Received 30 November 2020; | Revised 6 April 2021; | Accepted 9 May 2021
DOI: xxx/xxxx 10.1002/smr.2363

RESEARCH ARTICLE - EMPIRICAL

Enhancing Agile Software Development in the Banking Sector – A Comprehensive Case Study at LHV

Ezequiel Scott*[1] | Fredrik Milani[1] | Erki Kilu[2] | Dietmar Pfahl[1]

[1]Institute of Computer Science, University of Tartu, Tartu, Estonia
[2]LHV Bank, Tartu, Estonia

Correspondence
*Ezequiel Scott, Institute of Computer Science, University of Tartu, Tartu, Estonia. Email: ezequiel.scott@ut.ee

Present Address
Present address

Abstract

FinTech companies are challenging established financial institutions' dominance by offering the same products with a superior customer experience and delivering new features faster. The adoption of agile software development partially enables this competitive advantage. In response to this challenge, banks explore how they can improve their agile processes. LHV, a mid-sized bank, uses agile practices but faces the challenge of further improvement to stay competitive with FinTech companies. In this article, we explore how LHV can improve its agile software development process. We conduct a case study at LHV where we first derive eight change proposals based on a literature review and interviews. Then we report on how LHV implemented the change proposals and their perceived impact. Our results stress the importance of taking a coherent approach to improving agile processes by considering both business units and operations involved in the product life-cycle. It is also necessary to align organizational structures to enable team autonomy by, for instance, decentralizing decision authority. Finally, it is beneficial to adapt agile practices to their context and have an IT architecture and technology supporting the agile approach.

KEYWORDS:
Agile Software Development; Financial Institutions; FinTech; Software Process Improvement

1 | INTRODUCTION

The dominance of established financial institutions has been challenged by organizations that use information technology to provide financial services with better quality of service to customers [1]. Such companies are commonly referred to as FinTech (Financial Technology) companies, and their main competitive advantage lies in how they develop their products. For instance, incumbent financial institutions have many products supported by legacy systems and complex system infrastructure [2]. FinTech companies, on the other hand, focus on a few niche products supported by modern systems [2] and compete with incumbent financial institutions by offering the same products but with the competitive advantage of responding to customer needs and delivering new features at a faster pace [3].

Incumbent financial institutions have traditionally relied on plan-driven approaches for developing software [4,5]. Plan-driven approaches have been used to develop mainframe systems and, in collaboration with others, infrastructural solutions for financial transactions, such as SWIFT (Worldwide Interbank Financial Communications) for cross-border payments. However, such approaches do not meet the need for speed in developing and releasing new banking products and services [6,7]. FinTech companies, on the other hand, have adopted agile methods that allow them to respond faster to customer needs [8].

In response to this challenge, companies have examined improving their software processes [9,10] and, more specifically, agile software processes [11,12]. In this regard, critical success factors for agile methodologies have been studied from different perspectives related to adopting agile values [13,14] and executing agile projects [6,15]. Financial institutions have also examined adoption of agile development methods to better compete

[0]**Abbreviations:** FinTech, Financial Technology; SPI, Software Process Improvement; FI, Financial Institution; SDUM, Software Development Unit Manager

with FinTech companies[8], such as a Scandinavian bank[16], a large bank in the USA[5], a Dutch bank[17,18], and an Australian financial institution[19]. However, studies examining critical success factors do not specify how such factors are translated into reality. For instance, management support and agile mindset have been proposed as critical success factors[13,15] without defining how, specifically, they are translated into reality or expressed in the software development process once implemented. Furthermore, case studies predominantly cover banks adopting agile methodologies but not how banks that already have an agile software development process can improve them. In this context, we study LHV Bank (originally Lõhmus, Haavel & Viisemann).

LHV grew rapidly and identified the need for adopting an agile methodology and combined Scrum and Kanban methodologies. In recent years, LHV has experienced rapid growth of new customers, products, and market share. The growth has also led to an increase in the number of developers. LHV noted that the growth inhibited software development speed and their ability to compete with FinTech companies to deliver new features to the customers. Thus, although LHV employs agile methodologies, they have not been able to scale up the software development process to compete with FinTech companies effectively. Therefore, our research question focuses on finding out how the agile development processes at LHV can be improved.

This article is an extension of a short paper[20]. In our previous work, we conducted a literature review and interviews to elicit a set of change proposals to the existing agile development process at LHV Bank. In this article, the focus is on the implementation of the change proposals and their perceived impact. Thus, the main contribution is two-fold. First, we document the actual changes made based on the proposals and their implications on the agile development process once implemented. Second, we describe the impact of those changes, as perceived by the software development processes' stakeholders. We believe this case study provides an example that can be used by financial institutions interested in improving their agile software development processes.

To address our research question, we extended our original case study such that it now consists of two parts. In the first part, we reviewed literature and conducted in-depth interviews with FinTech companies to identify proposals for improving the existing agile development process. The management board of LHV discussed the proposed changes and initiated their implementation. Different to our previously published short paper, this part is presented more comprehensively in this article. In the second part of the case study, the extension, we conducted interviews with key people at the bank to explore how the change proposals were implemented and whether they produced the desired impact.

This article is organized as follows. Section 2 presents the related work, while Section 3 describes the enhanced case study. Then, in Section 4, we present the results, followed by a discussion and limitations in Section 5. We conclude the paper and outline possible directions for future work in Section 6.

2 | RELATED WORK

Most, if not all, companies engaged in developing software either face issues, such as slow or inconsistent delivery, that they seek to address or aim at further improving their software development processes. In this regard, Software Process Improvement (SPI) is understood as detecting inefficiencies in software development processes and resolving them[21]. As SPI can be conducted in different ways, studies have examined factors associated with successful SPI. For instance, Dyba[9] surveyed 120 companies to identify key factors for successful SPI. The results show that organizational aspects, such as creating a culture within which procedures can succeed, is as important as the technical procedures themselves. Likewise, Rainer and Hall[10] identified eight factors, such as training and mentoring, leadership, and internal process ownership, to be key. The same question is explored by Niazi et al.[22] but from the practitioners' perspective. Practitioners consider, among other factors, adoption of current technologies, leadership, and commitment from management to be essential factors. These studies conducted surveys across different industries to associate factors with successful SPI. On the other hand, we explore how such factors are expressed and applied when improving the software development process of a bank.

Software developers engage in SPI by adopting agile development methods. While agile was originally designed for smaller and co-located teams[23,15], it is also applied and studied on larger-scale contexts[24]. For instance, Razavi and Ahmad[11] reviewed studies to map agile software development challenges in large organizations. Putta[12], on the other hand, reviewed studies to examine why and how large and globally distributed organizations adopt agile methods. Other studies, such as in Laanti and Kettunen[25], focus on adopting a specific framework such as SAFe (Scaled Agile Framework). In contrast to such reviews that study agile in large-scale and global organizations, we focus on improving agile methods within a specific SME financial organization that already has familiarity with agile software development.

The implementation and further development of agile methods have not always produced expected results[26]. Therefore, studies have examined critical success factors (CSF) for agile methodologies. CSF has been examined for implementing agile methodologies. For instance, Dikert et.al.[13] studied CSF related to adopting agile in large-scale settings, Livermore[14] examined CSF when implementing agile software methods, and Misra et.al.[27] considered CSF for adopting agile practices from practitioners' perspective. Aldamash et.al.[7] and Chow and Cao[6], on the other hand, studied CSF in agile projects whereas Lindvall et.al.,[15] examined CSF by conducing an eWorkshop with experts. These studies show that, among other factors, management support [13,14], agile mindset [13,15], team composition and competence [6,15,27,7], autonomous teams [28,29,30,31,32,33], training and coaching [13,6,14,27,15], regular delivery [6,7], suitable technological support tools for agile [15], and adapting and combining agile practices [5,13,34] are important factors for successful agile software development. While these studies elicit CSF from larger sets of data (literature reviews and surveys), we examine how CSF can be implemented in software development processes. Thus, our work is complementary.

SPI has also been applied to financial institutions. Berkani et al. [35] conducted a case study at a central bank to study the triggers that enable an organization to adopt agile methods. They concluded that a gradual and cyclic approach is favourable. Roses et al., [36] surveyed a Brazilian bank to develop a framework for assessing if conditions are favourable for adopting agile methods. Hajrizi and Bytyci [4] studied the implementation of agile methods in a Kosovar financial institution and concluded that agile approaches' successful implementation depends on organizational structure and culture. Reddy [5] studied a large bank that implemented Scrumban but still encountered challenges with slow and unreliable software delivery. The author concluded that successful agile implementation requires focusing on the entire process and not on specific parts of the process. In another study, Mikalsen et al. [37] observed that agile transformation required the company to retrain its project managers into scrum masters. A sizeable Scandinavian bank decided to adopt an agile development methodology to support their digital banking platform [16]. To this end, agile teams were formed, and employers were trained in agile practices. Other researchers [19,37,38,39] studied the implementation of agile methodologies within financial institutions and observed the necessity of adopting an agile mindset in the development organizations. These studies focus on banks transition from waterfall-inspired to an agile approach. On the other hand, our work considers improving software processes at a bank that has already adopted an agile approach. Therefore, our work is complementary in that we examine the next step, i.e., to improve agile software development processes further.

In summary, working with SPI can be challenging. Adopting agile approaches to improve software processes, specifically in larger financial institutions, such as banks, is growing. Therefore, it is relevant to study how agility can be effectively implemented at such companies. However, existing research focuses on generic critical success factors without examining how, precisely, such factors can be implemented. Furthermore, case studies in financial institutions predominantly focus on adopting agile methodologies. Our work is complementary in that we study a case of a bank that seeks to improve an already adopted agile approach. In addition, our work provides insights into how critical success factors can be implemented.

