Master's thesis

NTNU
Norwegian University of Science and Technology

Andreas Saltom Rikheim
Eskil Helgesen Schjølberg

The use of Agile practices in regulatory evoked development

A case study of a large-scale inter-company project in the financial industry

Master's thesis in Computer Science
Supervisor: Marius Mikalsen
June 2023

NTNU
Norwegian University of
Science and Technology

Andreas Saltom Rikheim
Eskil Helgesen Schjølberg

The use of Agile practices in regulatory evoked development

A case study of a large-scale inter-company project in the financial industry

Master's thesis in Computer Science
Supervisor: Marius Mikalsen
June 2023

Norwegian University of Science and Technology

NTNU
Norwegian University of
Science and Technology

Abstract

Agile software development methods have gained significant popularity and recognition in the software industry due to their iterative and collaborative approach. These methods prioritize flexibility, adaptability, and customer collaboration to deliver high-quality software products for their customer. By embracing the principles of agility, development teams can respond to changing requirements, improve communication, and foster innovation throughout the software development lifecycle.

Previous literature has shown that agile software development methods can be applied under several different circumstances and way beyond the small, co-located single team which it was initially made for. However, the literature on agile software development methods in regulated environments is sparse but shows some fundamental challenges of applying agile software development methods in regulated environments.

Regulations in the financial sector play a crucial role in ensuring the stability, integrity, and transparency of financial markets and their institutions. Governments and regulatory bodies implement laws and regulations to safeguard investor interests, prevent financial fraud, and promote fair and efficient practices. Compliance is not only a legal requirement but also a critical aspect of maintaining trust and confidence in the financial system among the population. Financial institutions face the challenge of navigating complex regulatory landscapes while keeping their systems compliant.

This study presents a historical explanatory case study of a large inter-company software development project in the financial industry. The project was evoked by a regulatory change, and the thesis focuses on the challenges of being agile in a regulated environment and how the challenges were mitigated. In this study, we identified several mechanisms associated with either mitigating or heightening the challenge of being agile in a regulated evoked project. We also identified agile practices used in the project by using a predefined framework. In the analysis, the mechanisms identified were categorized into challenge categories found in the existing literature and new challenge categories discovered in this study. The findings lead to six propositions on how to be agile in a regulated environment, which has implications for theory and practice.

Acknowledgements

We would like to thank our supervisor Marius Mikalsen for great discussions during our research, the feedback from him has been of great value to us.

The project would not have been possible without all the informants who took time off from their busy days to help us and give insights into the project and their experiences. We are very grateful to the case organization who gave us access to their offices and internal documents.

We have appreciated the opportunity to investigate a real-life complex development project. This gave us insight into how the field of software development is practiced outside academia.

Trondheim, June 2023

Andreas Saltom Rikheim and Eskil Helgesen Schjølberg

Table of Contents

1 Introduction

After the introduction of agile software development methods, it quickly became the preferred method among practitioners. However, there are still some areas where practitioners struggle to apply agile software methods, including regulated environments. Since the agile manifesto was introduced in 2001, agile software development methods have been one of the main research areas in software engineering (Hoda et al., 2018). Agile methods have changed the software development paradigm to focus more on change tolerance, active end-user involvement, and evolutionary delivery (Dingsøyr et al., 2012). Today, a variety of industries adopt agile methods, and based on the 16th State of Agile report [1] the two top industries using agile methods are Technology and Financial services (digital.ai, 2022). Furthermore, half of the respondents in the same survey said they used a hybrid approach, combining agile methods with other development methods like waterfall and iterative. This indicates that even among agile communities, traditional software development methods' elements are still valuable.

In an attempt to find a theoretical core for agile development, Baham and Hirschheim (2021) suggest using a set of agile concepts to examine agility in different contexts. Several papers already study the use of agile practices in projects initiated for strategic reasons, such as increasing customer experience and revenue. However, several industries, especially the financial sector, face ever-increasing regulatory pressure, where the prime goal of some software projects is to satisfy these governmental demands. This leads to another type of software development project, where the traditional customer focus is swapped with absolute demands decided externally. However, how agile software development methods are applied in such a context is unclear.

The financial sector is one of the most regulated industries, and regulations are still increasing. In 2022 there were 94 new regulations in Norway, 41 of which impacted the financial sector[2]. The trend of heavy regulations in finance began after the financial crisis in 2007-2008. Prior to the financial crisis, there was a deregulation of the financial sector back to the 1980s. After the increase in regulations, the cost of compliance has increased for financial institutions (Ceps, 2019)[3]. Steel-

[1] The information is based on a survey with 3220 respondents recruited among members in agile interest organizations and from participants on agile conferences.

[2] From searches on https://www.regjeringen.no/no/dokument/lover$_{r}egler/id$438754/

[3] The report included 11 EU member states and 50 interviews supported the quantitative cost

Eye's annual compliance health check report [4] in financial services confirms the challenges the financial sectors face, as "keeping abreast with regulatory change" is the biggest challenge when trying to meet their regulatory obligation. (SteelEye, 2022)

While evidence shows the benefits of the agile methods in general, the evidence for agile methods in regulated environments is sparse.(Cawley et al., 2010) (Fitzgerald et al., 2013) Agile methods are well suited to handle changes in the project with its iterative processes and focus on incremental design. These characteristics fit well in an environment where new regulations lead to constant changes and demands for software development. On the other hand, regulations often demand a more central structure, with stricter rules, hard deadlines, and burden of proof (Cawley et al., 2010), which clashes with the principles of agile methods. (Fitzgerald et al., 2013)

To better understand how agile methods work in a regulatory-evoked project, we investigate a case of a large software development project in the financial sector in Norway. The project was not a standard large-scale project, as it was one where the whole industry chose to collaborate. The project was evoked by a legislative amendment passed by the Norwegian parliament in 2019 demanding all pension providers change the structuring and management of pension capital certificates (Lovdata, 2019). The change required the sector to make it possible for pension customers to transfer their pension capital certificates between the providers. This requirement leads to a common transfer hub between the companies and support for such transfers in the individual providers' systems. To achieve this, a cross-company project was initiated, which was managed by an independent consultancy. More than ten pension providers were involved in the project, in addition to an industry organization, a management consultancy, and a software solution company.

To analyze this project, we have performed qualitative analysis on interviews of participants. We have utilized a predefined framework (Baham and Hirschheim, 2021). As there doesn't seem to be an agreed-on definition of agile methods in the literature, we would use a set of proposed core concepts defining Agile (Baham and Hirschheim, 2021) as it is one of the newest articles, which gathers the existing definitions of agile and combines them into their own. By using these

estimates.

[4]The report has surveyed 170 senior compliance decision-makers in financial services across the UK and US

definitions, we are able to analyze how agile practices are used in a large-scale inter-company project evoked by regulatory demands and how their use is affected in such a context. As we know, there are challenges to using agile methods in both large-scale and regulated environments; it is helpful to use a framework to determine how they are being agile. In addition, we have used a deductive and inductive approach to categorize the retrieved data into categories of challenges based on the literature on agile software development methods in regulated environments and patterns discovered in our analysis. Our research goal was to investigate how *agile methods can be adapted in large-scale inter-company projects evoked by regulatory demands*.

Based on this research goal, we posed the following research questions:

RQ1: What agile practices fit a large-scale inter-company regulatory-evoked project?

RQ2: What challenges are faced when trying to use agile practices in a large inter-company regulatory-evoked project?

RQ3: How can challenges faced when trying to use agile practices in a large inter-company regulatory-evoked project be mitigated?

This study seeks to make the following contributions to research:

1. Provide a rich empirical description of a large inter-company agile software development project regulatory demands did evoke.
2. Propositions that can form a basis for a new theory on agile software development in a regulatory context

The first research question seeks to identify agile practices used in a regulatory project that fits the task well. The two remaining research questions aim to give insight into regulatory-specific elements contradicting agile practices and the mitigations used to stay agile. This can be useful for practitioners but also add to the literature concrete challenges and respective solutions in such projects. The first contribution, a rich empirical description, gives the reader an insight into a real-life project from the private sector and how practitioners use agile methods. The study adds an in-depth report of one case instance to the research, which, together with other case studies of the topic, can form a more generalizable theory (Flyvbjerg, 2006). The contribution can also inspire other practitioners to organize

such a project. The second contribution summarizes the study's key findings in the form of concrete propositions. Software engineering studies pay little attention to theory building, and few build on existing theory (Stol et al., 2016). By stating novel propositions, we seek to contribute to creating a theory to understand agile software development evoked by regulations.

This thesis is structured as an engaged scholarship publication (Mathiassen, 2017). Section 2 presents the background with a review of the extant literature on agile software development and regulations in finance. We will present the history of agile, proposed definitions, and examples of other case studies describing the use of agile today. We then describe the purpose of financial regulations, how it affects financial institutions, and the general progress of such regulatory demands. The section is completed by combining the two topics and reviewing the literature on agile software development in regulatory environments. This background material will construct and articulate the opportunity to contribute. Section 3 describes the method and framing of the study. This includes the data collection and the qualitative analysis performed. Section 4 presents the results of the data analysis and establishes an empirical foundation to make the contributions. This section will overview the challenges and benefits of using agile practices in the chosen context and propose solutions to the stated challenges. In section 5 we discuss the results, explaining and arguing for the contribution. The propositions will also be presented in this section. In section 6, we conclude, present some limitations, and suggest some further research. .

2 Background and related work

2.1 Agile software development

In this section, we first present the history of Agile and some proposed definitions in recent research. We then move on to large-scale agile, related research, and its characteristics.

2.1.1 Agile practises

The agile software development methods originate from a discussion amongst software developers who wanted to address the challenges with the existing methods; this led to the creation of the Agile Manifesto (Beck et al., 2001). The Agile Manifesto consisted of twelve principles based on several existing practices. The following four values sum up the principles:

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

These values stood in stark contrast to the waterfall methods that were heavily used at the time. Since the release of the Agile Manifesto, agile methods gained increased popularity amongst practitioners and have become a heavily discussed topic among researchers(Baham and Hirschheim, 2021) (Surendra and Nazir, 2019) (West et al., 2010). The widespread use of agile software development methods has led to several methodologies and frameworks, with Scrum as the most popular. As the interest in and use of agile software development methods quickly grew in the software industry, the scientific community has tried to keep up. However, as the practice did not come from the scientific community but from the industry; there was no theoretical core, only a set of values and practices. In the beginning, research on agile software was mainly descriptive, describing what agile software development methods were and reporting on some cases (Beck, 1999) (Schwaber, 2004), or prescriptive, explaining how agile software development methods should be implemented (Sharp and Robinson, 2004)

(Merisalo-Rantanen et al., 2005). This stage of the research on agile software development methods was later criticized for the lack of empirical evidence (Dybå and Dingsøyr, 2008), and theoretical evidence (Dingsøyr et al., 2012), which led to a call for an increase of research with theoretical perspectives (Abrahamsson et al., 2009). The call has led to several articles in recent years (Baham and Hirschheim, 2021) (Dingsøyr et al., 2022).

Even with the increased number of contributions, the research of agile software development methods still faces a big challenge, the lack of a theoretical core. In 2005 Strode wrote:

"Without a definition of agility, any author can state that their method is agile. The agile manifesto (Beck et al., 2001)is not a suitable set of criteria for defining agility. To be useful as an accurate indicator of agility, all of the manifesto principles would need to be met by any method that called itself agile. However, each of the commonly acknowledged agile methods fulfills some subset, but not all of the 12 principles of the manifesto (Visconti and Cook,2004), making it...a poor tool with to (measure agility)"(Strode, 2005)[5]

Eighteen years later and there is still not a clear theoretical core in the literature that is agreed on. There have been several attempts to define agile software development. Baham and Hirschheim (Baham and Hirschheim, 2021) gathered several different definitions and summarized them in their definition.

"A software development team's ability to anticipate, create, learn from, and respond to changes in user requirements through a process of continual readiness."

Together with the definition proposed by Baham and Hirschheim (2021), they provide four core concepts that capture the sense of agility and are found in almost all agile software development methods.

- Incremental design and iterative development
- Inspect and adapt cycles
- Working cooperatively/collaboratively/in close communication
- Continuous customer involvement

They argue that to be an agile software development methodology; you must include at least one practice that supports each concept. This helps the practitioners

[5]The quote is from page 24

who rarely follow a strict model but take inspiration from different methods and tailor their method to fit their situation and company. With these concepts, you can identify if a team works agile, even if they don't follow an existing method, by comparing their practices to the four concepts. In agile frameworks, the common approach to implement *Incremental design and iterative development* and *inspect and adapt cycles* is to break the product into smaller pieces and iterate over them. These iterations will give a better understanding of the smaller tasks and help the team get continuous feedback to improve the product. Both of these concepts are relevant for the technical and business side, as the development team will discuss with the customer to find the most pressing issues and then prioritize the tasks based on the discussion. After a given time, they will show their work, have a new discussion and get the chance to adjust, and discover errors early in the development. The inspect and adapt cycles can span different periods depending on the practice, from a few minutes during pair programming to weeks with demos and retrospectives. The concepts *Working cooperatively/collaboratively/in close communication* and *Continuous customer involvement* focus on teamwork and communication with the customer. Most agile methods have practices that support these in the iterative processes.

