group dynamics: II. Channels of group life; social planning and action research. *Human relations*. 1947;1,2:143-153.
37. Järvinen P. Improving guidelines and developing a taxonomy of methodologies for research in information systems. 2021. Jyväskylä. JYU dissertations.
38. Creswell JW, Klassen AC, Plano Clark VL, Smith KC. Best practices for mixed methods research in the health sciences. Maryland: National Institutes of Health 2013 2011:541-545.
39. Kaplan B, Duchon D. Combining qualitative and quantitative methods in information systems research: a case study. *MIS quarterly*. 1988:571-586.
40. Franke RH. Technological revolution and productivity decline: Computer introduction in the financial industry. *Technological Forecasting and Social Change*. 1987;31,2:143–154.
41. Watkins M. The first 90 days – critical success strategies for new leaders at all levels. Boston. Harvard Business School Press. 2003:45-46.
42. Elo S, Kyngäs H. The qualitative content analysis process. In: *Journal of Advanced Nursing* 2008;62,1:107–115. <https://doi.org/10.1111/j.1365-2648.2007.04569.x>
43. Strauss A, Corbin J. Basics of qualitative research (2nd end). 1998. Thousand Oaks, CA, Sage.
44. Meyer CB, Stensaker IG. Developing capacity for change. *Journal of Change Management* 2006;6,2:217-231.
45. Agrawal M, Chari K. Software effort, quality, and cycle time: A study of CMM level 5 projects. *IEEE Transactions on software engineering*. 2007;33,3:145-156.
46. Paasivaara M. Adopting SAFe to scale agile in a globally distributed organization. In 2017 IEEE 12th International Conference on Global Software Engineering (ICGSE) 2017:36-40. IEEE.
47. Graziotin D, Fagerholm F, Wang X, Abrahamsson P. What happens when software developers are (un)happy. *Journal of Systems and Software*. 2018;140:32-47.

48. Petersen K, Wohlin C. The effect of moving from a plan-driven to an incremental software development approach with agile practices. *Empirical Software Engineering*. 2010;15,6:654-693.
49. Berntzen M, Stray V, Moe NB. Coordination Strategies: Managing Inter-team Coordination Challenges in Large-Scale Agile. In: Gregory P, Lassenius C, Wang X, Kruchten P (eds) *Agile Processes in Software Engineering and Extreme Programming. XP 2021. Lecture Notes in Business Information Processing*, 2021;419. Springer, Cham.
50. Pries-Heje J, Johansen J. Spi manifesto. European system & software process improvement and innovation. 2010. Accessed online 2021: [eurospi.net](http://eurospi.net).
51. Ebert C, Paasivaara M. Scaling agile. *IEEE Software*, 2017;34,6:98-103.
52. Spohrer K, Kude T, Schmidt CT, Heinzl A. Knowledge creation in information systems development teams: The role of pair programming and peer code review. *ECIS*, 2013;CR,213.
53. Eloranta VP, Koskimies K, Mikkonen T. Exploring ScrumBut—An empirical study of Scrum anti-patterns. *Information and Software Technology*. 2016;74:194-203.
54. Zorzetti M, Signoretti I, Salerno L, Marczak S, Bastos R. Improving Agile Software Development using User-Centered Design and Lean Startup. *Information and Software Technology*. 2022 (in press);141:106718.
55. Kude T, Mithas S, Schmidt CT, Heinzl A. How Pair Programming Influences Team Performance: The Role of Backup Behavior, Shared Mental Models, and Task Novelty. *Information Systems Research*. 2019;30,4:1145-1163. doi:10.1287/isre.2019.0856
56. Distefano D, Fähndrich M, Logozzo F, O'Hearn PW. Scaling static analyses at Facebook. *Commun. ACM* 2019;62,8 (August 2019):62–70. DOI:<https://doi.org/10.1145/3338112>

*Table 1. Initial interviews included 41 people.*

Role	Informant count	Experience in business unit		
		<3y	3-5y	>5y
Programming	15	6	4	5
Architecture (technical)	6	1		5
Managing projects	5	1	2	2
Managing service	2	1	1	
Product management / Analysis / Business expertise	8		3	5
Scrum Master/PO	3	1		2
Others(sales, QA)	2			2

Table 2. Themes emerging from the interviews formed the basis of problem diagnosis.