3 | METHODOLOGY

In this section, we present the research question, the case study setting, and the design. In our previous work [20], we elicited change proposals by examining the agile development process at LHV, literature review, and interviews with FinTech companies (steps (1) to (3) in Figure 1). In this article, we focus on implementing the change proposals and their perceived impact (steps (4) and (5), Figure 1).

3.1 | Research Question

Case study research methodologies use predominantly qualitative methods to examine a particular reality within its real-life context and, particularly, when the boundaries of the object of study and its context are not clear [40,41]. Case studies can be used to confirm hypotheses [40,42], evaluate a specific method [43], or for exploratory purposes [40,42]. We employ the case study methodology because we seek to explore how software development processes at a financial institution can be improved and the perceived impact. Thus, we conduct an exploratory study of the software development process (object), which is not clearly distinguished from its context, to explore the following research question: *"How can the agile software development process at LHV be improved?"*

3.2 | Case Study Setting and Design

Given the above-defined research question, we sought a case that (1) is a financial institution seeking to improve their software development process, and (2) provides access to information. To this end, we selected a relatively young bank that sees itself as innovative and entrepreneurial. LHV[1] was founded in 1999 and has offices in Tallinn and Tartu, Estonia. The bank has about 550 employees and more than 300000 clients. The IT division of LHV is divided into two departments: development and operations. The setting of the study is the software development processes. The bank seeks to improve its software development process in order to become more competitive.

[1] LHV website – https://www.lhv.ee/

The case study design consists of five steps depicted by Figure 1 . Steps (1) serves to gain an understanding of the software process development at LHV before the software process improvement. Step (3) aims to elicit proposals on how to improve the process from both the literature and interviews financial institutions and FinTech companies. Steps (4) and (5) examine how the proposals were implemented and their effects as perceived by the bank.

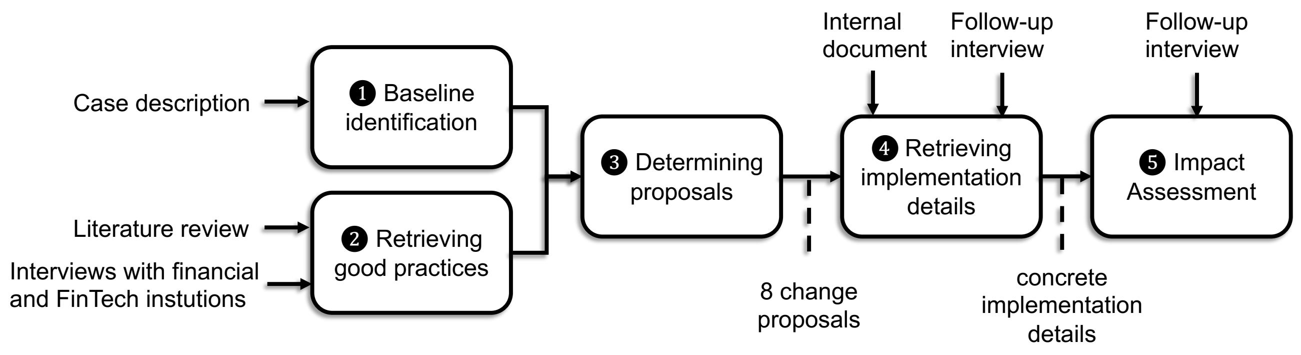
Case description
1 Baseline identification
Literature review
Interviews with financial and FinTech instutions
2 Retrieving good practices
3 Determining proposals
8 change proposals
Internal document
Follow-up interview
4 Retrieving implementation details
concrete implementation details
Follow-up interview
5 Impact Assessment

FIGURE 1 Case study design.

The first step, **(1) Baseline identification**, is to understand the state of the software development process at LHV before the software process improvement. We present the situation as described by the Chief-Executive Officer (CEO) of the bank. The second step, **(2) Retrieving good practices**, aims to elicit proposals to improve the process. To determine these proposals, we use a mixed-method approach consisting of a literature review and interviews with employees at both FinTech companies and financial institutions (FI).

The literature review summarizes the proposals as described in papers on agile software development within the context of financial institutions. The literature review was conducted by the third author whereas the first and second authors reviewed the results. The included papers discuss different aspects of agile methodologies, ranging from issues when applying agile methods in software development to case studies that report on adopting agile methods in financial institutions. It is worth noting that we found only a few studies describing agile methods in large financial institutions. For this reason, we decide to complement the review with interviews.

The interviews were conducted with domain experts from FinTech companies and financial institutions. In total, four experts participated in the interviews, and each one represented a different company. The interviewees worked for companies that use agile methods and practices for their in-house development process. The interviewees were acquainted with software development processes and had at least ten years of experience with agile. The companies covered different sizes, variety in years of existence and have experienced customer growth. Table 1 summarizes the companies that participated in the study.

The interviews were conducted by the third author in January and February 2018, and the first and second authors reviewed the findings. The interview guideline consists of semi-structured questions and is publicly available [2]. The interviewees did not consent to have the interviews recorded; consequently, we relied on the notes taken.

TABLE 1 Companies interviewed

ID	Company description	Staff	IT Staff
FI-1	A full-service and mature bank offering financial products to retail, corporate, and institutional customers.	2300	500
FI-2	A growing mid-sized bank leading in consumer finance products with strong growth in the past decade.	450	100
FI-3	A leading FinTech company offering that began to offer cross-border money transfer almost ten years ago and has grown to cover most of the world's currencies.	1200	250
FI-4	A 5 years-old leading FinTech company specialized in offering standard bank accounts and card services.	100	30

[2] Interview guideline available in Appendix A

In the third step, **(3) Determining proposals**, we took the input from the two previous steps (the baseline identification and the retrieval of good practices) to elicit a set of change proposals that suits LHV's context. The third step resulted in the elicitation of eight change proposals. In the fourth step, **(4) Retrieving implementation details** we examined the concrete implementation details related to the change proposals. The management board of LHV examined the change proposals and, in August 2018, held a series of meetings where they discussed their implementation. Following these meetings, LHV produced an internal document outlining how to implement the change proposals and how they were implemented. We analyze this documentation and complement it with answers provided by the interviewees during a follow-up study.

In the follow-up study, we interviewed eight persons between December 2019 and January 2020 following a semi-structured interview format. Table 2 lists the profile of those interviewed. All the participants consented to be recorded. Each interview lasted between 1-1.5 hours. In total, 9 hours of audio were transcribed using audio to text software. We were also given access to internal documentation. The transcripts were cleaned and analyzed by following the guidelines proposed by Saldana[44]. In the first round, a research assistant coded the transcripts into themes related to the change proposals. Then, in a second iteration, the themes were refined and mapped into implementation and assessment topics, organized by participant. For instance, for the second change proposal, we used the theme "organize relevant training". Then, excerpts from interviews that related to this theme were extracted and categorized under this theme. Next, we analyzed the excerpts under each theme. Examples of the refined themes are "how it was before", "what was proposed", "what was expected", "what was done", "how did it impact", and "what is lacking". The final analysis was reviewed by the first and second author, and a draft of the results was shared with the research participants for review and discussion.

The fifth step, **(5) Impact assessment**, is to understand the impact of the changes implemented. To this end, we analyze the second part of the follow-up interviews, which focuses on the changes' perceived impact. The analysis procedure is the same as the one applied during the previous step.

TABLE 2 Participants of the follow-up interview.

Code	Position	Area	Office
CEO	CEO	–	Tallinn
PO1	Product Owner	Retail Credit Products	Tallinn
PAM1	Product Area Manager	Payment Products	Tallinn
PAM2	Product Area Manager	Investment Products	Tallinn
HSE	Head of Software Engineering	–	Tartu
SDUM1	Software Development Unit Manager	Payment Products	Tartu
SDUM2	Software Development Unit Manager	Investment Products	Tartu
HTT	Head of Technical Team	–	Tallinn

3.3 | Organizational Context

When describing the organizational context as a baseline of our study, we focus on the development department that consists of seven development teams and one R&D initiative team. These teams are collectively responsible for 24 products. Each team consists of 8 to 12 persons and is headed by a Software Development Unit Manger (SDUM). The teams also have one Analyst, one Lead Software Engineer, 3 to 6 Software Engineers, and 1 to 3 Quality Assurance Engineers.

Product Owners are responsible for creating the product vision. Once the management board approves the product vision, the Product Owners collaborate with the SDUMs to plan and outline an annual product roadmap. The roadmap consists of epics, which enable rough time-estimations to facilitate resource planning and serve as input for prioritization. The development teams work with multiple products. Therefore, product owners have to agree with SDUM on the prioritization of epics. Commonly, those aligned with the company's strategic goals and those related to regulatory changes with external deadlines have higher priority. The SDUMs and Analysts divide the Epics into smaller tasks. The analysts described them in detail.

Once the task is detailed, the engineers begin the development. The team does the development. Once a task is fully implemented, the quality assurance engineers test the code. Testing activities include both manual and regression testing. If there are bugs, the task is sent back to the

[2]Interview guideline available in Appendix B.

engineers. In addition to working on the implementation of tasks, the teams have a "weekly backlog" for smaller tasks and bug fixes. Weekly backlog serves to avoid neglecting minor issues and bugs. The tasks in the weekly backlog are also described and prioritized by the Product Owner.

If no bugs are found during the testing process, the task is sent to the Product Owner, who approves it. Once a task is approved, the task is sent to the responsible SDUM for inclusion in the release. The engineers will release the task, and the system administrator will deploy it. The task is considered complete when the Product Owner has launched it.

LHV describes their software development process as agile since it includes several agile practices: the role and responsibilities of the Product Owners, the requirements organized as Epics and tasks, the relatively small size of the teams, and the work transparency achieved by the use of Kanban boards. However, the process has many characteristics that are similar to plan-driven approaches. For instance, teams are mostly functional since there are specific roles such as Analysts, Quality Engineers, Software Engineers. Moreover, the activities related to the implementation and testing of work items follow a plan-driven structure. Figure 2 illustrates the software development at LHV and how the roles are related to the different activities.