2.1.2 Large-scale agile

When agile software development methodologies were first introduced, it was meant for small, co-located teams. (Boehm and Turner, 2005)Later studies have proved that this isn't necessarily the only situation suited for agile methods. There have been several studies on agile on large-scale, which demonstrate how the agile software development methods can be applied successfully (Dingsøyr et al., 2022) (Paasivaara et al., 2018) (Dikert et al., 2016). The challenge of distributed teams av also been addressed by the literature, with several research papers proving the successful use of agile software development methods in distributed teams (Yap, 2005) (Stotts et al., 2003) (Boland, 2004)

After the success of agile software development methods, it was natural that organizations would want to implement this at a large scale. However, there was skepticism among both practitioners at researchers, and seven issues associated with scaling agile were presented after a workshop among 35 professionals (Reifer et al., 2003). Among these was implementing the methods without any adjustments, as this would be an easy solution for companies but would face sig-

nificant challenges when implemented. As a response to these challenges, hybrid approaches which combine traditional project management methods with agile methods among the teams (Batra et al., 2010) (Badampudi et al., 2013) (Dingsøyr et al., 2018) were introduced. These approaches gained significant traction among practitioners, which led to increased attention from the research community (Bick et al., 2018). The hybrid approach was prevalent as it was easier to introduce to more traditional companies, as they are familiar with the project management processes. However, there were still some challenges with agile at large scale (Bick et al., 2018) (Badampudi et al., 2013), and in the last years, we have seen projects where the traditional project management methods are replaced by processes more in line with the agile manifesto. A number of the methods have been established, such as Scrum-at-scale, SAFe, LeSS, Nexus, Agile portfolio management, Disciplined agile delivery, the Kanban Method, and the Spotify model (Dingsøyr et al., 2019). Challenges and success factors in implementing some of these methods have been researched (Edison et al., 2021).

One of the challenges that appear is Inter-team coordination. Agile software development methods remove traditional coordination mechanisms and replace them with lighter practices such as through face-to-face communication Strode et al. (2012). Table 1 from (Dingsøyr et al., 2022) shows the differences between the approaches to coordination in traditional and agile methods.

Traditional	Agile
Forward planning	Synchronization through activities and artefacts
Explicit documentation	Face-to-face communication (proximity)
Roles	Autonomous teams
Pre-defined processes	Boundary spanning across teams

Coordinating this way works well with small teams (Pries-Heje and Pries-Heje, 2011), but as the projects get bigger, the coordination gets harder, especially with dependencies among teams (Edison et al., 2021) (Bick et al., 2018). Large-scale agile was defined by (Dikert et al., 2016) to involve 50 or more people or at least six teams. The term 'very large-scale agile' describes projects with ten or more teams (Dingsøyr et al., 2014). In projects with this amount of teams, the challenges to scaling agile software development methods become apparent, especially inter-team coordination. Inter-team coordination has received a lot of attention from researchers (Bass, 2019) (Dingsøyr and Moe, 2013) (Bjørnson et al., 2018) (Dingsøyr et al., 2022), the knowledge about it has increased, and Dingsøyr et al. (2022) proposed five prepositions as a foundation for a theory which could

help future projects.

2.2 Regulations in finance

This section will give some insight into the development of financial regulations, their purpose, and their effect on financial institutions. We will start by reviewing the regulatory demands following the financial crisis in different jurisdictions. We then move on to present the regulations responding to the digitalization of the financial sector. The section is concluded by describing the state of financial regulations in Norway today. This section presents the drivers of the development projects evoked by regulatory environments. These drivers explain the nature of such tasks.

2.2.1 Post-crisis regulations

The financial sector has been facing a large number of regulatory demands in the past decades. After the financial crisis in 2007-2008, there was a consensus among policymakers that financial institutions had to be more strictly regulated. In the US, this became a reality with the passing of the Dodd-Frank Act in 2010. Following this law, there was a rapid increase in regulatory restrictions as reported by The Economist (2017b). However, these restrictions have later met some setbacks as administrations change (Economist, 2017a).

The urge for regulations in finance has also been present in the European Union following the financial crisis (Quaglia, 2013). As a result, financial regulation and supervision reform in the EU were initiated in 2009, specifically aimed at avoiding future crises by ensuring financial stability. This reform introduced new rules or amended existing ones in banking, securities markets, and financial supervision (Quaglia, 2013). These included directives and regulations concerning deposit guarantees, capital requirements, the "too big to fail" problem, fund management, and market infrastructure. The regulations have a binding legal force in all member states; while the directives state some results that must be achieved, each state can choose how to transpose them into laws. These post-crisis regulations show some of the increasing regulatory pressure financial institutions have faced in the last decades.

2.2.2 Digital finance regulations

Following the digital transformation of the financial sector, new types of regulation have emerged. According to Guellec and Paunov (2017) *"digital transformation is the digitization of previously analog machine and service operations, organizational tasks, and managerial processes"*. Arner et al. (2015) describes the period 2008-onwards as the third stage in the digital transformation of finance. This stage is characterized by using innovative technology to create new services. As technology is the primary driver in this stage, we see tech companies and new start-ups emerging with innovative concepts disrupting traditional financial institutions. These tech companies and start-ups are not under the same regulations as the large institutions, giving them a favorable regulatory arbitrage (Barroso and Laborda, 2022). However, when the traditional financial industry adopts these new technologies, existing regulations do not cover these innovative services, and new regulations are adopted continuously.

Policymakers also have an increasing focus on protecting consumers by ensuring healthy competition and innovation through regulation. As with the post-crisis regulations, many digital finance regulations are incorporated into EU law. One such legislation is the PSD2 directive passed in 2015 (EU, 2015). An example of a provision in this directive is to *"promote innovative mobile and internet payment services"*. This resulted in a law that demanded banks create open APIs on customer data so other actors could use this information to develop new services. Such laws and others related to digital finance are expected to increase in the years ahead, and Barroso and Laborda (2022) question their effect on financial actors.

This research on digital financial regulations shows how digital finance pushes legislators to increase regulatory pressure and how this results in demand for systems developed by financial institutions. Unfortunately, regulatory arbitrage also causes the institutions to spend resources on developing these systems rather than developing competitive and innovative products. As this issue is only expected to increase, financial institutions should use effective development methodologies to minimize this disadvantage compared to start-ups and tech companies.

2.2.3 Financial regulations in Norway

In Norway, financial institutions are facing a continuous change in regulatory demands. The financial sector can be seen as one of the most well-regulated

compared to other industries. Of the 94 regulations adopted in 2022, 41 were connected to the Ministry of Finance (Regjeringen, 2022). While not all of these regulations impact financial institutions directly, many aim to ensure financial stability and adapt to the rapid evolution of digital finance, as described in the preceding sections. The overall goal of financial regulations is, according to Armour et al. (2016) *"investor protection, consumer protection, financial stability, market efficiency, competition, the prevention of financial crime, and fairness"*. The nature of many of these topics is complex and debated mostly among professionals; the specifics of financial regulations are therefore often seen as a "black box" to the public (Abbott, 1995; Preda, 2009; Baker, 2014). As a result, these acts are rarely opposed politically and are passed regularly at a steady pace. This view was supported in this study by the Director of the industrial policy stating: *"It [financial regulations] is less party politics ... they are relatively complex, and if an agreement exists between the employee and employer organizations, there is a less political discussion about it. It is a complex matter which the politicians would like to stay out of. Some become politics, but it is mostly regulations decided by the ministry."*

As mentioned in the two preceding sections, many financial regulations are incorporated into EU law. Norway is not an EU member. It is, however, part of the single market through the EEA agreement and is thereby obligated to adopt market-related legislation from Brussels. There are, in total, 58 financial directives and regulations legislated by the EU, many of which also affect Norway (EU, 2023). Norway has, however, no formal way of influencing EU legislation, which is a pressing issue according to a recent study (Juuse et al., 2019). The same study also finds that Norway tends to be stricter than EU minimum in several areas and states: *"A similar trend [stricter requirements] concerns consumer protection, both in banking and financial markets (funds and securities industries), which has been elevated to another level 'despite' EU regulations. The 'common good' and consumer protection areas have been important in Norway and continue being subject to national law."* (Juuse et al., 2019).

This data and research show how Norwegian financial institutions face increasing regulatory pressure. However, it also shows that these are complex matters requiring expertise. This might make the development of systems complying with these demands more challenging than other development projects. Choosing an appropriate development method to address these challenges can therefore be seen as important.

2.3 Agile in regulated environments

The digitalization of the financial sector has increased the industry's need for software development drastically. This development is often related to creating or improving customer products or enhancing internal processes. As described in previous sections, financial institutions adopt agile methods for these development projects. However, as the regulatory pressure increases, many projects are also initiated to satisfy regulatory demand. When asked how many of the new regulatory demands required software development, the Head of Industrial policy in our study stated *"I would say all ... this is a digital business, there is nothing we do that doesn't require system support with changes and adaptions when our frame conditions are modified."*. Even tho some research has been performed on agile development in regulated environments; it is still unclear how agile methods fit such regulatory-evoked projects.

As mentioned in 2.1.2, agile methods were initially for small, co-located teams. Another constraint tied to agile methods was regulated environments (Abrahamsson et al., 2009). While the literature has proved the two first constraints wrong, there is little evidence of the adaption of agile methods in regulated environments (Cawley et al., 2010). When you look at the original values from the agile manifesto:

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

You can see that the last part of every statement fits with typical requirements for a regulated project, whereas agile software development methods value the first part of each statement rather than the last.

2.3.1 Involvement of specialists

When developing software in regulated environments, it is crucial to have expert input at some point in the process. Initially, someone has to convert the new regulations into understandable tasks for the developers and subsequently control

that the implementation complies with the legal formulations. Literature shows that this can be a major challenge (Fitzgerald et al., 2013) (Poth et al., 2020) (Stirbu and Mikkonen, 2020) (Hanssen et al., 2019) (Jönsson et al., 2012). LoD-PQR (Poth et al., 2020) is a general framework intended for use in regulated environments. This method has a sub-process where experts identify the compliance aspects of the product. After identifying the compliance aspects, the regulatory demands get mapped to different handover levels. Several compliance aspects get grouped in each handover level to form an LoD(level of done) topic. This way, the compliance aspects derived by the experts lay the foundation for developing the software. This leads to a lot more forward planning than the typical agile software process and lacks the continuous involvement of specialists, which would have been natural as iterative processes is a key part of agile methods. This method is very general, and the regulative processes vary from industry to industry, so implementing this would need further adjustments.

Another approach to performing compliance checks has been to use tools and workflows to create automated checks. The frequent use of DevOps in software development has enabled the integration of compliance checks, streamlining the automating test processes. For example, a study from 2020 created CompliancePal (Stirbu and Mikkonen, 2020), a GitHub app that automatically searches every commit based on compliance criteria and reports if the commit doesn't pass the test so a compliance officer can review the problem. The challenge of basing your compliance on an automation tool is the risk of not discovering a violation of regulation, as this can lead to significant consequences, both financially and also the reputation of the company.

A study from 2019 describes a method that was used to handle the PSD2 regulatory in a financial institution (Hanssen et al., 2019). This approach consists of three main steps, where the first step is an expert group identifying all the critical parts of the system. The next step is to address all the subsystems affected in the critical part. The last step is to estimate the possible impact of different scenarios and place them in categories such as ”realistic”, ”possible” or ”extreme”. In this specific case, it was used to estimate architectural debt, but the method was presented to be generally applicable to other consequences of regulations. This could be a good starting point, but it does not work as a full framework. It will also be costly in the long run if you make such an analysis for each new regulation, which you would have to do to stay agile.

Another industry case study (Fitzgerald et al., 2013) suggests integrating quality checks in a regulated Scrum process. The presented framework is called R-Scrum, where compliance control is performed at multiple stages in the development process. At the end of each sprint, the Quality Assurance group audits the output, which isn't a part of the development process. The quality assurance team identifies issues related to compliance with regulations and adds them to the backlog to be handled in a later sprint. This process is known as continuous compliance, staying true to the iterative processes of agile software development methods. The last sprint before release is a hardening sprint, where all the issues from earlier sprints and the final product with all the integrated parts must pass the quality assurance audit to be released. A challenge for this framework is the independent quality assurance team, which might lead to coordination problems as they lack detailed knowledge of the development process. Studies on Agile development show that teams based on specialized competence often struggle with communication across teams. Agile software development methods strive for autonomous cross-functional teams, which seems to be left out in this framework.

2.3.2 Documentation

Agile software development methods strive to bring the amount of documentation as close to zero as possible and instead spend time on product delivery and improvement. This goal is a challenge when working in a regulated environment where you have to be able to document that you comply with the regulations. Paulk says that *"Agile methods may be inappropriate for life-critical and essential moneys projects to the degree they oversimplify design and lack documentation"* (Paul, 2011). Other articles (McCaffery et al., 2016) (Krasonikolakis et al., 2020) seconds this thought and point to the challenges with documentation in agile environments.

A paper in the financial domain explains the need for documentation, showing examples in the context of failure *" Indeed, the heavy regulatory framework means managers get scrutinized when they report losses. They are expected to develop numerous reports and analyses, justify their decision-making and corresponding choices and present any mitigating actions"* (Krasonikolakis et al., 2020). The statement shows the need for financial institutions to trace problems down to their root, which you need the documentation for. An article (McCaffery et al., 2016) in the medical domain finds that the senior management only will approve of a change from waterfall to agile

software development methods if they can guarantee documentation at the same level. The same article reports a well-functioning product that passed all the tests but was still disapproved due to a lack of documentation and traceability.

Documentation is a real problem for agile methods. Still, two papers mentioned in section 2.3.1, CompliancePal and R-Scrum, give different approaches to the documentation challenge in regulated environments. CompliancePal links newly added software components to the documentation in a C4 (Brown, 2018) format to ensure efficient documentation. The software architect sets up the initial architecture. If a new component isn't added to the documentation, an issue will be created, and the compliance officer will be notified. This is a good way to make documentation more efficient, but this does not cover all the required documentation you need in a regulated project.