Main category	Category	Comments	Example codes
Cultural issues & instability	Hurry	46	hurry, no free developers, uneven workload, impossible schedule promises, firefighting, overworked architects
	Turnover	12	experienced people leaving, people leaving, cannot recruit new people, new people taking large shoes
	Negative culture	10	no ability for renewal, self-organizing was a red flag, blaming, separation between management and production
	Instability	8	moving people around, borrowing people, people frustrated about throwing people around
	Team spirit	8	team spirit, team doesn't feel shared responsibility, team member feel alone, sprint of working together
	Resourcing	7	resourcing challenges, projects competing for same resources, steering groups do not make decisions only discussions
	Interruptions	6	lot of interruptions, things are fragmented, frustration
	Role clarity	4	lack of role clarity
	Positive culture	2	great developers, good people who support each other and help
	Vision	Product vision/management	32
Lack of leadership		27	reluctancy to decide, following up on execution, people are seen as cost, team model needs to work better, no strategy
Context		15	no context, information through rumors, no discussion between projects
Pricing/selling		5	selling non-profitably, no clear pricing model, expanding business beyond legacy
Process & quality	Process	27	lack of professionalism, we take shortcuts all the time, no clear processes, sprints leaking, chaotic operations, homey feeling
	Testing	14	quality assurance weak, no planned testing, regression tests are done if there is time, not enough automated tests
	Documentation	8	no documentation, very light document reviews, db documentation, manuals are not updated
	Code reviews	7	no code reviews, code reviews are difficult, Only 2 years of code reviews at all, no interest in quality
	Scrum issues	3	scrum masters are in a difficult spot
	Refactoring	3	no refactoring, no interest in quality
Project management	Maintenance	3	maintenance issues, maintenance should be inside teams, Maintenance not valued enough just fixing quickly
	Definition of Done	2	no definition of done, cross team reviews would be good
	Administrative project manager	24	project management is administrative, poor planning, finalizing large projects is hard, spoonfeeding style, haven't seen project plans
	Byrocracy	2	reputation of bureaucratic processes
Knowledge & customer	projects vs. product	2	product vs. customer vs. project
	Knowledge	18	silobed knowhow, knowledge only in some peoples head, too few experts, new don't get up to speed, individual performance
Product & quality	Customer	17	more customer facing people, good customer relationship, too much following customer's whistle, reactive mode
	Technical complexity	12	monolithic product, complex product, large/huge codebase, difficult to learn product
	Continuous integration	12	too many code versions, breaking other branches, change logs are difficult to make, merging code across products and customers
	Technical debt	10	too much tech debt, duplication, quality varies, bugs, maintenance is caused by poor quality
Other	Product quality	6	lack of functionality, good package but poorly wrapped, tailored configurations per customer, not a real product
	Advice for researcher	4	lets not try too much at once, listen to us and be critical, need a proactive convincing attitude, only the loudest get a voice

Table 3. Interventions had empirical and theoretical motivations.

Stage	Time period	Motivation from interviews and data	Theoretical assumptions
Intervention 1: Improving communication, visual magement, and team stability.	Q2-2018 → Q4-2019	Key people were leaving, complaints about lack of context, people felt they were "jumping" around, context was unclear.	Scaling agile was failing, team learning did not have time to happen.
Intervention 2: Promoting quality, and recommending code reviews.	Q3-2018 → Q3-2019	Complaints about lack of testing, process and product quality issues.	Scrum was the main agile method used and it does not promote quality well enough.
Intervention 3: Enforcing code reviews.	Q3-2019 → Q4-2020	Measuring amount of reviews showed it was not always done.	More code reviews improve quality.

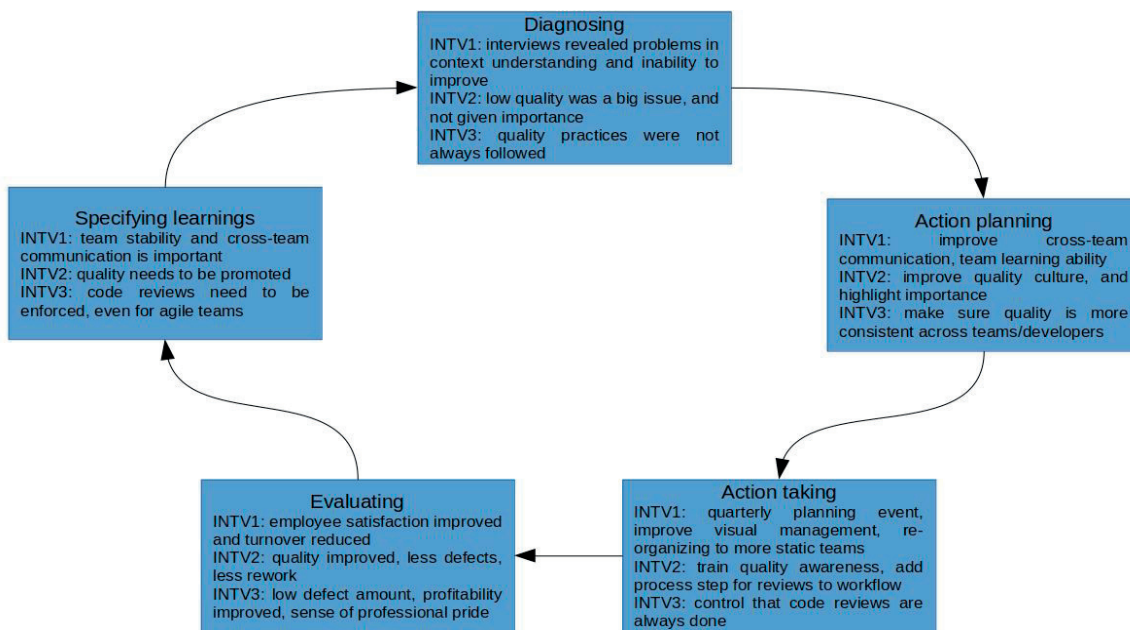


Fig. 1. Action research cycle included three interventions.



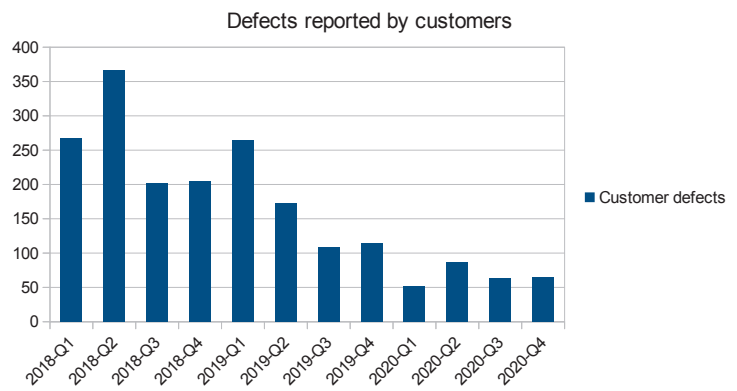


Fig. 2. Quantitative results indicate quality improvement.