4 | RESULTS

In this section, we present the results of our case study organized according to the design steps described above.

4.1 | Identified Main Challenges

LHV identified three main issues related to the organizational structure and the software development process. The first one concerns the imbalance between products and development teams. Seven teams develop Twenty-four different products. Some teams support products managed by different product owners, causing complicated procedures for agreeing, prioritizing, and roadmap development. Often, higher levels of management have to resolve conflicts. The second issue is related to the software development teams. Although the current software development process involves several agile practices, the separated roles (e.g., SDUM, Analyst, Engineer, Quality Engineer) results in a development process split into small and sequential steps that introduce delays. The third issue is related to releases. The releases, planned bi-weekly, are time-consuming, and they cause bottlenecks as other departments are involved.

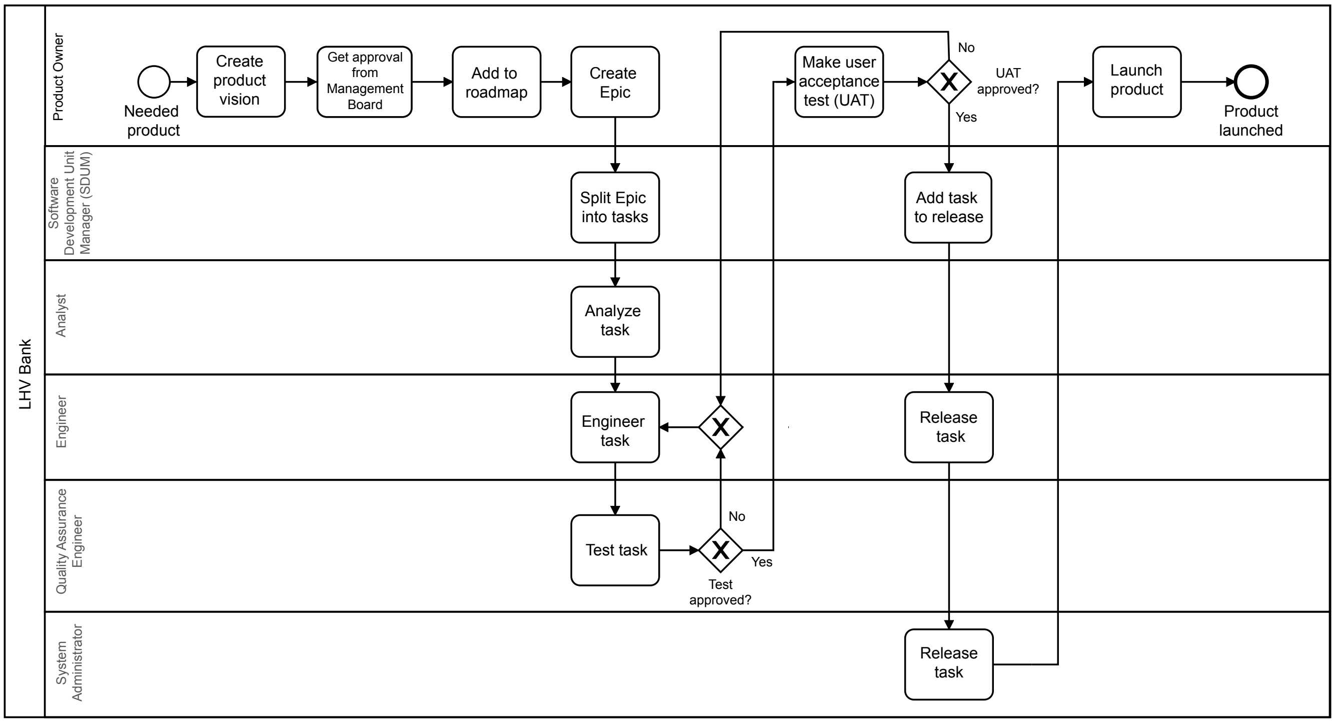

FIGURE 2 The software development process at LHV.

4.2 | Retrieving Good Practices

In this sub-section, we present the results from the second step of the case study design, i.e., retrieving good practices. These were derived from literature review and interviews and are, therefore, presented in that order.

4.2.1 | Findings from Literature

The review of the literature includes eleven publications that were reviewed and analyzed. All the publications were related to agile software improvement in the context of FinTech and financial institutions.

Agile is about the mindset of the whole organization. In 2015, a study conducted with financial institutions in Kosovo revealed that successful implementation of agile approaches depends on the organisation's structure and its culture[4]. Agile is about the mindset shared by the whole organization; that is, the attitude and behaviour of the employees beyond the given set of procedures, techniques, and rituals[28]. Financial institutions that promote collaboration and a culture of cooperation are in better standing and accept more easily agile improvements[14].

Different agile methods and practices should be combined. In 2016, the Scrumban concept was implemented in a large bank in the USA. The term Scrumban refers to the application of Kanban within a Scrum context[5,45] since Lean methods like Kanban can extend the benefits of Scrum. Although the bank had begun its agile transformation, it still struggled with slow and unreliable delivery of output due to the focus being on improving the delivery process's components rather than the entire process[5].

Agile training enables organizations to implement agile practices better. Agile training is one of the success factors for implementing agile software development[14]. Organizations that provide training to the teams have a more successful implementation of agile practices than organizations that do not provide that. The training enables organizations to develop know-how and prepare better for the implementation of the methodology. In 2014, the Scandinavian bank decided to renew its digital banking platform and use an agile development methodology to achieve that. As the organization and project were large, they decided to use SAFe. In total, 80 people were trained, and the Agile Release Train consisted of five development teams. The bank concluded that proper training was vitally essential, improving the delivery system significantly[16].

Following agile principles in full brings success.

Implementation and adoption of agile methods seem to deliver value when it is fully applied. Applying selected agile practices in a software development process that is fundamentally plan-driven or in contexts not aligned with an agile mindset can restrict the benefits of agile methods. Several case studies, such as KeyCorp[39], Suncorp[19], and Jyske Bank[38], have shown that partial implementation of agile methods can cause issues. For instance, at KeyCorp[39], they trained project managers into Scrum Masters. However, the development teams did not fully understand and perceive the new role and continued to see them as project managers. At Suncorp[19], they used an agile method for a major system replacement. However, failing to implement backlog management concepts, including discussion with the team and stakeholders for task prioritization, caused challenges with delivering the project within the estimated scope of time and cost. Similarly, Jyske Bank[38] noted that the planning reverted to the old plan-driven approach they aimed at replacing.

In summary, the literature review shows that successful agile transformation at financial institutions requires changes at both organizational and development level. From an organizational perspective, agile is about the mind-set of the whole organization. Furthermore, different agile methods and components should be combined, and training enables organizations to develop know-how and prepare better for the implementation of the agile methods. From a software development perspective, the literature review indicates that following agile principles in full contributes to success. Table 3 summarizes these findings along with their references.

TABLE 3 Good practices retrieved from the literature and the interviews with FinTech Companies and Financial and Institutions.

#	Finding	References
1	Agile is about the mind-set of the whole organization	[4,14,28]
2	Training enables organizations to develop know-how and prepare better for agile transformations	[14,16]
3	Different agile methods and components should be combined	[5,45]
4	Following agile principles in full brings success	[46,19,39,38]

4.2.2 | Findings from Interviews

The interviews with FinTech and financial institutions highlighted six findings, each of which is summarized below.

A modular system architecture and microservices are prerequisites for applying agile practices. All the interviewees agreed that system architecture is a critical success factor for applying agile practices. If the system architecture is monolithic or significant parts of the system are built as a single system, it becomes difficult to apply agile practices. The interviewees emphasized the importance of having a system composed of smaller modules, where microservices constitute the software development technique's foundation. Although all indicated that their new systems have a microservice architecture, three admitted that they still have monolitic legacy components. However, two of them are in the process of completely changing or splitting up the legacy code into smaller and more modular components.

Common objectives of team members are important for managing the team. The interviews revealed that having common objectives for teams is of utmost importance. Teams should be fully responsible for their product, not only from a development perspective but also from a business perspective. One interviewee explained that, at their company, teams are responsible for products. They set their business objectives, manage customer support and operations, develop the product, and meet sales targets. Regarding clear and shared objectives, one participant said that the company uses the Objectives and Key Results (OKR) framework and tool[3]. The OKR framework[47] allows companies to define targets at different levels such as company, team, and personal levels to increase the organisation's visibility of goals. However, the participant mentioned that the focus should be on achieving goals at the team level rather than on the individual developers' productivity.

Cross-functional teams can cover the full development cycle of a product. The interviewed companies use the Scrum framework but with modifications according to their particular needs. Teams, typically, consist of a Product Owner, 4-8 Developers/Quality Analysts and 1-3 other necessary positions for the development of specific products. Thus, companies establish cross-functional teams that have the required skills for developing their product. Larger companies also have a Scrum Master, a Project Manager, or a Product Engineering Manager. Smaller companies do not have a particular Scrum Master position and, instead, transfer that role's tasks to the Product Owner or Lead Developer.

Being responsive to changes is key. The interviews showed that the agile development process should consider changes happening in developing a product. For this reason, the companies use the concept of a minimum viable product (MVP). An MVP is a product that only has the essential features required to solve the customer's problem. An MVP gives early and direct feedback about the product from the customer. Such feedback is valuable input to developing the product further. If the MVP satisfies the customer, it is developed; otherwise, the product can be changed. Since financial services are heavily regulated, the MVP is organized in two stages: alpha and beta. In the first stage (Alpha), the MVP is usually used internally with company employees. After that, in the second stage (Beta), the MVP is rolled out to specific customers. If the MVP survives the Beta stage, it is launched as a new product. The companies also stressed the importance of constant exchange of information is vital. The companies often have quarterly or monthly meetings, where the management describes the business goals and directions, and the teams give an overview of features under development. Another commonly reported practice is cleaning the product backlogs.