R-Scrum makes documentation a part of the process, which is done in multiple stages of the development process (Fitzgerald et al., 2013). First, they create a design template with rules the developers must follow. Then, the team has its user documentation they are responsible for. If the developers change the design during a sprint, the changes will be peer-reviewed and sent to the quality control group. The quality control team will then control the proposed changes and, if approved, update the design. The quality control team must also document all the test results acquired at the end of each sprint. R-Scrum also has ”living traceability,” meaning the developers must link all code to specific tasks. This ensures you can document how every job is solved in the code. A ”dev check” is in place to ensure that there isn't any code in the repository with no linked tasks. The documentation solution here leads to more work for developers on process formalities which the agile software development methods want to eliminate. It also requires an external team to control the changes, which is another step away from autonomous teams.

It is important to remember that the differences in the regulated industries might affect how strict the documentation needs to be. For example, Mccaffery (McCaffery et al., 2016) looks at the medical industry, where products often have a risk-tolerance of zero, while sectors such as finance can accept small amounts of risks. In finance, you can sometimes put up the cost of providing documentation against the cost of not complying with regulations. This will also affect how methods can be applied and how they work in different regulated environments.

2.3.3 Incremental design and architecture

Incremental design is a central concept in agile software development methods. The developers can receive feedback on their products and continuously improve and adjust the software. According to this principle, you deploy your code as fast as possible so the customers receive new features and can give you feedback so you can correct problems early. Continuous deployment and continuous integration are important parts of agile software development methods. Still, they face some challenges in regulated environments, as the software often must be done entirely and have full compliance before you can deploy it.

Regulatory sandboxes have been one approach to solve the issue of being unable to test the product early and receive feedback. There are different kinds of regulatory sandboxes, with their purposes (WorldBank, 2020) depending on the domain and the environment. However, one of the main challenges with sandboxes is that it is hard to correctly replicate and predicate how the market behaves. Therefore, with regulatory sandboxes, it is important to remember that it isn't a direct substitute for market testing but can be used as a good indicator.

From the earlier cases mentioned, R-Scrum (Fitzgerald et al., 2013) is the only one which addresses this challenge in their methods. They design each sprint around a specific feature, so they have something new which is completely done to deliver at the end of each sprint. This made it possible for the marketing team to demonstrate and market the new features so that the developers could receive feedback. The quality assurance team made this possible with their checks, as described in 2.3.1.

The incremental design and architecture also positively impact development in regulated environments. With new regulations being applied at a high pace, forcing the financial institutions to keep up, the incremental design focusing on module-based architecture allows the banks to adapt faster. This gives the financial institutions increased IT flexibility which helps decompress the regulatory pressure (Reitz et al., 2018). This article also found that micro-services help mitigate the pressure, as this also strengthens IT flexibility. Micro-services make it easy to modify and add new features as the code does not directly impact other parts of the system. According to (Scott et al., 2021), micro-service architecture is one of the keys to having autonomous teams and implementing agile software development methods in regulated environments.

2.3.4 Scaling agile in regulated environments

All of the formerly mentioned papers on agile in regulated environments study agile software development in methods used in single teams, so what challenges are added when you try to scale them? According to (Steghöfer et al., 2019), there are three main areas of challenges, living traceability, continuous compliance, and organizational flexibility. Living traceability goes beyond the traceability between requirements and test cases. It requires the developers to continuously create, update and remove trace links between different components, requirements, and test cases as they change during development. Continuous compliance is constantly producing the necessary regulatory documentation to ensure you can document your compliance at any stage of the development process. This starkly contrasts the established way, where the regulatory documentation is produced at the end of a project. The last challenge is organizational flexibility, divided into three subcategories: safe ecosystem, change management, and way of working. A safe ecosystem is particular for the automotive company in the case study and not relevant to this study. In the established practice, decisions tied to regulations must be made by some form of centralized team, which goes against agile principles. Change management is about moving the decisions to the teams, making them more autonomous and able to move forward at their own pace, and not being held back by bureaucratic processes. The way of working focuses on the challenges of coordination and reuse. It suggests using modularisation and architectural isolation of functionality as a way of helping with reuse and avoiding excessive coordination by removing dependencies.

Some of the challenges of being agile described in Steghöfer et al. (2019) are close related to those presented in section 2.1.2 on large-scale agile. Change management is similar to autonomous teams as a way to avoid centralized decision-making slowing down the development process. The way of working is similar to inter-team coordination in avoiding dependencies. The concepts living traceability and continuous compliance are found in the literature on documentation in section 2.3.2. Both concepts are part of the R-Scrum method introduced by Fitzgerald et al. (2013).

2.4 Chosen theory to analyze the case

The majority of existing literature on agile development in regulated environments focuses on small projects in single development teams and projects working to comply with existing regulations. One article was found that studied agile scaled in regulatory environments, see section 2.3.4. On the other hand, no reports found focused on agile projects evoked by regulatory demands. Following this, we will study a large-scale agile project evoked by regulatory demands. The study will use categories identified in the presented literature, and a theoretical framework proposed by Baham and Hirschheim (2021).

2.4.1 Categories identified in literature

The chosen categories from the literature are the ones described in 2.3.1, 2.3.2, and 2.3.3, namely:

- Involvement of specialists
- The need for documentation
- Flexible architecture

These categories capture special needs identified in regulatory environments that in some way affect agile practices. The chosen categories will be used to categorize elements identified in the case project and their impact on the agility in the project. These elements will later be referred to as mechanisms. Additional categories were discovered in this study and will be presented in section 3.4 on data analysis.

2.4.2 The theoretical framework used

To identify agile practices in this project a theoretical framework was used. This framework is based on the four core concepts suggested by Baham and Hirschheim (2021) described in section 2.1.1. As there are known challenges of using agile methods in both large-scale and regulatory environments, the practices found in the case project are analyzed against the core concepts of agile to determine to

ASD techniques	Incremental design and iterative development	Inspect and adapt cycles	Work collaboratively	Continuous customer involvement
Planning game (PM)	X	X	X	X
Small releases (PM/SE)		X		
Pair programming (SE)		X	X	
Test-driven (SE)		X		
Coding standards (SE)		X	X	
...				

Table 1: Framework for discovering agile practices

what degree they are agile. An example of the use of this framework with some core agile practices is presented in table 1

By using the core concepts framework, we can identify agile practices used in a large-scale agile project evoked by regulatory demands. By comparing these identified practices with the mechanisms found in the categorization described in the previous section, we also discover if some elements both qualify as practices and mechanisms. Following this, we will be able to discuss how the identified practices and mechanisms influenced the project regarding being agile. This is our primary method for answering the research question and making the contributions stated in section 1.

3 Method

3.1 Choice of study type and philosophical paradigm

We have designed a historical explanatory case study to investigate the research question -*How can agile methods be adapted in projects evoked by regulatory demands?*. Such a study is characterized by investigating past events, explaining why things happened as they did, focusing on the outcomes, and comparing identified factors to the existing literature(Oates et al., 2022). As mentioned by Fitzgerald et al. (2013), little research exists on this topic, and as shown in the background section, there is a lack of case studies with this focus. This study's wish is to go more in-depth into the phenomenon, which is more possible with a case study than, for example, a survey (Oates et al., 2022).

The study is historical, as the project under investigation happened in the past. We chose to study a natural setting since the development of practices in agile software development is heavily influenced by practitioners facing unpredictable situations. As this field lacks a proper theoretical core (Baham and Hirschheim, 2021), practitioners' view of the topic is of high interest. We have chosen explanatory as the type of study as we, in addition to describing a phenomenon, also compare it to the literature and identify influence factors (Oates et al., 2022). Finally, we regard the selected scenario as a typical instance. This premise is founded on the understanding that we are examining an industry characterized by intense competition, where significant practice variations are improbable. In addition, as stated by Flyvbjerg (2006), *"One can often generalize on the basis of a single case ... the force of example is underestimated"*. The types of generalization relevant to this study are "rich insight" and "implications" as defined in Oates et al. (2022). The study will give a rich insight as we will provide a new understanding of a situation, and we will imply what might happen in other similar cases and present some recommendations.

The interpretive paradigm has been selected as the approach for this study. This paradigm is deemed suitable as the research question does not involve confirming or refuting a hypothesis but instead seeks to uncover, investigate, and elucidate the management of regulations in agile software development within the finance industry. Moreover, the study shares many features of Interpretivism as outlined in Oates et al. (2022), such as researchers' reflexivity, the study of people

in natural social settings, qualitative data analysis, and multiple interpretations.

3.2 Case study design

This case study aims to increase the knowledge of the use of agile software development in the context of regulations in a large inter-company project.

The case organization was chosen by convenience since they were interested in a collaboration and passed the criteria of being a financial institution. As a specific case project to investigate, the organization had one suggestion that met our criteria. We were also given access to some ongoing development processes, providing supplementing insight into software development in regulated environments in general. The chosen case is a large-scale software development project, which was instructed to be agile by the project initiator. This instruction made the project pass the criteria of being agile. As the project was also initiated by an enacted law and a series of associated regulations, our criteria for case selection were met.

The project was distributed across multiple organizations, all with individual projects dependent on a central communication hub. With several organizations, coordination and inter-team dependencies were seen as significant factors. The project being large-scale was not a criterion. Still, this characteristic can be seen as a consequence of the regulatory nature of the project and is therefore weighted in the study.

The unit of analysis in this study is the agile practices adopted by the participants in the project and the inter-team coordination between them. We will also analyze the use of other development practices since using these instead of agile methods is of interest. The study will also focus on observing practices identified in the literature on agile software development in regulated environments to see if practitioners in this study use any of them.

The collaboration with the case organization was initiated in December 2022. At the first meeting, we presented the findings from a literature review we had authored that semester. The review was on *Agile development in regulated environments in finance*. As the organization was in the crosssection of being highly regulated and had incorporated the agile mindset, a case study on this topic was of interest to them. We were then granted access to the office space of the case

organization in January 2023 and initiated the study the same month.

3.3 Data collection

Our strategy for data collection was relatively clear from the start. Due to the nature of the study, as described in the two previous sections, interviews were a natural choice. Document collection and observations were chosen as supplementing data as these were made available. Since the case organization was located in several cities, interviews were done both digitally and physically. In the following sections, we will present how the work location affected the study, how the interviews were conducted, the documents collected, and how the data was stored. A summary of the data sources used in the study can be found in table 2.

Data source	**Description**	**Number**
Interviews	Interviews were performed semi-structurally, and both physically and digitally depending on the informants location. Most of the interviews consisted of one informant, but two interviews consisted of two informants.	15 interviews.
Documents	We were given access to existing documents from the project consisting of meeting presentations, concept presentations, process documentation, and API documentation.	14 documents
Observations	We spent time in the office of the case organization where we observed the daily work and took field notes. We spent on average 6 hours there each day, and participated in small talk, lunches and worked side by side with the developers.	28 days

Table 2: Data collected in this study

3.3.1 Physical presence

In January 2023, we were given access cards to the case organization offices in Trondheim. We were invited to sit among software and business developers in an open-plan office. Consequently, we became part of the work environment, ate lunch, and participated in small talk with the employees. By doing this, we gained insight into the culture and habits of the workplace. Some of the people present had been participants in the main large-scale project; others contributed to the supplementing study of agile development in regulated environments. Our presence at the company offices was fixed two times a week from January through May, with some exceptions.

This physical presence had some characteristics of an ethnographic study(Oates et al., 2022). Although the organizational culture was not the focus of this study, we used several elements from the ethnographic research method to capture contextual data. We wrote field notes on informal discussions about relevant matters such as agile development and regulatory demands, other topics like technology stack used, roles in this industry, or just general information about the business. These notes were mainly of the substance type, but some analytical notes were also produced. As both researchers have some experience as developers, we could easily associate and empathize with the employees, a central element of the holistic school in ethnography research. In this regard, we also had to be aware of the reflexivity present, as we had preconceptions, were of different ages, and had different backgrounds and education. These elements were considered when analyzing the field notes, in addition to awareness of how the study objects might perceive us.

3.3.2 Interviews

As mentioned briefly in the introduction to this section, interviewing was the preferred strategy for collecting data in this study. There are three main reasons for this. Firstly, we wanted to obtain detailed information from participants about the phenomenon. This follows from the first contribution in this study *"Give a rich empirical description..."*. Secondly, our questions were occasionally complex, needing explanation or follow-up. They were also open-ended and might need to be customized differently for different people. This follows because people had different backgrounds, and some were unknown of agile practices. Thirdly, we

wanted to explore some emotions together with the answers. Feelings associated with certain experiences can give valuable insight and is challenging to capture through pre-defined questionnaires. This is especially relevant to contributions 2 and 3 about success factors and challenges, as the opinions on these matters often are expressed with emotions like joy or frustration.

The recruitment of interview objects was done in different arenas. More than a third of the interviews were people from the office where we were physically present. These were recruited during lunch breaks, coffee breaks, or just "across the table". Following the informal invitation, an email would be sent, and a time and place would be scheduled in Microsoft Teams. This was usually done a week prior to the interview. All these interviews were conducted physically in a meeting room at the case organization's office. In this group, there were both interviews directly connected to the large-scale case, and some focused on the existing development routines in relation to being regulated. The other recruitment arena was digital and consisted of offices of the case organization located elsewhere and other organizations involved in the large-scale case. In this arena, the primary strategy was snowballing. We started with a subject with a central role in the large-scale project, and from there on, we recruited based on mentioned roles and names. Through this strategy, we could reach relevant people from several organizations and different levels of the organizational hierarchy. All these interviews were scheduled in teams, usually a week prior, and conducted as a Teams video meeting. In total, 15 interviews were conducted with 17 participants. The participant's roles span from software and business developers to people working with politics; all the roles covered in the study are presented in table 3.

Before each interview, there was some preparation that had to be done. The interview objects were sent an information letter together with the mail invitation so that they could be prepared. Our preparation consisted of information gaining and discussions on a researcher's roles and identity. We gathered all relevant online information on the upcoming interview object to avoid spending valuable interview time on known facts. This also helped us customize the interview to focus more on topics known to the participant and especially to prepare ourselves by acquiring basic knowledge on this topic. Considering a researcher's role and identity, we discussed how the objects would perceive us and what gaps existed between us and the informants.

The interviews were mainly two-to-one, with two researchers and one interview

object. On two occasions, two-to-two interviews were carried out. This was mainly because the interview subjects had worked closely on the project at focus, and it was suggested by themselves. We carefully considered the two subjects' work relations in these situations, so it would not affect the interview. All the interviews were semi-structured. We had a list of themes and a set of core questions we wanted to cover, see appendix A, but we were willing to change the order, exclude questions or add follow-ups. These themes and core questions will be discussed further in the analysis section. The interviews were always initiated with an explanation of the research study and some mutual introduction. All the interviews were scheduled for one hour and lasted between 30 minutes and one hour as recommended by (Oates et al., 2022). We usually conducted 1-2 interviews a day and never exceeded three. All interview participants' roles and companies are in table 3.

Code	**Role**	**Company**
BD1	Business developer	Minor actor
BA1	Business analyst	Minor actor
SD1	Software developer	Minor actor
SD2	Software developer	Minor actor
HSD1	Head of strategy and development	Large actor
SD3	Software developer	Minor actor
DIP1	Director of industrial policy	Large actor
DM1	Delivery manager	Large actor
BA2	Business analyst	Minor actor
DM2	Delivery manager	Large actor
PM1	Project manager	The consultancy
PM2	Project manager	The consultancy
PM3	Project manager	Minor actor
PO1	Product owner	The software Co
LD1	Lead developer	The software Co
PM4	Project manager	Large actor
SD4	Software developer	Large actor

Table 3: The participant's roles in the project

3.3.3 Document collection

In addition to conducting interviews, we use documents as a data source. The documents related to the historical large-scale project were documents already existing prior to our research project. They include meeting presentations, concept presentations, process documentation, and API documentation. The analysis section will present the complete view of the documents collected and analyzed. These type of documents was obtained by asking interview objects about them after the interview. Often the interview object would mention something about a document in the interview, and then we would follow up by asking to get them sent by email.

This study also retrieved other types of document data, such as public records and published reports. The use of public records has been related to understanding the scope of regulations. For example, we have used governmental sites to retrieve data on the number of regulations and acts passed that affect finance. In addition, we used the Norwegian Parliament pages to study the process of passing a law. We also used Lovdata (2019) to find information about the specific law that evoked the large-scale case. Finally, published reports are mainly used to get some quantitative insight into international regulatory pressure and agile practices.

3.3.4 Data management

In this project, we had to comply with several demands concerning data management and participant consent. To be allowed to start the study, we had to apply SIKT (SIKT, 2023) on what data we would collect, how we would collect the data, why we needed the data, and how we would manage it. We also had to attach our interview guide, see appendix A and create an information letter, see appendix B for the participants. All this is due to the strict regulations concerning the GDPR legislation.

Before each interview, the participant had to consent to the information letter. In the physical interviews, the declaration of consent was printed out and signed by the participants. In the digital interviews, the participants wrote an email reply consenting to the attached information letter. The letter contained information about the study purpose, the participant's rights, and how the participant's data is managed and used.

All the interviews were recorded and transcribed. The interviews done by teams were video-recorded and automatically transcribed. This transcription was, however, not good enough, so they were also manually transcribed. The physical discussions were audio recorded and then manually transcribed. All the transcriptions were anonymized, both concerning individuals and institutions. All recordings and transcriptions were stored in our private Microsoft Team's channel as this complies with the security demands for such data. The same goes for all the documents collected in this study.

3.4 Data analysis

In this study, we have performed qualitative data analysis. This follows from the qualitative data collected through interviews and documents. The choice of these retrieval methods has been argued for in the previous sections. Qualitative data is non-numerical data where analysis is all about abstracting the verbal, visual, or aural themes or patterns from the data (Oates et al., 2022).

The analysis began with a complete read-through of all the interviews to get a general impression. We then defined some criteria for selecting units of text for further analysis. These criteria were defined in two categories:

- Segments of data relevant for describing case context
    1. Segments describing actors and roles involved in projects
    2. Segments describing projects origin
    3. Segments describing regulatory politics
- Segments of data relevant to the RQs
    1. Segments describing agile practices
    2. Segments describing documentation
    3. Segments describing the involvement of specialists
    4. Segments describing the architecture

The research process involved multiple read-throughs conducted individually by each researcher, followed by a thorough comparison of our respective findings. In cases where discrepancies arose, we engaged in constructive discussions to

evaluate and determine whether the divergent findings should be included or not. The collaborative nature of our work, with two researchers involved, proved advantageous as it allowed us to leverage each other's ideas, engage in insightful discussions on complex topics, and ultimately gain a comprehensive and holistic understanding of the case, which led to a more robust and nuanced analysis

3.4.1 Discovering categories

An inductive approach was used to discover two additional challenge categories: industry collaboration and absolute demands. This finalizes the criteria for RQ-relevant segments:

5. Segments describing the industry collaboration
6. Segments describing absolute demands

"Industry collaboration" appeared as a category due to the inter-company dependencies discovered in the read-through. "Absolute demands" became a category for challenges that appeared in the project due to the absolute nature of governmental decisions. The selection of the other criteria was explained in section 2.4. Our analysis shows that the flexible architecture isn't affected by the regulatory environment, as this is the preferred architecture for other reasons. Therefore we have decided to remove this category.

3.4.2 Identifying Agile practices

A new read-through was performed using the first of the RQ-relevant criteria for selecting units describing agile practices. We then used a deductive approach to analyze these segments. The deductive approach was based on the Agile definitions proposed by Baham and Hirschheim (2021). The framework for identifying agile practices proposed by Baham and Hirschheim (2021) was also presented in section 2.4. The definitions used in this framework are reproduced in table 4.

When analyzing the data with this framework, we identified 70 text units describing some agile practices that qualified to be part of one or more of the four core concepts. These units were sentences drawn from the interviews and are overlapping, so this does not represent 70 unique practices. Although we have not conducted a statistical analysis, it is worth mentioning that 35 of these units were retrieved from material describing the large-scale case. When analyzing the units of text against the core concepts, only the definitions in table 4 were used; the resulting selection is therefore also partly based on the individual interpretations of these. An extract of the analysis can be seen in figure 1

Based on this analysis, the results will be presented in a table in section 4.1.5

Text	Incremental design and iterative development	Inspect and adapt cycles	Working cooperatively/ collaboratively/in close communication	Continuous customer involvement
*"Ellers sånn prosessen rundt selve utformelsen av lovverket, forskriften og sånt er det gode prosesser på, at det **mulig å komme med innspill**. Så var det vel Finans Norge som kom med innspill på veiene av aktørene."*			1	1
*"Altså vi har to **ukentlig standups og litt retrospektiver**"*		1	1	
*"Men vi prøvde vel oss på Scrum, vi hadde sånne **sprinter** i danica fordi vi hadde ett lite team."*	1	1	1	
*"Her vil man helst ha det ganske lite og definert så man kan **release oftere**,"*	1	1		
*"Vi snakket sammen og så **definerte vi oppgaver i Jira**. Så delegert man ut til de som skal gjøre oppgavene."*			1	
"Det lå nok en røff beskrivelse i tilbudsforespørselen. Dette var jo høsten 2019 at man valgte leverandør også tror jeg at man utover høsten jobbet med en detaljert kravspekk og **brukerhistorier** og så jobbet man vel utover 2020 med mer detaljer kravspekk til selve løsningen."	1			
*"Men tenker jo mer på det generelt da, at man lager en arkitektur som kan endres, litt sånn **komponent basert**, kapsler inn logiske biter så man slipper å endre ett helt*	1			

Figure 1: An extract of the Excel document used in the analysis

where we have gathered the statements into different practices and marked which concepts it is a part of. The table will have the same setup as table 1.

Core concepts	Facets of agility in the literature
Incremental design and iterative development	Anticipating change by working iteratively – in short, delivery cycles – and thereby reducing the scope of the product to small increments to create opportunities for inspection; Creating change through incremental software design in response to change from what has been learned
Inspect and adapt cycles	Anticipating change by instituting ceremonies for inspecting and adapting (i.e., learning from and creating change in response to discovered changes) the product increment (e.g., simplifying – "just enough" – design, testing software frequently) and the development process (e.g., updating work statuses, reevaluating team processes, reprioritizing requirements)
Work collaboratively /cooperatively/in close communication	Anticipating change through recognizing and predicting changes in one's environment; Creating change as a team by working together to respond to change from what has been learned collectively
Continuous customer involvement	In addition to the cell above, centralizing user requirements changes by working together with the customer to collectively identify and respond to change early through close customer involvement

Table 4: Definitions used when discovering agile practices from Baham and Hirschheim (2021)

3.4.3 Identifying mechanisms

Based on the criteria for the challenge categories(2-6 in the list), except Architecture, a selection was performed on all the interview data collected. The resulting units of texts were transferred into individual sticky notes on a *miro board* (Miro, 2023). A 4-set Venn diagram was created on this miro board, one set for each identified category. All the sets intersected, so there were 15 different combinations of overlapping sets. The sticky notes with units of text were then placed in the Venn diagram on the right intersection. This process required interpretations and discussions. We also marked all the units describing a mitigation or heightening of

the challenge, each with its colored dot. Figure 2 presents the Venn diagram used.

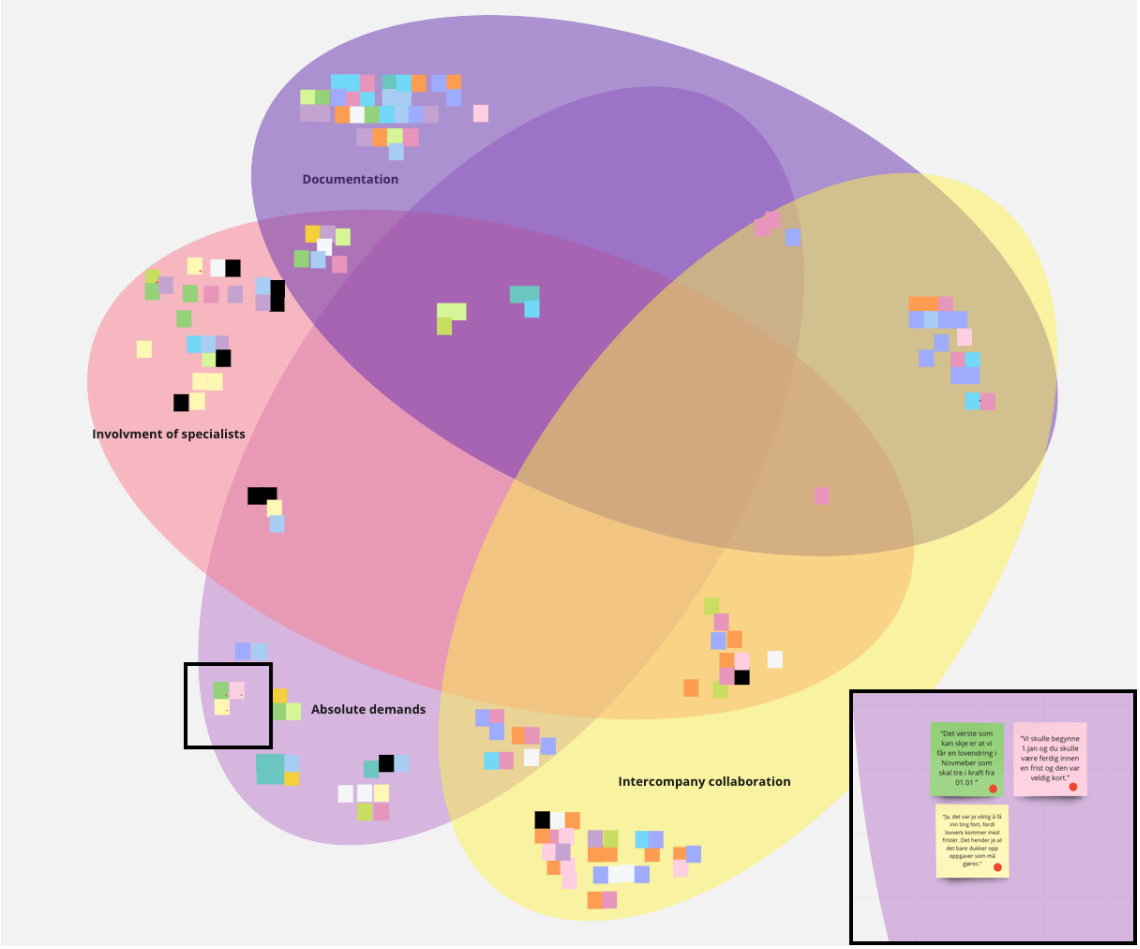


Figure 2: 4-set Venn diagram used to sort sticky notes with units selected text.