Automated testing accelerates the agile development process. The companies are all interested in reducing time and resources spent on quality assurance of source code. The participants working for FinTech companies merge the developer and quality assurance roles, which means that the developers write the test cases. On the other hand, the participants working for financial institutions said they have a separate quality assurance position to fulfil the regulatory requirement based on the four-eyes principle. In any case, all stressed the importance of having an automated testing process. Regression testing is the most common software testing technique used. One participant mentioned the use of Test-Driven Development (TDD) at work.

Autonomous release processes give independence to teams. All the interviewees identified the importance of having an autonomous release process in place. The automated release process gives the teams independence to launch new products or features when they deem suitable. An autonomous release processes give teams the full authority to develop and launch new products or features. Consequently, the process becomes faster because the team does not depend on other departments or resources. In case of bug fixing, the team can fix the reported bugs independently. The teams can decide whether a new feature is made available to all or part of the customers. To support their autonomous release processes, the companies have built their technological solutions or bought ready-made orchestration software from the market.

4.3 | Determining Change Proposals

The input from the current software development process at LHV, learning made from the literature review, and the insights from interviews were considered in proposing change proposals. These proposals aim at improving the development process at LHV. Proposals 1-5 concern management aspects of the development process, whereas proposal 6 targets the process itself. Finally, proposal 7 and 8 concern the system architecture and the release process. Table 4 shows a summary of the change proposals and how they are aligned to both the baseline and the findings from the literature and interviews.

Introduce an agile mindset in the whole organization. The literature suggests that being agile is not restricted to the software developers only. Instead, other involved departments must also adopt an agile mindset. In one study, it is shown that the success of implementing agile approaches

[3]OKR product site - https://weekdone.com/resources/objectives-key-results

requires the mindset of the employees to change[4]. For instance, when shifting from plan-driven methods to agile, the customer perspective is brought closer to the development teams. Therefore, the interactions between customers and development teams increase, and end-users focus shift from checking tasks to getting the right things done[14]. The interactions become essential and, as those working to deliver product features have different skills, collaboration among the team members and sense of accomplishment become important[45]. In such environments, change is natural, regular feedback is sought and given, and learning to adapt to integrate learning made with each iteration necessary[48]. Thus, the first learning from the literature review is that the organization needs to adopt an agile mindset, not only the development teams. Structures that enable closer collaboration between departments must replace rigid silos that separate business and IT. Furthermore, the management board must be less controlling and become increasingly decentralized.

Organize relevant training. When improving the development processes, training employees in agile practices has shown to be a success factor[14]. The training should consider the organization's context, including the organizational culture and the stage of agile adoption[14]. In this regard, agile coaches can bring relevant support to the organization. A sizeable Scandinavian bank reported training to be a key factor for successful agile adoption. The bank trained 80 people with a gradual formation of agile teams and using 10-week cycles that included innovation and planning iterations, development iterations, and release planning sessions[16]. Thus, the second change proposal remarks that training is vital for a successful implementation of agile practices. The learning should be about agile methods, practices, and practitioners from other companies should be involved. Furthermore, the training should take the specific characteristics of financial institutions into account.

Assemble teams for products. The interviewed companies stated that they apply a modified version of the Scrum framework. Typically, the teams consist of a product owner, 4-8 developers/quality analysts, and 1-3 positions, such as designer or data scientist required explicitly for the product. Therefore, the development teams are composed to include all the skills required to develop a product[49]. Thus, the change proposal is to assemble teams for products to have the skills required to be responsible for and manage their products' entire life cycle. In additions, teams should consist of 7 to 10 people with a mutual set of roles, as suggested by the Scrum Guide[50].

Set common business objectives for the teams. The interviews also revealed that having common objectives for team members is important. Accordingly, the teams need to be fully responsible for their products. Such a structure allows teams to set their business objectives and not only deliver tasks. To manage this, teams can use the OKR (Objectives and Key Results) framework[47]. Using such management tools to set the objectives is also used to measure team results, thereby moving the focus from the individual developer's productivity to that of the team.

Give more autonomy to the teams. In the interviews, all companies repeatedly pointed out that the agile development process should consider the changes happening in the process of developing a product. For teams to have responsiveness to change, a certain degree of autonomy is necessary. The interviews emphasized the importance of teams having authority to prioritize and adapt the product according to needs.

Review the elements of the agile development process. The increased involvement of business resources in the development process[48], requires work to be organized differently. Agile methods such as Scrum can provide structure for how to organize work. However, other agile methods such as Kanban have benefits that, if combined with other methods, can yield better results[45]. Therefore, combining practices from different methods, such as enhancing Scrum with Kanban[5] is beneficial. A large American bank successfully implemented Scrumban to combat problems they had with slow and unreliable task delivery[50]. Thus, financial institutions can adapt their agile development processes to their context by infusing relevant agile practices from other methods. The idea is to learn from the other financial institutions' best-practices and combine that with the bank's experiences when implementing agile practices.

Use modular system architecture and micro-services. All the interviewees stated that agile methodologies work better when the system's underlying architecture is modular. Having an infrastructure organized in small components and micro-services allows for concurrent development and should increase the speed in the development of the components.

Automate the release process. An autonomous release process gives teams independence and allows them to speed up the development by making the releasing phases shorter. In consequence, teams can react to bugs faster. A redesign of the whole release process is essential when the actual release process involves intense manual work and involves people from different departments.

4.4 | Implementation of Change Proposals

The analysis of the documentation provided by LHV shed light on the concrete implementation details of the change proposals. These proposals were planned as shown in Figure 3 . The follow-up interviews served to clarify and corroborate these details.

4.4.1 | Introduce Agile Mindset in the whole Organization

To address the first change proposal, LHV made the product development and delivery process aligned with agile values, aiming to improve collaboration with development teams. LHV provided employees with agile training (change proposal 2), restructuring the product organization (change proposal 3), and decentralizing the decision power (change proposal 3).

TABLE 4 Alignment among the identified baseline, the findings, and the change proposals.

Baseline	Literature Review Finding	Interview Finding	Change Proposal
Four-level organizational structure. Divisions and departments are responsible for products.	Agile is about the mindset of the whole organization [4,14,28]	Responsiveness to changes culture is the key element in agile development	Introduce agile values and culture organization-wide.
	Training enables organizations to develop know-how and prepare better for agile transformations [14,16]	Organize relevant training.	
Each team has specific roles assigned (1 Analyst, 1 Lead Software Engineer, 3-6 Software Engineers and 1-3 Quality Assurance Engineers)		Common objectives of team members important for managing and measuring team results.	Assemble concrete teams for products.
Teams work on their own products and shared products. There are 24 products with 1 PO assigned.		Cross-functional teams are having all skills to cover the full development cycle of a product.	Set common business objectives for team members.
1 SDUM leads 2 teams, 7 dev. teams and 1 R+D, 8-12 person team			Give more autonomy to the teams.
A core component with other large components		Modular system architecture and microservices are the prerequisites for applying agile practices.	Use modular system architecture and microservices
Development and Operations are two different departments		Automated testing speed up the process in agile development Autonomous release process gives the independence to teams.	Automate the release process.
The development process is hybrid- there are several elements from agile (lean, kanban boards, transparency, epics, scrum roles) but there is also a specific workflow of activities similar to waterfall.	Different agile methods and components should be combined [5,45] Following agile principles in full brings success [46,19,39,38]		Review the agile development method used with its components.

A year after the discussions, the most visible expression of introducing agile values and culture was decentralizing the decision power. Previously, the management set the priorities. However, as the bank grew, such a structure caused prioritization issues, conflict over resources, delays in decision making, and restricted the bank from scaling up its development processes [CEO]. Initially, the bank responded by increasing the number of team members, increasing the average team size from 7 to around 10. Increasing team size did not prove effective. The centralized decision structure needed to be changed because *"the company is getting larger and larger and more and more products are developed"* [HSE]. As one product area manager expressed it, *"... being more agile ... means basically delegating responsibility downward"* and that, now *"product owners are Agile"* [PAM1].

Introducing agile in the whole organization also resulted in relieving SDUMs from conducting non-value adding activities. Previously, the business side approached the SDUMs to request the development of new features. Now, the requests are made using the project management tool Jira;

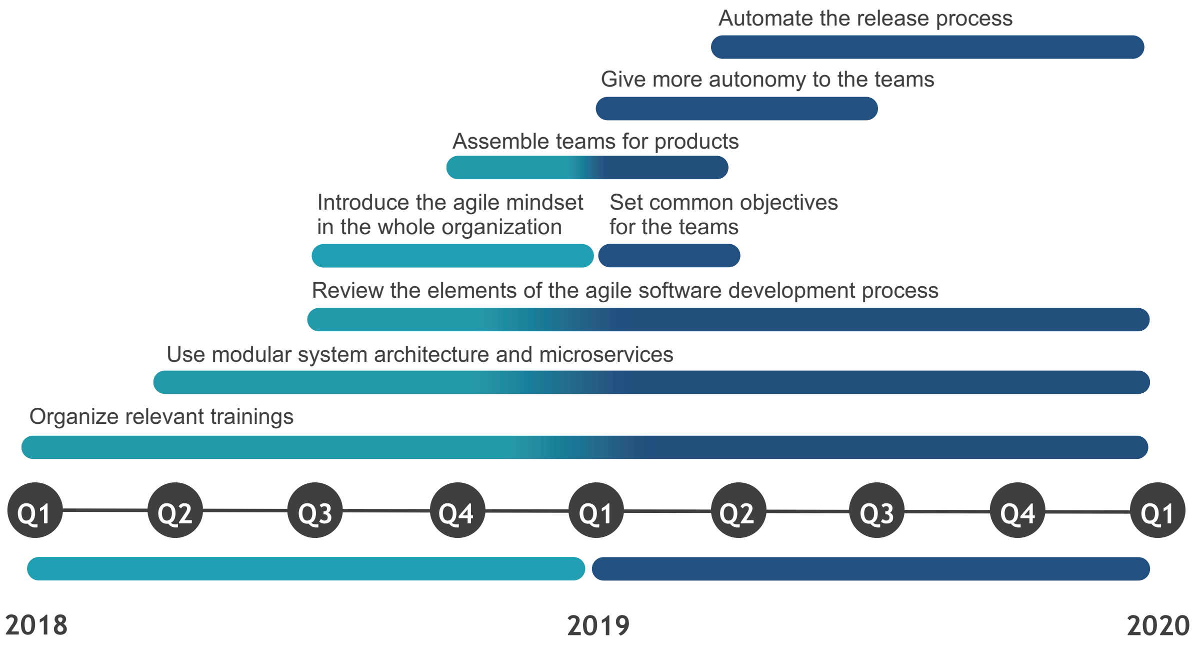

FIGURE 3 Change proposal implementation plan. The figure show how the change proposals were allocated to quarters (Q1-Q4) over the period 2018-2019.