The results of this analysis will be presented in tables in section 4 with the mechanism identified, the category/categories it fits under, and a short description. The table will have the same format as sample table 5

3.5 Research Quality

In this study, we have focused on several aspects of research quality. The quality factors generalizability and reflexibility have already been accounted for in section 3.1 and 3.3.1, respectively. This section will explain other quality factors, such as reliability and additional validity elements. Finally, we will account for the research ethics considered in this study.

Category	Mechanism	Description
Documentation	Living traceability	The orders are connected to specific code in Jira in the development.
Absolute demand	IT audit	IT audit were performed twice a year where documentation was controlled.
Involvement of specialists, Absolute demands	Legal intervention	Product-oriented and compliance-oriented legal specialists involved in planning and control of development.
Industry collaboration	5 guiding principles	Principles used by the project management to justify denials of change requests.

Table 5: Data collected in this study

3.5.1 Reliability

Unlike in positivistic research, we do not believe that there exists some absolute truth on how to perform development evoked by regulatory demands. Consequently, we think that the research results, to some degree, are affected by the researchers conducting it. This is based on the conception that this field is part of social construction driven mostly by practitioners, and the results obtained in the study might be short-lived and changing. That being said, this research field rapidly gains evidence to pose theories and likely benchmarks. The fact that this whole field of research originates from a mindset makes it hard to claim the repeatability of research results.

3.5.2 Validity

In addition to generalizability and inflexibility, which are already accounted for, other methods have been used to ensure validity. Verbatim quotations have been used to assure readers that points made in this study are based on things actually said, not only interpretations of what was said. Some triangulation of data was used to ensure that the interview's interpretations were in line with the message intended by the informant. Allt We corresponded via email and had to clarify

via meetings with several participants at the end of the research period. As this was an interpretive study, there is no absolute benchmark to test against, as we believe there is no single objective reality in this matter.

3.5.3 Ethics

When performing this study, there were several aspects related to ethics that we had to consider. This does altogether concern the participants of the study. By participants, we refer to informants, people connected to informants, us as researchers, and the academic community. First, we had to consider the data protection rights of our informants. This was ensured by applying for storing such data, en ensure that we stored it according to the laws, a process described in section 3.3.4 on data management.

The informants also have several other rights as participants. In short, they have the right not to participate, the right to withdraw, the right to give informed consent, the right to anonymity, and the right to confidentiality. All these rights are mentioned in the document of consent described in section 3.3.4 and found in the appendix B. All the informants participated voluntarily and were not under any pressure or demand from their superiors. Each participant was sent the verbatim quotations used in the thesis collected from their respective interviews and allowed to withdraw or edit their statements. Several informants chose to edit their verbatim quotations to make them more precise. As the consent document was also an information letter, informed consent was given. All informants were anonymized by only using their role and company affiliation when stating the quotations. We also complied with the right to confidentiality by keeping all interviews, recordings, and documents securely stored privately.

4 Results

We will first describe the background and the goal of the EPK(Egen pensjonskonto) project. This presentation will be based on an analysis of internal and external documents on the project and on descriptions gathered in the interviews as described in 3.3. Then, in the remaining part of section 4.1, we will give a rich description of the EPK project from the planning phase through the development phase and to the test and deployment phase. By this, we make our first contribution - *Provide a rich empirical description of a large inter-company agile software development project regulatory demands did evoke*. At the end of each section, we will present a table of the mechanisms identified in the respective phase. Following this, we will present the identified agile practices. Finally, section 4.2 will take a closer look at the different mechanisms in all the categories and describe their effect on the project.

4.1 The EPK project

4.1.1 The project background

In Norway, your pension is gathered in PKBs(Pensjons kapital bevis). Prior to the EPK project, the PKBs were placed with a pension provider based on the company you worked for. In 2019 the parliament passed a law that allowed employees to decide their pension provider, regardless of the company chosen by their employer. It was also decided that PKBs would be gathered and kept with one pension provider. If you want to keep your PKBs separated, you would have to reserve yourself against gathering them.

The law was passed to increase the competition between the pension providers and had wide support among the different parties in the parliament. The Norwegian pension providers manage 20-25 billion Euros, and this law was estimated to save 80 million Euros for the consumers by reducing the fees paid to the pension providers. Furthermore, the law also strengthens the consumers' rights by giving them the freedom of choice of their pension provider.

The EPK project lasted from August 2019 to February 2021. Even though the law was passed early in 2019, many of the regulations weren't ready before the fall of 2020 as the department had to work out the regulations based on the law passed

by the parliament. We have studied one of the main initiators among the pension providers, a minor pension provider, and the development of their internal solution, as well as the common technical solution known as PKR(Pensjonskonto registeret) and the project management of the whole project. The EPK project consisted of 13 pension providers and their internal development teams, an external technical solution provider with one development team, a project management team from the consultancy, and several inter-company teams with different specialists led by Finans Norge. The involvement of all actors is shown in figure 3 and will be explained in the following subsections.

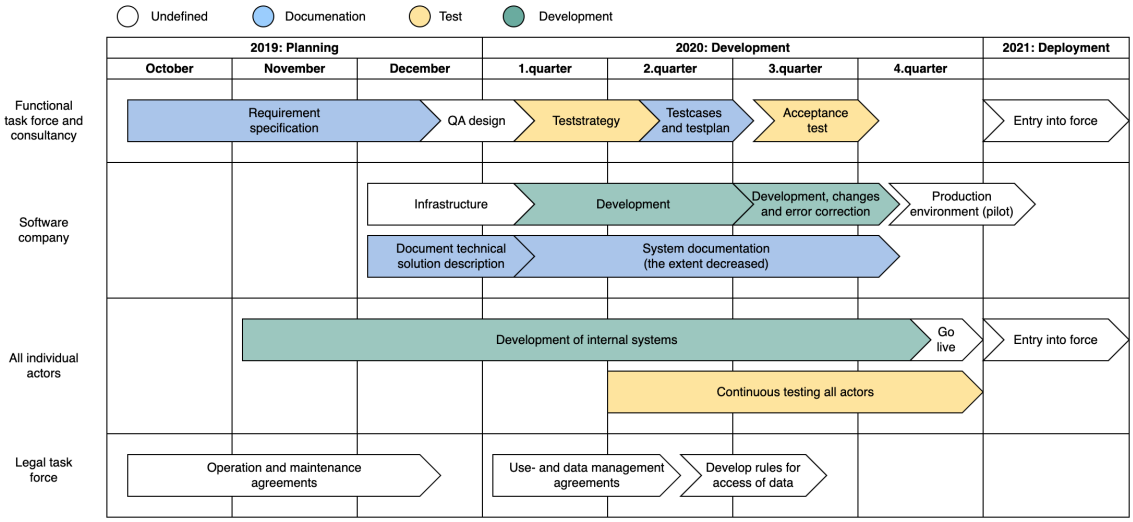


Figure 3: The timeline shows the involvement of two of the task forces mentioned, the consultancy, the software company, and the pension actors. In addition to working closely with the functional task force, as shown here, the consultancy was responsible for tying all the streams together. This timeline is simplified to fit the focus of this study and is created based on several exhaustive flowcharts collected in the study.

4.1.2 The planning phase

As the new law was proposed, the pension providers came with input and feedback during the consultant rounds. After some internal discussions, the large pension provider quickly realized that this would significantly impact the company, both financially and practically.

"I took a closer look and tried to assess how this would impact us, both from the business perspective, but also how we practically had to make adjustments with our technical sys-

tems. So in the beginning I did that, and realized that this would be the biggest change we had experienced in a long time" - HSD1

After some time, the large pension provider decided they wouldn't be able to solve this independently, and a cross-functional task force agreed that they had to reach out to the largest companies in the industry. The other companies agreed this would require a shared infrastructure, a hub, for the whole industry. As this would require a completely new system for the entire industry, they had to get a third party to develop the solution and lead the process. The industry established a task force consisting of representatives from different actors in the industry who together made a requirement specification which they sent out to different consultant companies. The group making the requirement specification consisted of technical architects, functional people who described the different cases, and legal persons who focused on the regulatory aspect. Finally, a consultancy was chosen for the project management, with the software company as the technical solutions provider, and the project could start. Figure 2 shows how the project was organized in the planning phase. Different task forces worked within their expert domains and reported to the project management, which had an overview of the complete picture and then reported to the advisory committee.

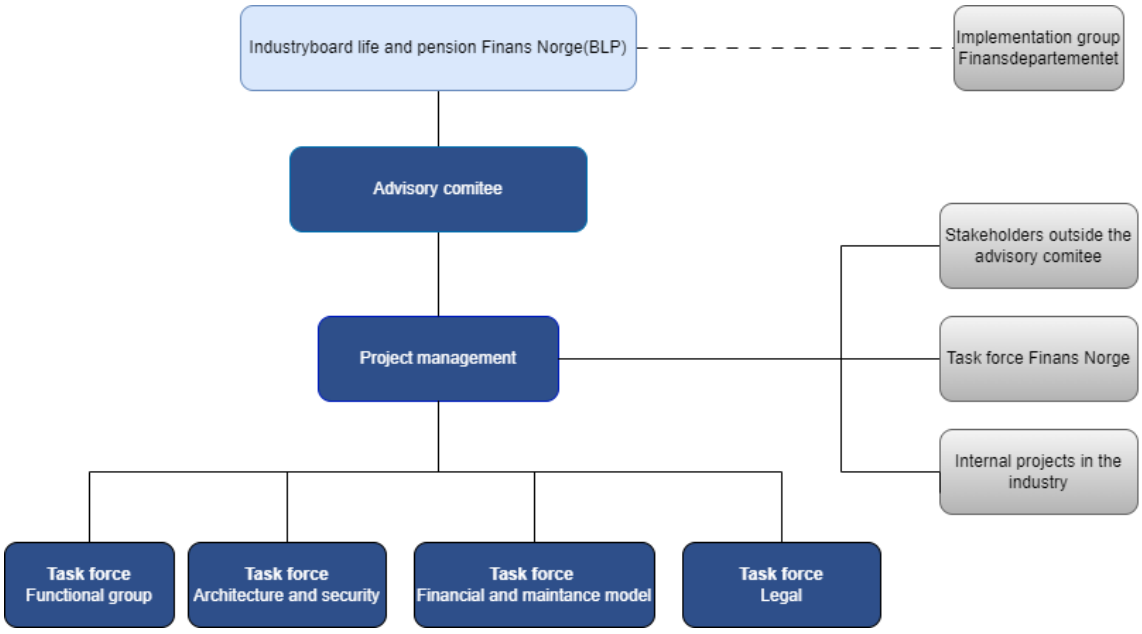


Figure 4: Project structure based on documents from the case organization and simplified to fit the study.

The first four months, September to December, were spent creating user stories, workflow charts showing how the processes would connect, how the pension

providers would interact with PKR, and further elaborating on the requirement specification. This was done by the consultancy, industry representatives, and the software company with 2-4 workshops each week during the fall. During this process, there were serious discussions, and the consultancy had to ensure that the project became the objectively best and not fit for one company. As the requirement specifications were done, all the companies had to agree to the specifications, and then the consultancy received the mandate to deliver based on this document.

The new regulation was not an advantage from a revenue standpoint, meaning that some income would disappear. Consequently, some intense discussions followed when we developed the functional requirements. The companies also had to adjust or develop their IT systems, and the size of the required investment was dependent on the functional requirement and different from company to company" -PM1

In the request for proposal, the software company had explicitly said that they would be working agile, with an iterative process, but this was not how it started. As the requirement specification document got finished, the software company was asked to deliver a description of their solution as an answer to the required specifications before they could start developing. The software company was skeptical about this and told them that they could make an overall description, but it wouldn't be detailed, and it would have room for change as this would be required to make a good solution. It was clear that the beginning of the process had signs of waterfall methods which is supported by statements from the informants.

"We had two months to answer the requirement specification with a description of a solution, so we went from an offer that says we are gonna work agile, and two weeks later they say no, we will have a requirement specification with a description of the solution, We just went for a waterfall method right from the start, and we tried to fight that all the way" -PO1

"The requirement specification followed a more traditional waterfall method. However, when the software company started the development we switched to a hybrid model in order to accommodate for changes along the way" - PM1

Some informants explain the use of waterfall practices as a part of a hybrid model. They believe it wouldn't have been possible to stick to only agile methods as

the number of people involved and parallel processes with dependencies in the different companies made forward planning vital for their ability to work on their respective projects.

"The reason why we started out with the traditional waterfall method was because the companies needed a set requirement specification in order to start the development on their side. Both the software company and the pension providers worked agile" - PM1

"In retrospect, it was a little strict and a waterfall method because it was a new regulation, this is the law, this is the date so we had to be strict, but at the same very agile since we didn't have time to go into details and it was a lot of people involved and lots of stuff. We had to be very agile, but it was big and complex and X probably felt he had to be strict to get us to the finish line, but at the same time be understanding and open to suggestions and ready to adjust when he realized we had missed something. It was really demanding and a hybrid model project, but we do that a lot of the time, it's always some new regulations, some big and some small." - HSD1

At the same time, the manager of the internal project at the large pension provider says that the requirement specification wasn't strict enough, as it left some parts open to interpretation. This led to different interpretations among the industry companies, where all were right according to the document. This led to further discussions among the representatives, and it was clear that the actors wanted a solution that required the least amount of work in their specific system and interpreted the document to fit their systems. These discussions were lifted to the advisory committee, where a decision was made.