then, Product Owners consider the request's viability and hand it over to the corresponding SDUM. In the words of one SDUM, *"I save a lot of time teaching them the business side about things that are really non-value-adding, such as how to enter a task in Jira. And then, of course, the quality is better because I don't have to get these descriptions of what they want, but rather I get the business requirements. And this is because they don't turn to me. Now they turn to the Product Owner"* [SDUM1].

4.4.2 | Organize Relevant Training

As a result of the discussions on change proposal 1, the bank initiated training activities. LHV focused on training the Product Owners, having training events with people from both development teams and the business side, and introduce a "know your bank" training [CEO].

LHV organized a Product Owners' club that regularly meets (monthly) to discuss and share experiences. External speakers are also invited to share their experiences with product ownership. For instance, one invited speaker presented how his company applied the tribe method popularized by Spotify. Also, relevant literature related to the Product Owner and agile methodologies were provided to the Product Owners [CEO].

Furthermore, new discussion spaces were created for employees, aiming to foster the understanding of agile practices and bring together both business and development perspectives. In the words of a product area manager, *"we organized a kind of workshop for the IT teams and product management teams together [involving] product managers, analysts and the lead developers [to discuss] about agile product management development practices"* [PAM1].

In addition, LHV organized a new training "know your bank". One SDUM explained that, *"if the team has finished a project, the people who have been involved in the project present it. So, everybody can get to know what is going on in the bank"* [SDUM1]. The training has affected the work of product development. One of the product area managers noted that the training *"impacted the way we conduct things. I feel better about it, we are more dynamic in reprioritizing our roadmap…" and "get to understand each other and actually started implementing different ways of doing the analysis"* [PAM1].

4.4.3 | Assemble Teams for Products

LHV had twenty-four products, which were managed by Product Owners and supported by seven development teams. Each development team developed several products, interacted with different Product Owners, and faced prioritization conflicts that had to be escalated to the management board. The organization was restructured around the products. To this end, the list of products was reviewed, rearranged, and organized into four product areas: clients and channels, investment products, financing products, and payment products [CEO]. Each product area is headed by a new role, the Product Area Manager (PAM), who is fully responsible for their products. Within each product area, there are Product Owners. One

Product Area Manager expressed: *"I am going to have five product managers in these areas. Now the product managers are effectively Product Owners. It's the same thing."* [PAM1]

The decision power has been decentralized to the PAM, who decides on the prioritization based on the product strategy and availability of development resources. As one PAM expressed it, *"we are all riding on the same limited resource, which is development teams. My product area is handling basically what we have at our disposal, i.e., two teams."* [PAM1]. The Product Owners can prioritize features for their own products but the PAM role requires coordination with the SDUM: *"The Product Area Manager is the one who has to decide or give the correct priority list."* [HSE].

The impact of this change is that, as expressed by one SDUM, *"we are more specialized in just focusing on customer segments"* and *"… moving towards having, even more, let's say clusters or in separate departments"* [SDUM2]. Now, the PAMs and SDUMs regularly discuss the roadmap that is now more dynamic and adaptive. Furthermore, as one SDUM shared, *"the clarity of our domain [the same as the business area] has been increased."* [SDUM1]. The re-organization of the product structure removed conflicts related to the prioritization of products.

4.4.4 | Set Common Business Objectives for Team Members

Following the restructuring of the product organization, LHV set quantifiable business objectives and qualitative development objectives for the teams. At the time of discussions, business objectives had been defined for the products. Therefore, these were used as quantifiable business objectives for the product organizations. For the development teams, qualitative objectives were set. However, there have been difficulties in implementing this change proposal. The product areas have defined business-oriented objectives, such as customer growth, and tech-oriented goals, such as up-time. One Product Area Manager explained that they have *"quarterly meetings with one team and monthly meetings with another team. So, with the bigger team, we did quarterly business reviews and with the smaller team monthly meeting where we look at how many new customers we have on"*, but they also *"look at various indicators that there we look at all sorts of things like up-time."* [PAM1]

The IT teams are aware of these goals as they are communicated. However, although the objectives are set for the products, the Product Owners and development teams have a different focus and, therefore, need different objectives for measuring their effectiveness. The Head of Software Engineering expressed these challenges with the following statement. *"Currently I don't see, and I don't feel that IT would have the goals of the business. We have our own goals, technical goals."* [HSE]

Furthermore, development teams might not feel ownership of objectives set by Product Owners. The Head of Software Engineering continued by saying *"if you want to make a goal for someone that he or she owns, then he or she has to have … a word in it as well, … if the goals are created somewhere ... and then they're presented to you, then you can't have it. Like it's not your goal. You don't have ownership."* [HSE]

4.4.5 | Give More Autonomy to the Teams

The insights on common objectives, teams having the skills required for managing the product, and responsiveness to change necessitates a degree of team autonomy. The reorganizing of products and appointment of Product Area Managers were prerequisites for giving the product teams autonomy. With the PAMs in place, teams were given decision power over the product, such as new product development, product enhancement, and prioritization. With the objectives set for the teams, the direction is defined, and PAMs are given authority to determine how best to achieve the objectives.

4.4.6 | Automate the Release Planning

LHV identified the administrative procedure connected to the release process as a cause of bottlenecks. As one SDUM expressed it, *"before … responsible business people had to agree to release tasks. But the problem was that they often didn't know. But when you had like 20 tasks you had to do; it was like more just a formal thing for them."* [SDUM2]. The approval process did not add value as it had become a formality and was managed via emails and chats. One SDUM clarified that *"I had to gather a sign-off from all the stakeholders"* [SDUM1]. In line with granting teams' autonomy, the sign-off procedure was, therefore, discontinued. With the sign-off waiting time removed, the release process became faster.

The removal of the sign-off was the first step. However, teams need to become independent of the operations department when managing releases. The interviews confirmed that the autonomous release process enables a higher degree of independence to the teams. With such a release process, the releases can be done when needed and reduce the time for task deployment.

LHV decided to redesign the release process to allow for an automated and continuous release process. To this end, a task force was created to improve the release process by implementing several changes. For instance, the bank started using Gradle Wrapper (tool for project management), Spring Boot (tool for monitoring the health and performance of systems), and Docker Containers (tool for enabling the release of smaller and modular parts) [CEO]. The last tool has enabled the development teams to make more minor releases daily without involving other departments. Although progress has been made, continuous delivery is not in place. As one product area manager puts it, *"when I know once the team gets to the point where we are really able to release, whenever a feature gets ready, this is my end goal. The team is not there yet, partly because they're still dependent on these legacy parts and partly because of the quite frankly development methods which have not adjusted to the new model of working."* [PAM1]. It is not only technical tools that need to be implemented, the work should also be aligned with the principles of continuous delivery because *"to have continuous delivery in place, … your team is organized in a different way as well. And I think probably the team has not caught it"* [PAM1].

The autonomy of the teams is also visible in the different approaches they use. For instance, in one of the product areas, the Product Owner meets with customers, specify the requirements, consider the financial viability of the proposed product, and considers how to *"slice and dice"* the product into smaller parts in order to fit it in the list of priorities. In another product area, Product Owners focus on providing the development teams with business requirements. Likewise, the development teams operate differently. In one development team, the SDUM assigns resources to develop product features from a pool of about 20 persons. In another development team, the resources assigned to develop a specific product or feature take more responsibility in defining requirements.

4.4.7 | Review the Elements of the Agile Development Method Used

In reviewing the elements of agile development used at LHV, four aspects emerged as particularly interesting. These are roadmaps, code review, software testing, and reporting time spent on tasks.

LHV used roadmaps as a means for the annual planning of work for development teams. The use of roadmaps began in September and finalized by December. By then, the bank had an overview of what is to be developed in the coming year. However, as changes occur, the roadmap was adapted. The roadmap was used more as a planning tool for the development teams. After reviewing the used practices, the roadmap was changed to be more business-driven and the output of the collaboration between the Product Area Manager (PAM), the Product Owners, and the Software Development Unit Manager (SDUM). As one of the SDUMs clarified, *"we had this approach for the road mapping earlier, but this year we started doing it on the business area level. If before we just wrote the development plan for the team, now we are doing this in collaboration with the Product Owners and the business managers."* [SDUM2]. As a result, the roadmap is regularly discussed and updated, i.e., *"[the bank is] more dynamic in prioritizing our roadmap."* [PAM1].

Full code reviews were reported as time-consuming, with limited value in relation to the effort required, and demotivating for the team members. Therefore, full code reviews were replaced with senior engineers performing code reviews when needed. There have also been improvements related to software testing practices. The use of automated tests increased considerably. Despite this increase, *"… the testing is actually very slow because if you want to release something, you have to do all the tests…"* [HTT].