"Yeah, the requirement specification was a little loose and could leave some room for interpretation. And then it was different actors who interpreted how the solution would work in different ways. That led to some discussions" - PM4

Table 6 summarizes the mechanisms discovered in the planning phase based on the categories described in section 3.4.

Category	Mechanism	Description
Absolute demands	Consultation rounds	The consultations round was an opportunity for the industry to come up with their input on the law
Documentation and absolute demands	Requirement specification	The original requirement specification was made by a team consisting of representatives from different companies in the industry and sent out to consultant groups
Industry collaboration	Functional task force	Task force consisting of representatives from the industry who created the requirement specification
Industry collaboration	Competition	When competitors work together on a common project, it's hard to keep opinions objective
Industry collaboration and absolute demands	Forward planning	Due to the regulatory aspects of the project, a strict deadline and cross-company dependencies, forward planning was important for the project

Table 6: Identified mechanism in the four categories in the planning phase

4.1.3 The development phase

The delivery of this project was a technical solution that answered the demands forced by the EPK law and the following regulations. To fulfill these demands, the technical solution had to be implemented so that all pension providers in the industry could use it to transfer PKBs among themselves. This resulted in two different development processes, one central process developing the common register, PKR, and 13 individual processes developing support for such transfers in all the provider's existing systems. In this study, we cover the views of four of these 14 organizations engaged in the development: two of the pension providers, the managing consultancy, and the software company developing the

central register. Of the two pension providers, one was a large significant actor, and the other was a minor one.

Development of the central register - PKR Following the planning phase described in the previous section, the development of the PKR started in December 2019. This was considered an agile development project and was organized into ten sprints lasting from 15. December 2019 to 4. August 2020, each sprint lasting for three weeks. The two first sprints, lasting from 15. December to 6. February did not involve any development, only the production of solution documentation and infrastructure preparations. After the first two sprints, the software company should present a solution description and complete API documentation to the functional task force and the managing consultancy. This resulted in around 35 API endpoints being presented. The industry project saw this as way too many and too complex, so the software company reduced the number of APIs to 15 significantly more complex ones.

The subsequent eight sprints were focused on development, and all functionality to be worked on was predefined in each sprint. As this software company had a strong focus on following the agile mindset in all its projects, it was also the obvious choice for this task. This project was, however, considered a special case compared to their other projects. This company usually delivers software products to less demanding customers, where the choice of processes and methods are entirely up to them. However, as mentioned in the planning section, it was in the RFP process a wish that the software company should work agile, so there were initially no discrepancies in this matter. Nevertheless, the early demand for a solution description created a more static development plan than an Agile team would normally prefer. Despite this, the PKR development team was set on changes in the development plan and initiated several Agile practices, including sprint planning, retrospectives, demonstration, continuous testing, and the scrum master role.

This project faced ever-changing demands where regulations were determined in parallel and functionality was continuously added or changed. These demands were arranged in task forces, as introduced in the previous section. Every time project issues were identified by the software company or the individual actors; they would be discussed and solved in the relevant task force. Legal issues were present throughout the project as this solution was evoked by a legal act and

faced several regulatory demands. These issues could be raised in all parts of the project, such as in development teams, at demonstrations, by the industry actors, or the in the project management. The managing consultancy would then ensure this was discussed in the legal task force. As PM2 stated *"It was legal issues all the time ... it was very nice to have the legal task force with legal experts from the industry actors and Finans Norge."*.

The PKR regulations were also not finished at the point of development. This was, of course, challenging, and PM2 stated *"The regulation was not finalized, but we had a preliminary draft of the regulation ... the Ministry of Finance had set a go-live date, so we realized that we had to start before the final version of the regulation was ready"*. However, this ongoing regulatory work made it possible to have some influence and collaboration with the legislators. This was accomplished through the ministry task force, which was set to work with the implementation group in the Ministry of Finance. Several participants saw this opportunity as a great advantage, as former experience had shown the ministry's difficulties with understanding technical limitations. Overall, as the project faced continuous changes and unpredictability in many areas, the managing consultancy and the software company agreed on a pragmatic relationship. This implied not having a rigid contract but rather accepting some deviation from both parts during the development period.

During the entire development period, the functional group had meetings to discuss the functionality of the PKR. This group consisted of representatives from the software company, the managing consultancy, and the five biggest pension providers. In the first three months of the development period, these meetings were held physically at the consultancy's offices. However, due to the coronavirus pandemic in March of that year, these meetings were held digitally for the remaining part of the project. According to LD1, was this a relief as the traveling was time-consuming and made the meetings less flexible *"The first three months we had to have physical meetings ... we were there for a challenging 2-hour meeting, had to travel back and forth, and then the day was suddenly gone ... with corona the efficiency sky-rocketed"*.

At the end of each sprint, the software company held a demonstration for people from the consultancy and 5-8 of the biggest pension providers. These demonstrations were meant to present the progress and a forum for questions and feedback. However, in addition to this, they became a precious forum for discussion

and clarification among all actors where new problems were raised, and new functionality was proposed. PO1 from the software company described this phenomenon as proof of why the preplanning should not be too detailed, stating *"You can plan or develop as much as you want, but the moment people see the actual screenshot of the application, that's when the true ideas and solutions get generated."*. In relation to each sprint and demonstration, continuous testing was also performed. The test environment was up and running three weeks into the development period, so after each sprint, the new functionality was released in this environment.

The PKR software solution was heavily documented. The initial 70-80 pages of architecture documentation, solution description, and API references were followed up by a continuous documentation process during the development period. API documentation was produced and updated in relation to the continuous deployment to the test environment. This enabled all the industry actors to get on-boarded and use the test environment continuously as new functionality was added each sprint. This documentation was produced by *OpenAPI* (Linux-Foundation, 2023), software for automating the API documentation process. The generated documentation was also added to *Confluence* (Atlassian, 2023a), an all-in-one solution for project management. The detailed functional description was also updated continuously as new functionality was added and changed. LD1 stated that this was very useful since discussing the functionality specifics with the industry actors *"just by the phone"* with no schematics to relate to would be very hard. The solution documentation was seen as extensive, a view confirmed by LD1 stating *"This is actually one of the most well-documented solutions I have worked with"*.

This was a pure API-based solution from scratch, so the technical implementation was not considered complex. LD1 from the software company stated *"Technically speaking was this no groundbreaking functionality or technology, we used standard of-the-shelf solutions and known technologies."*. This was, as presented in the next section, opposed to the implementation in the core systems of the pension providers running on large mainframe computers. PO1 responsible for the PKR implementation said *"They [the pension providers] have mainframe computers, old and lots of integrations, it is multiple systems with different customer data, to them this project is significantly more complex. We, on the other hand, would start with clean slates, write only APIs ... that's why I have some understanding that they needed a specification in advance."*. This was also backed up by another informant mentioning the industry actors' legacy systems, stating that they had a truly challenging task.

The industry actors' internal development In parallel with the development of the PKR, all ten industry actors had to customize their internal systems to correspond with the PKR. As informants from the software company implied, this was not a straightforward task. All these companies had different systems, many running on large mainframe computers with various integrations. Also, all had their interests and did not wish to develop any more than necessary. This led to a challenging task for the managing consultancy, having to find compromises and guard the implementation orders given to the software company.

The tolerance for including functionality requested by the industry actors had been high during the planning phase, but it became severely lower in the development phase. To limit the requested change orders from the industry actors, they were evaluated against five guiding principles established at the beginning of the project. PM1 shed light on this issue and stated *"If all these ten companies had talked to the software company directly, the software company would get opposing orders all the time, and that would just not have worked."*. In this study, we have collected data from two of these industry actors, one significant and one rather small.

The large industry actor studied is one of the five significant actors who initiated the industry project as a response to the EPK act passed in the parliament. In this company, a development project would usually be initiated by the business department based on some business goals. This department would then draft a requirement specification, which would be processed in collaboration with business analysts and translated into technical requirements. Cross-functional teams would later develop the software, while the business department was the product owner and responsible for the solution. More formally, their development model was described as closely related to the classical V-model.

In the EPK project, their development process was described as slightly different. The requirements specification to be used in the internal project was derived from the initial solution description created by the software company. As described in the previous section, this was quite uncertain, and changes were expected. As PM4 stated *"It was challenging, the specification on the PKR solution was a little vague and made room for interpretation. The industry actors interpreted it differently which led to discussions."*. As the common industry project, in fact decided the solution requirements and were subject to changes and interpretations, the internal business department was instead part of an iterative process. DM1 stated *"This project was regarded as a more technical delivery than a cooperation with the business department,*

at least in the beginning before the solution was testable ... Instead of receiving demands from the business department, the solution was built from a technical perspective". The iterative process had two layers. One layer consisted of cross-functional development teams with business people, business analysts, and developers. The other layer consisted of the management committee making crucial decisions.

The development project was exposed to several control mechanisms during its iteration cycle. Although evoked by a regulatory demand, this project was considered a development of a new customer product. Following this, many of the standard processes required when creating a new product were also applied in this project. One of these processes required legal support. In this project, two types of legal support were used, product-oriented and compliance-oriented. This legal intervention was mostly present at the beginning of the project, where some legal demands were set in collaboration with the business department. It did, however, occur that these legal experts also gave advice directly to the cross-functional team during the development. It also happened that the compliance jurists were involved in the preparation of acceptance testing, which is the responsibility of the business department and was performed at the end of the project. Developers and business analysts also performed tests on the internal project during development. These were unit tests conducted by the developers continuously, but also regression tests and acceptance tests if the software was deployed. As a result, some functionality was continuously deployed to production. However, more complex features affecting other system parts were deployed semi-annually as part of a bigger release. In relation to this release, large regression and acceptance tests would be performed to ensure that no existing functionality would be undesirably affected. Due to these semi-annual releases, a skeleton of the EPK project was deployed to production six months before being used.

As expressed earlier, this internal development project was implied to be challenging by other parts of the project. This was due to their software architecture and the fact these systems already had a lot of existing functionality to consider. To overcome this challenge, the company created a more flexible architecture consisting of two layers. The first layer was the mainframe computers responsible for the core functionality involving software processes across the entire company. The other layer was put outside the mainframe and consisted only of the EPK functionality. This layer was built in *Flowable* (Flowable, 2023), a low-code business automation platform. This platform generates applications based on busi-

ness process models and was the layer communicating with the PKR. This flowable layer was the intermediate link between the mainframe and the PKR and triggered activity in the mainframe. There were, therefore, several dependencies between the team working on the mainframe and the team working on Flowable. These dependencies would be handled in weekly status meetings and were described as *"long and circumstantial"* by SD4. Both teams used sprint planning to organize their work in predefined periods, but only the Flowable team used practices such as pair programming, demonstrations, and test-driven development.

Several of the practices in the internal EPK project were well documented. The overall development process used, earlier described as the V-model, was documented and described in the internal wiki. Every event following a demand placed by the business department is stored in *Jira* (Atlassian, 2023b). This includes all discoveries and adjustments performed on that task. When a certain functionality is implemented, the Jira reference of the corresponding task is connected to the code produced. The produced documentation is used for collaboration and later in an IT audit. The IT audit aims to check that systems comply with regulatory and technical demands and that all the defined processes are followed. When describing IT audit, DM1 stated *They [the auditor] take a random sample from the documented Jira tasks and checks if the respective roles (i.e requirement specifier, business analyst, developer, and deployer) has executed their task according to the development process (V-model). It would be considered a compliance breach if any of the mentioned roles deploys without getting a formal signoff from relevant business-/risk owner.*.

The other industry actor studied had a minor role in the industry project. They were kept in the loop on the development of the PKR but did not have the resources to contribute to or influence the project to the same degree as the five most prominent actors. However, all participating actors had to initiate an internal development project to communicate with the PKR and offer the EPK product to their customers. As this company was the subsidiary of a larger conglomerate, they were running on the core systems of their parent company. Following this, the development of the internal EPK had to be distributed across multiple locations. The core systems were running in Denmark, but parts of the functionality were outside the core systems and developed in Kuala Lumpur. This functionality was assembled in an external component known as the Pension Connector Hub (PCH). This hub was the intermediate link between the PKR and the core systems of this company.

Although the development of the internal EPK system was done in other countries, the company in Norway did testing and coordination with the industry project. The testing was performed continuously in cooperation with people in Denmark for the last six months before the EPK solution would go live in 2021. However, BA1 implied some uneven division of labor stating *"The final 3 months were rather intense as the deadline approaching quickly"*. A complete test environment for the core system was developed, where two dummy companies were created to test transfers the companies through the PKR.

Table 7 and 8 summarize the mechanisms discovered in the development phase based on the categories described in section 3.4.

Category	Mechanism	Description
Documentation	Solution description	70-80 pages of solution description and architecture produced prior to development. This was continuously updated on Confluence.
Documentation	API documentation	This documentation was produced and released to external users continuously using OpenAPI and Confluence
Industry collaboration	Demonstrations	Demonstrations of the PKR were continuously held for multiple actors
Absolute demands, Industry collaboration and Involvement of specialists	Ministry task force	Group of legal experts and project managers communicating with the ministry on the regulations.
Industry collaboration	Pragmatic relationship/Flexible contract	A pragmatic relationship between the project management and the software company was agreed upon.
Industry collaboration	Test environment	Functionality was continuously deployed to a test environment so all actors could test against their systems.
Absolute demands	Flexible architecture	To layered architecture were used, where most of the functionality was put outside the core systems in Flowable processes.
Industry collaboration	5 guiding principles	Principles used by the project management to justify denials of change requests.