4.4.8 | Use Modular System Architecture and Microservices

The bank's core component was developed some 20 years ago and has remained the basis for most products. The core component is relatively large and does not enable development teams to work on the same component concurrently. The bank realized that its core component has become a legacy system and decided to rewrite it. The component was divided into four smaller components (payment, investment, clients and on-boarding, and internet bank). As the head of the technical team expressed it, *"we can't make all our services very small. But yeah, we did want to have a mapping where there is one system for one team that there wouldn't be like a, a huge system where our four, four teams are developing and they have to communicate and uh, yeah, it makes things slow"* [HTT]. The first component to be rewritten was for payment products. The SDUM of this team said that *"we were the first team writing the core system from Cold Fusion to Java. There is still something left in the core system"* [SDUM1]. This team has begun to see benefits from this migration. As the same SDUM conveyed, *"the developers feel the ownership in the system. In their new system, they can do automated tests, in the old system in Cold Fusion they cannot do it. … The Cold Fusion is like spaghetti. If somebody is doing some feature, it very affects and creates many bugs"* [SDUM1]. The Head of Software Engineering also noted quality improvements and higher speed. He expressed that *"I think it's common sense that with the new system where you have quite good quality code and a well-designed system, then there is obviously much faster to develop stuff. Then all the resistance. So speed and quality improvement, and happiness among the developers"* [HSE].

The Product Area Managers are also noting improvement, in particular for the releases. The PAM for payment products expressed that *"the majority of the part of the payment has been actually rewritten and it has had a very good effect on the team's release autonomy and flexibility cause, they don't need to release some things in the night because previously they literally took a lot back down on Wednesday at 11 in the evening. Now a lot of releases happen during the day and it's just more bearable for many people and of course, the infrastructure has developed."* [PAM1].

The product owners have come to understand and respect the need to invest in system maintenance and the necessity of managing technical debt. Therefore, development teams are given time to focus on such matters. In the words of one PAM, *"every next thing, will take slightly longer if you don't clean the house every now and then. So, for this year, we have an agreement with the IT teams that they can set aside about 20% of their time to deal with technical debt as they see fit."* [PAM1].

4.5 | Summary of Impact

The first change proposal's result introduces an agile mindset in the whole organization, resulting in the bank taking a process view, including the product owners and operations in agile software development thinking. Product owners, therefore, use Jira for documenting requirements, resulting in SDUMs no longer acting as support to product owners in entering, for instance, user stories in Jira. The training (change proposal 2) enabled building a shared understanding of agile practices and improving the collaboration between product owners and development teams. Product owners have a "club" to increase their understanding of agile and apply agile practices in the context of their role as product owners.

The products were consolidated into four areas, each with its dedicated development team (change proposal 3). This change removed prioritization conflicts. In combination with giving the Product Area Managers (PAMs) authority to prioritize, this change increased the speed of decision-making and the ability for teams to adapt to changes. The planning of work for SDUM has become easier as they no longer have to plan and coordinate tasks from different product owners. The consolidation of products facilitated the setting of common business objectives for the teams (change proposal 4). This change gave the product owners clarity about the objectives and how success is to be measured. However, these goals are not easily transferable to development teams. Although both PAM and SDUM oversee the development of the same products, their work and measure of effectiveness differ. Defining goals that align with the product objectives and, at the same time, are meaningful for the development teams is still an open challenge.

The changes implemented for granting more autonomy to the teams resulted in decentralizing decision power to PAMs (change proposal 5). Previously, prioritization conflicts occurred when a development team had two or more product owners and escalated decisions on what product task to develop. This issue has been eliminated. Likewise, the concept of approval from all stakeholders (the sign-off procedure) has been eliminated as the product team is trusted with the responsibility to decide when a task is ready for release (change proposal 6). The elimination of the sign-off procedure has enabled tasks to be released faster. One of the SDUMs estimated the saving to be at least a week.

The review of the agile elements resulted in the bank improving product development's overall process (change proposal 7). The change in how roadmaps are used has improved the collaboration between the PAM and SDUMs. Furthermore, the roadmap has become dynamic, and the basis for discussion on how to best use the development teams' resources. The bank also, in reviewing the agile elements, begun working on automating tests. This has enabled increasing the degree of automated tests, which has reduced the time required for tests and improved the quality. The final change concerns rewriting the legacy system to replace the monolithic architecture with a modular one and automate the release (change proposal 8). The work has already been concluded for one of the product areas (payment products). The new structure has enabled the payment team to become more autonomous in development and deployment. Furthermore, regression tests to ensure that specific changes do not impact other functions are no longer necessary to the same extent.

In summary, the CEO of the bank expressed that *"… the effects of the improvements can be seen"* as the *"… the software process has become faster and more agile"*. The main effects, according to the CEO, lies in that *"the structure of product development management was reorganized by creating a separate PAM role [and] autonomy given to development teams by changing the release sign-off process, increasing the number of automated tests, getting rid of full reviews of new code"*. Furthermore, the CEO stated that the delivery process improved as *"a separate technical team was created and staffed and the whole delivery process improved by taking into use various modern frameworks"* and the decision to *"rewrite the core legacy system and through that process to divide it into four smaller and independent systems"*.

5 | DISCUSSION

This section discusses the research question of how LHV can improve their agile software development process. Our results indicate that the software development process can be improved by considering four aspects: introducing an agile mindset, aligning product and team structure with agile approaches, the coexistence of agile practices and methods, and alignment of the technical environment with agile development methods. Below, we discuss each of these four aspects. Finally, we discuss the main limitations of our study.

1. **Introducing an agile mindset.** Both the reviewed literature and experiences at the bank support the notion that agile is not only a method for developing software. An agile mindset can only be introduced if there is a strong and visible commitment from management. Prior studies [13,15,14,9,25] have also noted this as a critical success factor. The agile mindset needs to permeate the organization if it is to be successful. Before the changes (baseline), only the IT department followed an agile method. The other departments involved with product development, i.e., the preceding (product organization) and succeeding (deployment) steps, did not. Therefore, inefficiencies occurred. The agile mindset must permeate the whole software development process, from product or feature idea until post-deployment. Therefore, an agile mindset is not confined to the development teams. Studies on CSF, such as [13,14,15,27,9], seem to predominantly focus on the processes of developing software. However, the agile mindset must permeate all involved departments. Therefore, agile training is important, as has been stated before [13,14,15,27,7]. The training should extend beyond software development teams and, in particular, include product owners and management.

2. **Aligning product and team structure with agile approaches.** An agile mindset is a prerequisite for improvement but not sufficient on its own. The CSF of team composition [6,15,27,7] and autonomy [28,29,30,31,32,33], have been stated in previous research. Our findings add to the insights into how team structures aligned with agile approaches can be achieved. LHV restructured the product organization, the development teams, and their responsibilities to be aligned with agile methods. To this end, LHV restructured their product organization by reducing twenty-four products to four product teams. They also e a new role (Product Area Manager – PAM) and structured the development teams so each team, led by an SDUM, supports only one product area. An area product owner's role is consistent with previous research [12] that found this role to be important for scaling agile methods. To further bind the product organization with development teams, roadmaps evolved from a static plan to a dynamic tool for maintaining a shared understanding of the development plans ahead. The need to take a dynamic approach to manage the product portfolio is also consistent with previous research [12]. Furthermore, LHV introduced changes that improved team autonomy by equipping them with resources required to be responsible for the full process. This was achieved in two ways; delegation of decision mandate and setting product goals. An agile mindset requires top-level management to trust the teams by delegating decision power to the product owners. This confirms the finding that the top-down decision process impacts teams negatively [33]. Furthermore, goals are defined for the products that allow teams to work towards the same objectives. This is in alignment with research [29] emphasizing the importance of teams being able to decide what and when to develop.

3. **Coexistence of different practices and methods.** Scrum and Kanban inspired LHV's baseline agile development process. However, LHV sees value in adopting agile practices from other methods. Therefore, rather than adopting a standard agile methodology, they complemented their software process with agile practices to improve the software development process. This, i.e., combining agile practices to define a custom-made agile development process, is also confirmed in other studies [21,22,5,51,13]. For example, the core banking team can use a plan-driven approach, primarily because they do not have many changes in the application itself. Meanwhile, peripheral applications such as mobile banking, internet banking, and wealth management products might follow an agile approach. When reviewing the elements of agile development used at the bank, four practices emerged as particularly interesting: roadmaps, code review, continuous software testing, and reporting time spent on tasks. Thus, LHV adopted agile practices to their context by infusing relevant agile practices from other methods. Although their objectives must be aligned, different teams can use different methods and practices, and agile practices can coexist with more traditional approaches. Other authors have pointed this out as well [25,52].

4. **Alignment of technical environment with agile development approach.** The fourth aspect concerns technology. In improving the agile software development process, LHV implemented organizational changes. However, our case study shows that the technology must be aligned as well. Teams must develop their product independently of other teams working on other products and deploy their code when ready to scale. Furthermore, the bank recognized the need to automate repetitive tasks, such as testing. The necessity to have technical solutions supporting agile practices have been, particularly for frequent delivery in agile projects, proposed in previous studies [6,7]. To this end, LHV identified the need to adopt a modular system architecture that enables teams to work independently. Furthermore, such an architecture is required if teams are to take ownership of the release management process. Such an architecture enabled LHV to automate the release process and, thereby, teams become independent of the bi-weekly release schedule. Finally, when combined with different tools, such an architecture enabled an increase in automated testing that reduces time on repetitive tasks and improves quality. This is in alignment with previous research [6,7] that confirms the need for micro-service architecture, but in contrast with other studies that find large-scale agile projects requiring cross-team collaboration [53]. In this study [53], a large project (120 participants in a 4-year long project) was studied and found that team autonomy is difficult to maintain. While LHV has cross-team collaboration for specific areas, such as customer onboarding and regulatory issues, they do not have such large scale projects. It should, however, be noted that studies confirm the need for micro-service architecture for team autonomy [29] and for addressing challenges with maintaining and evolving legacy systems [11].