Table 7: Identified mechanism in the 4 categories in the development phase

Category	Mechanism	Description
Involvement of specialists and absolute demands	Legal intervention	Product-oriented and compliance-oriented jurists involved in planning and control of development.
Involvement of specialists and Absolute demands	Acceptance testing	Thorough acceptance testing by business analysts, business department, and occasionally compliance-jurists
Documentation	Defined development method	The development method was described and illustrated on the internal Wiki.
Documentation	Task tracking	All orders placed by the business department was documented and updated in Jira by business analysts and developers as the order was processed.
Documentation	Living traceability	The orders are connected to specific code in Jira in the development.
Absolute demand	IT audit	IT audit were performed twice a year where documentation was controlled.
Involvement of specialists	Legal task force	A legal task force with legal experts from the different actors were used to solve legal issues.

Table 8: Identified mechanism in the four categories in the development phase

4.1.4 The test and deployment phase

In parallel with the development phase, the managing consultancy led the work of developing test cases and a complete test plan for the industry test. This test was an acceptance test wish would ensure that the PKR worked according to the defined specifications when used by all the industry actors. This industry test began in August 2020 and went over 12 weeks. The five biggest actors had to participate in this phase so that the chosen test cases could be tested. PO1 in the software company described these tests as *"Extremely comprehensive"*. During

the same time, they still had to implement new functionality as new regulatory changes arrived. These late changes were, however, predicted, so the data model was customized to handle such additions.

Most of the test cases against the PKR were predefined and delivered to the most prominent industry actors by the managing consultancy. Some tests were also supplemented later when seen the need for. The large industry actor studied had the goal of finishing their internal development prior to this test period. This goal was not reached, so interface testing and API requests performed some of these tests in a stubbing environment. DM2 at the industry actor stated *"It was suboptimal, but we were able to test by having some functionality in place"*. It was also mentioned that some other actors tested only with a stubbing environment, indicating that numerous actors had difficulties finishing their internal projects on time. The test plan was divided into sprints, defining what functionality to test in the different periods.

There were several challenges related to testing against other actors. The main challenge mentioned was cooperating with an external party instead of just fixing problems internally. To manage this cooperation efficiently, a Teams-channel was used to report concerns and issues. This channel consisted of the managing consultancy, developers from the software company, and relevant people from the different industry actors. On bigger issues, a Teams-call was set up to solve the matter. All the tests planned and performed were documented in a large excel-sheet describing the processes and tasks to be completed by different people. Another challenge mentioned was testing the money transactions, which are very hard to test in a test environment. To solve this, penny testing was performed in the production environment using dummy pension agreements to transfer small amounts of money between pension accounts. Multiple issues were discovered when performing this penny testing.

The minor industry actors were not part of the initial acceptance testing; they were required to pass some test criteria to be allowed to connect to the production environment of the PKR. This test was initiated in December and performed among five minor companies, as the large actors were done with their test period. The minor actor studied described challenges relating to other companies and not doing everything themselves.

The system was by law set to be publicly available on 01.01.2021. However, the industry project could not comply with this deadline but agreed with the legisla-

tors to postpone the deadline by one month. This additional month was used to upload all the pension agreements from all the actors, which were multiple million. DM2 stated in relation to the upload *"We began the first of January, and had a really short deadline"*. Following this, 70-80 billion NOK of financial pension assets were to be realized and transferred between the actors to uphold the purpose of the law. As such asset movement in a short period would have influenced the markets, there were strict rules on the amounts of transfers that were allowed each day. This urgency was expressed by DM2 stating *"There were strict rules on what you had to do when, and you must deliver. You cannot accept having problems with your systems, you just had to deliver as promised"*.

Table 9 summarizes the mechanisms discovered in the test and deployment phase based on the categories described in section 3.4.

Category	**Mechanism**	**Description**
Industry collaboration	Industry acceptance test	All big actors participating in an acceptance test for the PKR.
Industry collaboration	Test communication in Teams	Issues related to industry tests were solved in a Teams channel
Documentation and Industry collaboration	Test-plan	The industry test-plan was documented and updated as tests were performed.
Absolute demands	Penny-testing	Penny-testing performed in production as money transactions are challenging.
Industry collaboration and absolute demands	Test guard	All actors had to pass certain tests to be allowed to access the production environment.
Absolute demands	Strict rules	Certain tasks had to be done according to very strict rules to avoid influencing the financial markets

Table 9: Identified mechanism in the 4 categories in the test and deployment phase

4.1.5 Identified agile practices in the EPK project

Related to RQ1 - *What agile practices fit a large inter-company regulatory-evoked project?* - we have identified several agile practices used in this project by following the method described in section 3.4. All of the identified practices are mentioned in the project description in the three preceding sections and can be found classified in table 10.

As shown in table 10 are the concepts *Inspect and adapt cycles*, *Working collaboratively* and *Continuous customer involvement* involved in many of the practices compared to the concept *Incremental design and iterative development*. This might suggest that anticipating change by working iterative and designing software incrementally was harder to fulfill than the other concepts. However, some of these practices overlap with the mechanisms found in the preceding sections. This includes consultation rounds, ministry task force, and sprint demos. This implies that these agile practices overcome some of the challenges of being agile.

Practices identified	Incremental design and iterative development	Inspect and adapt cycles	Working collaboratively	Continuous customer involvement
Consultation rounds			X	X
Backlog grooming		X	X	X
Userstory workshops			X	X
Ministry task force			X	X
Sprint planning	X			
Sprint demo		X		X
Continuous testing		X		X
Simple design	X	X		
Domain specific communication channels			X	
Coding standards		X	X	
Flexible contract /pragmatic relationship			X	X

Table 10: Identified practices with agile concepts

4.2 Challenges and mitigations of agile practices in regulated environments

Our informants generally considered this project very successful, especially compared to a former implementation of a new regulation that could have been a cooperation project, "Aksjesparekonto". However, there were some challenges in implementing agile methods in a project of this nature, and we will present the challenges in the different categories described in section 3.4, and how the project tried to mitigate these challenges. By this, we begin to answer *RQ2: What challenges are faced when trying to use agile practices in a large- inter-company regulatory-evoked project?* and *RQ3: How can challenges faced when trying to use agile practices in a large inter-company regulatory-evoked project be mitigated?*. The results given in the following sections will be of the mechanisms found in the tables (6, 7, 8, 9), presented categorically.

4.2.1 Involvement of specialists

A project of this nature demands heavy involvement of specialists, mainly from the legal perspective. New regulations often touch on existing laws and systems which require minor adjustments. However, the EPK law was a significant change from the status quo. With the need for a completely new system, which requires all the companies to interact with, the regulatory implications were many and complex. A task force comprised juridical specialists from different companies to answer this challenge. The juridical task force ironed out the regulatory consequences and how to keep the hub compliant. They also discovered challenges where they needed to get new authorizations in the law to use and keep different information. During the development, the juridical task force dealt with several inquiries about legal issues and compliance. Issues would be brought to the project management, who would consult the juridical task force and make decisions with their input.

"There were questions all the time about the law. We had a legal task force with representatives from the pension providers and Finance Norway. We needed them to interpret the law, and to help assist in making a data processor agreement, cooperation agreement between the companies, affiliation agreement to the technical solution." - PM2

4.2.2 Documentation

Many parts of this project were documented. Most of the documentation was produced prior to development. This was requirements specification, solution description, and API documentation. This documentation process was performed to predict the finished solution or at least some sense of the finished solution. However, the initial documentation failed to predict the outcome of the development, and the result was radically different than first presented in the documentation. The changes appearing during the project were continuously added to the documentation, but the focus on detailed documentation eventually declined. There are some challenges related to this documentation. Several study participants argued that some basic initial solution documentation was critical in coordinating a starting point between the affected actors. Others saw it as unnecessary since the solution would look different than initially documented anyway. A solution discovered in this project is using OpenAPI to automate the documentation process.

Some of the documentation also aimed to assure some final validity of the product or utilized process. This includes order tracking, task tracking, and test-plan documentation. This documentation was used in IT audits to ensure that the software solution complies with all functional and regulatory requirements. The test plan controlled an extensive acceptance test, ensuring that the software company had delivered according to functional and regulatory requirements. The challenge faced here is that developers must spend time documenting every step performed to satisfy the IT audit. This is, however, solved quite efficiently with Jira references. A predefined comprehensive test plan is challenging, as predicting the tests needed to accept the solution is hard. The project solved this by making room for late changes in the test plan.

4.2.3 Industry collaboration

Industry collaboration is one of the new discoveries in our analysis. The project differs from other cases due to the collaboration of a whole industry which led to new and different challenges. The competition among the collaborators was a new challenge, and one of the primary considerations was how much you could and should share. This could vary from company to company depending on if something could give you a competitive advantage, the company's size, or the

business strategies.

"It was time consuming, and at times the companies had conflicting interests both from a strategic, commercial and technical standpoint" - PM2

As mentioned in 4.2, there were several discussions where the company's interests were clear and often conflicting based on their commercial plans around the EPK regulation and IT systems. To mitigate these challenges, the consultancy acted as a neutral third party overseeing the process. The consultancy decided on five principles the project should be built around and would be used to guide them through any company conflicts. The five principles were:

- Equal competitive terms
- Minimum solution
- Open and transparent
- Flexible for the future
- Neutral solution

"The project established some guiding principles. Often when you have principles like that, you make them and leave them in a drawer, but we used these continuously throughout the project. When the companies did not agree, we as consultants acted as a neutral third party" - PM1

The five principles were used often and helped the consultancy as the neutral part of the project without any self-interest other than delivering the project on time.

Another challenge with a collaboration project was the number of parallel development processes with dependencies between them. To use an agile method for a project of this nature was deemed too complex, as mentioned in 4.2, so they made a detailed requirement specification, got a solution description, and had several discussions with the different task forces before the development could begin. During the development phase, the demos described in 4.3 became important as they gave a clearer understanding and a natural way to bring up issues and discuss solutions. It was also a way to ensure everyone was involved in the discussions, keeping everyone up to speed and on the same page. Beforehand, the

forward planning was important as it made it clear to the whole industry what they could expect and when to expect it.

A lot of the functionality of the PKR requires several actors, so the companies needed to help each other with the testing, as mentioned in 4.1.3. The consultancy made a test plan for different scenarios that the companies should test their internal system against. Due to time constraints, the large pension provider couldn't do this against their mainframe, but they managed to do interface testing with API requests. During the testing, you often depended on another company and dialogue to see what happened on the other side and if things worked as they were supposed to. To mitigate this challenge, an industry test group discussed and helped each other with the testing of the systems and a dedicated Teams channel for questions and help.

4.2.4 Absolute demands

When a law is passed, it is absolute, and it is a limited possibility to influence the outcome. This challenge is mitigated by participating in consultation rounds initiated by the parliament. Although these consultation rounds are standard procedure when a law is passed, it was, in this case, extra important to use them as the law had unknown technical aspects. When the law was passed, no one knew the technical implication of such a law. It was the pension providers themselves who realized that they would need a common technical solution. From our informants, it's clear that the lack of technical understanding among the legislators has been a challenge for the project.

"The problem is that the law, the ones who wrote it, have no clue what complex solution such a law would result in" - LD1

"In projects like this, it is important to understand the regulators and authorities don't always understand the technical implications and limitations... Today I work on different projects together with the government, and I have the same experience in some of my projects" - PM1

To mitigate this challenge, the project also managed to establish a task force to work with the ministry. Once again, the consultancy's reputation as a neutral third party with no self-interests, combined with the input they received from the pension providers, gave them a lot of credibility in their discussions. Having

such a group gain the trust of the ministry is a rarity but is thought of as one of the reasons for the success among the informants.

"In one of my ongoing projects we do not have a task force working working with the Ministry of Justice like we had in the EPK project, and this makes the decision process more complicated" - PM1

When a law is passed, it is also decided when it comes into force. This date is final, which gives any development project tied to it a hard deadline. A strict timeline was set to ensure that the product was ready for the deadline, with the acceptance testing set to begin six months before the deadline. The adaption of a more waterfall-inspired method centrally was to make sure that the project would get to the finish line in time. The hard deadlines were also seen as an asset by the project management as they created a sense of urgency.

If we didn't have a regulatory deadline, the projects would have taken several years to complete. My experience is that creating a sense of urgency in development processes is the key to success." -PM1

Projects arising from absolute demands often pose unique challenges as they may lack enthusiastic support from management and developers. Furthermore, these projects typically lack a clear customer need or evident advantage for the company, making them appear obligatory rather than value-driven. In the EPK case, the industry would lose 80 million euros which makes it clear that this project isn't value-driven for the industry. This challenge was mitigated by using the motto stated by the consultancy saying *"More pension for each krone saved"*. This motto was used baseline in many decisions.

While the law was passed in early 2019, the last adjustments to the regulations happened in the fall of 2020. The project had some ideas of how things would look due to their discussions with the ministry, but the uncertainty of things would have to be in the end was a challenge. As a result of this uncertainty, they decided to have an early deadline and have some slack in the plan to make room for adjustments and implementations of changes.

"We had to have a much tighter deadline, than what we would have if we knew there wouldn't be any changes. We had to leave a lot more slack in the timeline than what we would have needed if we were to have the final regulations at hand from the start." - PM2

The developers at the software company who worked with the industry project

mitigated this by working iteratively in sprints in an agile manner. They also noted that this challenge wasn't as big for them since they worked on a completely new system, the challenge was greater for the internal projects.

"The biggest challenge with the late changes wasn't necessarily for the software company. The software company would probably be able to implement the changes, but then the rest of the companies would have to implement and adjust their systems before the deadline." - PM2

When the solution at the minor pension provider was designed, they kept the possible future changes in mind. It was clear to them that things would be added and changed before the law came into force. As a result, they created a more flexible architecture with a focus on modifiability.