In summary, when improving agile development processes at LHV, the scope was not confined to the development teams. While improvements to agile methods within the development teams can deliver efficiency gains, taking a coherent approach delivers more value. To this end, they are fostering an agile mindset for the development teams, other departments involved in the product development process, and management. The organizational structure should be changed to support autonomous teams that have overall responsibility for their products. Also, agile practices can deliver better results when the required architecture and technology is in place. Therefore, to benefit from developing agile software development processes, a coherent approach must be taken that considers culture, organizational structures, software development processes, and technology.

5.1 | Lessons Learned

LHV has worked on scaling its agile development process by considering product owners, development teams, and technical teams. Thus, LHV approached agility from a process-oriented viewpoint. Accordingly, the changes include the work conducted before and after the software's actual development, i.e., product owners' work and after deployment. The analysis and changes implemented for the whole product development process were confidence based, i.e., based on qualitative methods and the accumulated experiences at the bank. The bank has been able to address the main weaknesses and change accordingly. However, to further improve the processes, less visible weaknesses must be addressed, i.e., eliminating wastes in the process. The next level of scaling and improvement of software development processes requires data-driven approaches. For instance, the development teams use Jira from which data can be extracted and analyzed using, for instance, process mining techniques[54] or data mining to identify, for instance, wastes[55].

Responsiveness to change also concerns communication between the product organization and development teams. Therefore, LHV has regular meetings to provide an overview of the business, the planned and developed product features, and product direction. The bank uses the roadmap as a tool for managing changes. However, when the changes are too frequent, it is problematic. One of the Software Development Unit Managers (SDUMs) expressed that the Product Area Manager (PAM) is cancelling ongoing work on product features to prioritize a new task. On the other hand, the PAM re-prioritizes tasks when existing work on specific tasks take longer than planned and, therefore, delays the start of higher prioritized tasks. It is important to find a balance between no change and too frequent changes. When begun work is discontinued, it is a waste of development efforts. To this end, the bank could use data-driven assessments of task sizes to improve the estimates' accuracy that could reduce the need for re-prioritization.

Finally, it is important to bear in mind that every good process will eventually become inefficient[56]. Therefore, the work with improving the agile processes is continuous. An example is that of ABM Amro, a bank in the Netherlands that initiated a successful agile transformation in 2016 but, after a few years, found new inefficiencies and new improvement opportunities[57]. They, therefore, decided to implement a combination of cloud and DevSecOps[58] to further accelerate product innovation and development[57].

5.2 | Limitations

Our study's goal was to examine how LHV improved their agile software development process. Therefore, it is suitable to conduct a case study[41]. However, there are inherent limitations with the case study methodology, such as construct validity, reliability, internal validity, and external validity[40].

Construct validity refers to the extent to which the used tools target what the researchers have in mind and what is investigated[40]. The proposals are based on a literature review and interviews. The interviews gave additional findings compared to the literature review. Therefore, it might be possible that the literature review was not exhaustive enough. However, by using multiple sources, this threat was reduced. Another threat to validity arises from the fact that the interviewees understood the questions differently than intended. To mitigate this threat, we had the questions reviewed by a person not involved in the study. Furthermore, this threat was reduced as the bank used the case studies' results.

Reliability refers to the degree of dependency between the results and the researcher, i.e. would the same results be produced if another researcher conducted the study. This threat was tackled as the involved persons at LHV had several meetings to discuss and verify the results without the researchers' presence (member checking)[40].

Internal validity refers to the degree to which the relationship one explores or test is not influenced by other factors[40]. The study might give a subjective overview of the product development, IT development, and delivery processes of the LHV Bank. This is partly due to LHV Bank being a regulated institution. Therefore, for security reasons, there are restrictions on how many detailed descriptions can be shared. Thus, our representation of the current software processes might not be entirely correct and complete. However, this threat to validity was reduced by having the results validated by those interviewed. This ensured that the main results are valid even if detailed descriptions were not accessible.

External validity concerns the extent to which the findings can be generalized and applied beyond the setting of the study[40]. The findings from the first part of the case study, the interviews with other financial institutions, were based on four companies. Although the companies varied in size and maturity levels, they all have software development departments in Estonia. Thus, they might not represent the software development processes of financial institutions operating in different regions. Case study methodology has an inherent limitation regarding external validity. Therefore, the results are limited to the extent they can be generalized. The results are naturally dependent on the domain experts and the context and purpose of the study. The results on improving the agile software development process are replicable at other banks, but results may vary due to the reasons above.

6 | CONCLUSION

In this paper, we examined the research question of how agile software development can be improved. In particular, we explored the improvement of such a process within the context of a financial institution that had reached a plateau and sought to become more competitive against FinTech companies. To this end, we combined a literature review, interviews, and a case study to address the research question. We found that the software development process's improvement is not restricted to the organizational unit that develops software. Instead, changes need to be made in all organizational units involved with product development. Therefore, the changes will involve human resources, organizational structures, and technology.

Those working with product development need to become increasingly aware of the agile mindset required and how to work within such a mindset. Organizational changes require restructuring how the company is organized to emulate the product catalogue, enable decentralization of product-related decisions, and create more autonomy for the teams on both the business and development side. Lastly, the improvement of the agile process requires that the technology enables the implementation of changes. To support teams that can drive product development independently, teams must deploy code when ready, automate repetitive tasks, and have a modular system architecture.

In this study, we employed qualitative methods to discover improvement opportunities, how to address them, and their perceived impact. Therefore, the results are limited by limitations commonly associated with case study methodology. However, for future work, we aim at using a data-driven assessing the impact of the implemented change proposals. To this end, we will examine the Jira logs to assess the software development processes at LHV quantitatively.

ACKNOWLEDGMENT

We would like to thank the anonymous reviewers who contributed to the paper with their comments. We would also like to thank Veronika Vlasova for helping the authors during the follow-up interview analysis. Ezequiel Scott, Fredrik Milani, and Dietmar Pfahl are partly funded by the Estonian Centre of Excellence in ICT Research (EXCITE).

References

1. Gai K, Qiu M, Sun X. A survey on FinTech. *Journal of Network and Computer Applications* 2018; 103: 262–273. doi: 10.1016/j.jnca.2017.10.011
2. Vasiljeva T, Lukanova K. Commercial Banks and Fintech Companies in the Digital Transformation: Challenges for the Future. *Journal of Business Management* 2016.
3. Lee I, Shin YJ. Fintech: Ecosystem, business models, investment decisions, and challenges. *Business Horizons* 2018; 61(1): 35–46.
4. Hajrizi E, Bytyci F. Agile software development process at financial institution in Kosovo. *IFAC-PapersOnLine* 2015; 48(24): 153–156.
5. Reddy A. *The Scrumban [r]evolution : getting the most out of Agile, Scrum, and lean Kanban*. Pearson Education . 2016.
6. Chow T, Cao DB. A survey study of critical success factors in agile software projects. *Journal of systems and software* 2008; 81(6): 961–971.
7. Aldahmash A, Gravell AM, Howard Y. A review on the critical success factors of agile software development. In: Springer. ; 2017: 504–512.
8. Romanova I, Kudinska M. Banking and fintech: A challenge or opportunity?. *Contemporary Studies in Economic and Financial Analysis* 2016; 98: 21–35. doi: 10.1108/S1569-375920160000098002
9. Dyba T. An empirical investigation of the key factors for success in software process improvement. *IEEE Transactions on Software Engineering* 2005; 31(5): 410–424. doi: 10.1109/TSE.2005.53
10. Rainer A, Hall T. Key success factors for implementing software process improvement: A maturity-based analysis. *Journal of Systems and Software* 2002; 62(2): 71–84. doi: 10.1016/S0164-1212(01)00122-4
11. Razavi AM, Ahmad R. Agile development in large and distributed environments: A systematic literature review on organizational, managerial and cultural aspects. In: IEEE. ; 2014: 216–221.
12. Putta A. Scaling Agile Software Development to Large and Globally Distributed Large-Scale Organizations. In: ICGSE '18. ACM. Association for Computing Machinery; 2018; New York, NY, USA: 141–144
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. Livermore JA. Factors that significantly impact the implementation of an agile software development methodology. *Journal of Software* 2008; 3(4): 31–36. doi: 10.4304/jsw.3.4.31-36

15. Lindvall M, Basili V, Boehm B, et al. Empirical findings in agile methods. In: Springer. ; 2002: 197–207.

16. Ivar Jacobson International . Nordea A Uniform Heartbeat with Help from Scaled Agile Framework ® and IJI. https://www.ivarjacobson.com/sites/default/files/field_iji_file/article/nordea_case_study1.pdf; 2015.

17. Birkinshaw J. What to expect from agile. *MIT Sloan Management Review* 2018; 59(2): 39–42.

18. Jacobs P, Schlatmann B, Mahadevan D. ING's agile transformation. *The McKinsey Quarterly* 2017.

19. Couzens J. Implementing an enterprise system at Suncorp using Agile development. In: IEEE. ; 2009.

20. Kilu E, Milani F, Scott E, Pfahl D. Agile software process improvement by learning from financial and fintech companies: lhv bank case study. In: Springer. ; 2019: 57–69.

21. Pino FJ, García F, Piattini M. Software process improvement in small and medium software enterprises: a systematic review. *Software Quality Journal* 2008; 16(2): 237–261.

22. Niazi M, Mishra A, Gill AQ. What Do Software Practitioners Really Think About Software Process Improvement Project Success? An Exploratory Study. *Arabian Journal for Science and Engineering* 2018; 43(12): 7719–7735. doi: 10.1007/s13369-018-3140-3

23. Boehm B, Turner R. Management challenges to implementing agile processes in traditional development organizations. *IEEE software* 2005; 22(5): 30–39.

24. Paasivaara M, Lassenius C, Heikkilä VT. Inter-team coordination in large-scale globally distributed scrum: Do Scrum-of-Scrums really work?. In: IEEE. ; 2012: 235-238

25. Laanti M, Kettunen P. SAFe adoptions in Finland: a survey research. In: Springer, Cham. ; 2019: 81–87.

26. Suryaatmaja K, Wibisono D, Ghazali A, Fitriati R. Uncovering the failure of Agile framework implementation using SSM-based action research. *Palgrave Communications* 2020; 6(1): 1–18.

27. Misra SC, Kumar V, Kumar U. Identifying some important success factors in adopting agile software development practices. *Journal of Systems and Software* 2009; 82(11): 1869–1890.

28. Miler J, Gaida P. On the agile mindset of an effective team–an industrial opinion survey. In: IEEE. ; 2019: 841–849.

29. Salameh A, Bass JM. Spotify tailoring for architectural governance. In: Springer. ; 2020: 236–244.

30. Stray V, Moe NB, Hoda R. Autonomous agile teams: challenges and future directions for research. In: ACM. ; 2018: 1–5.

31. Salameh A, Bass J. Influential factors of aligning spotify squads in mission-critical and offshore projects–a longitudinal embedded case study. In: Springer. ; 2018: 199–215.

32. Moe NB, Dingsøyr T, Dybå T. Understanding self-organizing teams in agile software development. In: IEEE. ; 2008: 76–85.

33. Moe NB, Dahl BH, Stray V, Karlsen LS, Schjødt-Osmo S. Team autonomy in large-scale agile. In: AIS Electronic Library. ; 2019: 6997–7006.

34. Keenan F. Agile process tailoring and problem analysis (APTLY). In: IEEE. ; 2004: 45–47.

35. Berkani A, Causse D, Thomas L. Triggers analysis of an agile transformation: the case of a central bank. *Procedia Computer Science* 2019; 164: 449–456.

36. Roses LK, Windmöller A, Carmo EAd. Favorability conditions in the adoption of agile method practices for Software development in a public banking. *JISTEM-Journal of Information Systems and Technology Management* 2016; 13(3): 439–458.

37. Mikalsen M, Stray V, Moe NB, Backer I. Shifting Conceptualization of Control in Agile Transformations. In: Springer. Springer; 2020: 173–181

38. Svejvig P, Nielsen ADF. The dilemma of high level planning in distributed agile software projects: an action research study in a danish bank. In: Springer. 2010 (pp. 171–182).

39. Seffernick TR. Enabling agile in a large organization our journey down the yellow brick road. In: IEEE. ; 2007: 200–206.

40. Runeson P, Höst M. Guidelines for conducting and reporting case study research in software engineering. *Empirical software engineering* 2009; 14(2): 131.

41. Yin RK. *Case study research and applications: Design and methods.* Sage publications . 2017.

42. Flyvbjerg B. Five Misunderstandings About Case-Study Research. *Qualitative Inquiry* 2006; 12(2): 219–245.

43. Kitchenham B, Pickard L, Pfleeger SL. Case studies for method and tool evaluation. *IEEE software* 1995; 12(4): 52–62.

44. Saldana J. *The coding Manual For Qualitative Researchers.* London: Sage Publications. 3rd ed. 2009.

45. Ladas C. *Scrumban And Other Essays on Kanban Systems for Lean.* Modus Cooperandi Press . 2008.

46. Jureczko M. The level of agility in testing process in a large scale financial software project. *Software engineering techniques in progress, T. Hruška, L. Madeyski, and M. Ochodek, Eds. Oficyna Wydawnicza Politechniki Wrocławskiej* 2008: 139–152.

47. Niven PR, Lamorte B. *Objectives and key results: Driving focus, alignment, and engagement with OKRs.* John Wiley & Sons . 2016.

48. Kyte A, Norton D, Wilson N. *Ten Things the CIO Needs to know About Agile Development.* Gartner . 2014.

49. Kniberg H, Skarin M. *Kanban and Scrum - making the most of both (Enterprise Software Development).* Lulu.com . 2010.

50. Sutherland J, Schwaber K. The scrum guide. *The definitive guide to scrum: The rules of the game. Scrum. org* 2013; 268.

51. Al-Baik O, Miller J. The kanban approach, between agility and leanness: a systematic review. *Empirical Software Engineering* 2015; 20(6): 1861–1897.

52. Nurdiani I, Börstler J, Fricker S, Petersen K. A preliminary checklist for capturing baseline situations in studying the impacts of Agile practices introduction. In: IEEE. ; 2018: 25–28.

53. Rolland K, Dingsoyr T, Fitzgerald B, Stol KJ. Problematizing agile in the large: alternative assumptions for large-scale agile development. In: Association for Information Systems (AIS). ; 2016: 1–21.

54. Marques R, Silva dMM, Ferreira DR. Assessing agile software development processes with process mining: A case study. In: . 1. IEEE. ; 2018: 109–118.

55. Scott E, Milani F, Pfahl D. Data Science and Empirical Software Engineering. In: Springer. 2020 (pp. 217–233).

56. Vom Brocke J, Rosemann M. *Handbook on business process management 1: Introduction, methods, and information systems.* Springer . 2014.

57. Agile's next level: ABN AMRO's hybrid cloud–DevSecOps transformation. https://www.mckinsey.com/business-functions/mckinsey-digital/our-insights/agiles-next-level-abn-amros-hybrid-cloud-devsecops-transformation#; . Last Accessed: 2021-01-21.

58. Myrbakken H, Colomo-Palacios R. DevSecOps: a multivocal literature review. In: Springer. ; 2017: 17–29.

How to cite this article: Scott E, Milani F, Kilu E, and Pfahl D, Enhancing agile software development in the banking sector—A comprehensive case study at LHV., *J Softw Evol Proc., 2021;e2363.*

APPENDIX

A INTERVIEW GUIDELINE

- Background of the company:
  - Name of the company
  - Field of activity of the company: (a) Active since; (b) Products offered; (c) Size; (d) Growth in the last 5 years

  - Please describe the organizational structure of the company: (a) Size and division of the personnel; (b) Size and division of the development personnel; (c) Size and division of the IT development personnel

- Product development process in the company:
  - Please describe the product development process in your organization
  - What kind of analysis and analysis tools are used?
  - Are you using user stories or any other techniques?
  - What makes it successful in your opinion?
  - What could be improved?
- Software development process in the company:
  - Please describe the software development process in your organization
  - What kind of agile method is used? Which components are you using from Scrum?
  - Which components are you using from Extreme Programming?
  - Which components are you using from Lean Software Development?
  - Which components are you using from Kanban?
  - What makes it successful in your opinion?
  - What could be improved?
  - How is the testing process organized?
- Software delivery process in the company
  - Please describe the architecture of your system
  - Please describe the software delivery process in your organization
  - How is organized release planning?
  - How are organized releases?
  - Are you using continuous delivery process?
  - What makes it successful in your opinion?
  - What could be improved?
- Scaling agile and growth-related issues in the company
  - How many tasks do you have in the product backlog and in development?
  - How are you measuring the speed of development?
  - Has the speed of development changed in the last years?
  - How have the processes changed with the growth of the company?
  - Are you dealing with Software Process Improvement issues regularly?
  - Are you using frameworks that scale agile? What kind of framework is used?
  - What makes it successful in your opinion? What could be improved?
  - What are the current issues and proposals discussed to improve?

B FOLLOW-UP INTERVIEW GUIDELINE

- Introduce agile values and culture organisation-wide based on the development process
  - Can you tell us a little bit about the changes as compared to the last year in regards to the process/ collaboration/ roles/
  - How do you see these changes expressed in the daily/ weekly/ monthly work?
  - What is the impact of these changes? Did it meet the expectations?
  - How were these changes implemented?
  - What is your work that is better after the changes? Why is it better? How do you know it?
  - What is still needed/ can to be improved? Why and how?
- Organize the relevant training
  - How is training organized at your team?
  - What are the types of training? (a) Their frequency; (b) Topics
  - Have you had any training during the last 2 years?
  - Has anything changed (topic or other aspects) of training at LHV in the past 2 years?
- Assemble concrete teams for products
  - Does every product have a concrete team?
  - How the changes impact on quality, speed, size?
  - Have the sizes of the team's changed in the past years?
  - What can be improved (why and how)?
  - Does every product have a concrete team?
  - How the changes impact on quality, speed, size?
  - Have the sizes of the team's changed in the past years?
  - What can be improved (why and how)?
- Set common business objectives for the team members
  - Can you tell us about the common business objectives?
  - What kind of business objectives, who set them, how are they set?
  - Do they change often, how do they change, why do they change?
  - How are the objects visible or expressed in the process?
  - What impact does it have?
  - Was it good?
  - What can be improved?
  - Autonomy to the teams
  - In which way teams are more autonomous?
  - How does it impact?
  - Has it anyhow increased/decreased
  - Is it good for the team's daily work?
  - How was it implemented?
- Review agile development methods

  - What are the changes you see have been made in the past years to the development process? (e.g., code review, testing, deployment, communication)
  - How are the changes visible in the process? (e.g., artifacts, new statuses, new ways)
  - What is the impact of the changes in the process?
  - Is it good or not?
  - How the changes were implemented?

- Automate Release Process
  - Can you tell us about the release process (e.g., your role in the release process, the process, artifacts, etc.) and changes made for improvement during the last year?
  - What was the impact of release process automation?
  - Was it good?
  - How was it implemented?

- Modular system
  - How has the change to the modular system been done?
  - Which impact did it have?
  - Was it good?
  - How was it implemented?

- General
  - What is the main challenge you see now?
  - What would you like to change and in which way?
  - Anything else you would like to mention?