"So we knew that parts of the process probably would change before deployment or close after. That made us very careful in the design of the solution. We knew there would be changes, so we made decisions that would make us more flexible and ready for them when they come" - SD4

5 Discussion

This section discusses the findings presented in the results. First we gather the main findings in a figure were we connect mitigating mechanisms to the categories, and the propositions derived from the mechanisms. We then explain our contribution to the research problem stated as our research goal - *How agile methods can be adapted in projects evoked by regulatory demands*. This is done by arguing for the responses to the RQs in the previous section. We first discuss selected mechanisms identified in each category and compare them to the reviewed literature. The discussed mechanisms were selected as they were seen as the most prominent for the success of the project. We discovered several agile practices used in the project, found in table 10. Some of these agile practices also fit into the challenge categories, such as the three viewed in figure 5. The same goes for consultation rounds, continuous testing, and domain-specific communication channels. These are considered the most relevant practices to this study and will be discussed further in this section. The other agile practices identified fit well in this project but did so independently of the regulatory context and will, therefore, be discussed in their own section, independent from the categories. At the end of each subsection discussing the different categories, one or more propositions will be proposed in relation to that category. By this, we make the second contribution *Propositions that can form a basis for a new theory on agile software development in a regulatory context*.

5.1 Prominent mechanisms

During the analysis there were a few mechanisms that stood out as more impact full than others for the projects ability to stay agile. Three of these mechanisms are also classified as agile practices and can therefore be found in table 10 as well. Others are non-agile mechanisms and can be found in either of the tables (6, 7, 8, 9) on mechanisms presented in the previous section. These mitigations is the base for the propositions that will be stated when we discuss their respective categories. The relation between the challenge categories, selected mitigating agile mechanisms/practices, selected mitigating non-agile mechanisms, and the propositions to be stated can be found in figure 5.

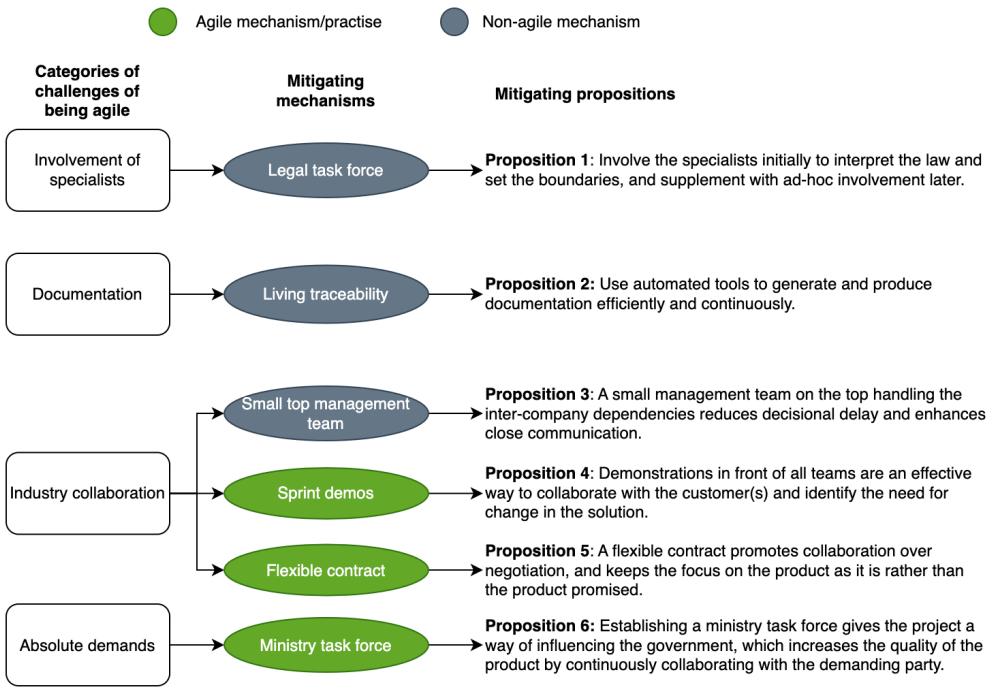


Figure 5: Figure describing the relationship between the challenge categories, the mitigations, and the resulting propositions .

5.2 Involvement of specialists

The involvement of specialists is a challenge for agile software development methods in regulated environments, as described in section 2.3.1. In our results, we find some of the same challenges, as the inclusion of legal is a significant factor in the development process. There was no structured legal involvement during the project, as some of the existing methods suggested (Stirbu and Mikkonen, 2020) (Fitzgerald et al., 2013). The legal team did a lot of work in the planning phase before the development started, which has some similarities to LoD-PQR (Poth et al., 2020). Compliance and regulations were worked into the development tasks given to the developers. However, with the adjustments to the regulation after the law was passed and the development had started, the legal team also had to follow up later in the process. This was done in an organic manner, with the legal team involved when their input and expertise were needed. This way of working stays more faithful to the agile manifesto with less ceremony and structure. The legal team did not interact directly with the developers, which is the preferred method of communication in agile practices as it encourages face-to-face conversations (Beck et al., 2001). They instead communicated through business analysts or project managers. The informants explained this with the need

for someone to work as a "translator" to ensure they understood each other. As compliance and the regulations were worked into developments task, there were no checks during the development. The legal team did have some input for the acceptance test, but this test was mainly for functionality, not a legal checkpoint.

There were no complaints regarding the involvement of specialists in this project, which might suggest that this approach suits the regulated environment. Therefore we suggest the following proposition: *Proposition 1:*
Involve the specialists initially to interpret the law and set the boundaries, and supplement with ad-hoc involvement later.

5.3 Documentation

This study supports some reviewed literature on the need for documentation in the regulatory domain. Section 2.3.2 describes the need for documentation to prove that you comply with regulations. An example of such in this study is the IT audit. Documentation in the studied case is produced partly to comply with the demands when checked by the auditors. To satisfy the auditors, tasks are tracked from their origin to their finished implementation. The developers also use Jira references to link tasks to specific parts of the code. This practice was also found in the literature and referred to as *living traceability* in the R-Scrum method (Fitzgerald et al., 2013).

The literature reviewed also mentioned using automated documentation tools such as *CompilancePal* (Stirbu and Mikkonen, 2020). In the literature, this was presented as a way to mitigate documentation's negative effect on agile practices when documentation is needed. The use of such a tool for the same purpose was also found in this study. The need for API documentation in this project was undisputed. Using *OpenAPI* to automate this process might therefore be seen as the most agile way to accomplish this necessity. One can argue that the need to produce API documentation is independent of the environment being regulatory or not. However, the urge to produce it prior to development is probably so. This issue can be seen as more relevant to the *Industry collaboration* category and will be discussed further in that section.

The probably most prominent example of documentation discovered in this study is the solution description. The urge for such a pre-planned description was mainly due to two circumstances: the cross-company industry project and its

demanding origin. This will be discussed in later sections. However, it was also spent time updating this documentation continuously, which is a use of resources that, based on the reviewed literature, clashes with the agile mindset. Despite this, none of the developers in this study complained about spending time producing documentation. Some mentioned its usefulness, as described in section 4.1.3. It is multiple explanations for this, I) the documentation tools used are efficient, II) business analysts with technical knowledge might share this responsibility, and III) the documentation production might be seen as necessary and "worth the trouble". Independent of the developer's views, was the documentation process given some slack. PM2 justified this decision by stating *"The first versions of the software documentation were extremely heavy, and the project management team together with the companies realized that we had to take on a more pragmatic approach to documentation in order for the developers to have time to develop the solution as well. Hence, we agreed on which parts of the documentation that was necessary for the developers working in the different companies to develop their solution. For instance, the API documentation was essential for the pension providers."*. Based on this, we can assume that the project started with rather extensive documentation demands but later saw the resources better spent elsewhere. This perspective might be more in line with the view agile research has on documentation.

Based on this discussion, we propose the following proposition: *Proposition 2: Use automated tools to generate and produce documentation efficiently and continuously.*

5.4 Industry collaboration

We see no examples of industry collaboration in the reviewed literature on agile development, but it significantly impacted the EPK project's agility. The main problem was the number of parallel development processes with dependencies. This required a lot of forward planning, leading to several of the documents mentioned in 4.2.2. Forward planning goes directly against the principle of agility, which focuses on synchronization through activities and artifacts (Dingsøyr et al., 2022).

When we compare what we found in our study to the existing literature on agile software development methods in regulated environments, there are some differences to keep in mind, especially regarding industry collaboration. The col-

laboration of a whole industry is rare, so some of these challenges might not be a part of another project in a regulated environment, especially those connected to the competition. However, as mentioned in section 2.2, such regulations are expected to be more frequent in the future. Of the other challenges, the number of teams is just as important as the companies they belong to.

An interesting question is how much planning was really necessary. As the PKR was going to be developed by an external organization, they had to do the requirement specification. However, the returned solution description and the API document might not have been necessary. While both the central management and the industry claimed that this had to be ready for the industry to develop its products, the software provider believed that the solution description and the APIs wouldn't be representative as there would be a lot of changes down the road. It might lead to them having their hands tied. When the software company was chosen, they had made it clear that they would be working agile, but they were not given the complete freedom of agile methods, at least not from the beginning. This can give the impression of the central management thinking of agile as something nice to have but something you can't trust with the critical things. Ultimately, the final APIs had nearly double the endpoints as in the API documentation delivered before development, and no one complained. This suggests that the changes to the original document didn't affect anyone, and you can question if it was necessary.

According to the reviewed literature (Dingsøyr et al., 2022), one of the most significant challenges with large-scale agile projects is the inter-team coordination of dependencies. When using our chosen framework for analyzing agile practices, it is hard to classify such inter-team work as "working in close communication". This is because the decision-making and requirements specification is distributed across different teams. In this project, these dependencies also span across sites and companies. To solve this issue, the EPK project had the top team consisting of management consultants. This team was small and neutral, and all the coordination went through them. On multiple occasions, there was direct communication between the other actors, but the managing team either coordinated it or they were participants. The essence was that this small team always had a complete overview. Using this structure, the project reduced the inter-team dependency issue only to include inter-team dependencies between each team and the top managing team. This made close collaboration significantly easier and was deemed a success factor in this project by multiple participants. In this regard, it should be

mentioned that this team had an extreme workload throughout the project. One could argue that autonomous teams instead of such top management would be a more agile approach, as implied by other studies on large-scale agile (Dingsøyr et al., 2022). However, this would be very challenging as the participating teams worked on many different systems in different companies.

Some of the consultancy's five principles to guide them through the projects fit well with the agile mindset. *Minimum solution* lets the software company produce working software faster, making it easier to do incremental design and iterative processes. *Flexible for the future* helps the team keep the solution receptive to possible changes. The agile software method isn't just the implementation of new processes; it is also about the agile mindset, which is strengthened by choosing principles that supports it.

The testing was also well planned and had its phase at the end of the project, which isn't necessarily very agile. The fact that actors had to test with API calls and stub environment indicates a strict schema with little room for changes. The informants explained the need for such a strict plan to ensure everyone was on the same page and that they needed the companies together to go through the tests. The test environment, which was up and running after just three weeks, bare some resemblance to a regulatory sandbox which has been suggested as a way of staying agile in regulated environments (WorldBank, 2020). This allowed the companies to continuously test their solution with the existing version of the PKR during their own development. In addition, there was a dedicated teams channel for testing, where the testers from all the companies could ask and help each other.

There have been other significant new regulations in the financial domain, such as "Aksjesparekonto", where the industry didn't collaborate, creating chaos as there wasn't a system ready to handle it. [6] and was seen as a disaster, both by the public and the industry. *"We then used "Aksjesparekonto" as an example. Suddenly there were new tax laws that no one was prepared for and there was no system for transferring accounts between the companies, and there were lots of weird things happening and in general chaos"- DIP1* This project, on the other hand is seen as a success by all involved which suggests that a industry project benefits everyone involved and the applied method in this case might be the best suited for a project of this nature.

[6] https://www.nrk.no/norge/innforing-av-aksjesparekonto-har-endt-i-kaos-1.13819930

Based on the discussion in this section we propose the following propositions:

Proposition 3:
A small management team on the top handling the inter-company dependencies reduces decisional delay and enhances close communication

Proposition 4:
Demonstrations in front of all teams is an effective way to collaborate with the customer(s) and identify needs for change in the solution.

Proposition 5:
Flexible contract promotes collaboration over negotiation, and keeps the focus on the product as it is rather than the product promised.

5.5 Absolute demands

Absolute demands are one of the new categories identified in this study, including multiple subtopics such as hard deadlines, lack of influence, and lack of motivation. All of these topics are based on the law being non-negotiable, which can be seen as a drawback when trying to be agile in a project evoked by regulatory demands. The non-negotiable assumption is not entirely correct in the EPK case, but it is a fact concerning already passed laws. In the legislative work concerning the EPK law, the pension providers could influence the legislators to some degree before passing the law using the consultation rounds. However, even for technical experts, predicting the technical implication such a law would lead to is difficult. Many technical requirements are absolute when the law is passed, with no room for negotiation. These absolute demands contradict the core agile concepts such as *Incremental design* and *Continuous customer involvement*. As the EPK law was followed up by a series of specified regulations, not all absolute demands were set prior to development. This was exploited by the industry project using a ministry task force, which is considered an agile practice in our analysis presented in table 10. In this situation, the legislator takes the role of being a customer. Many participants saw this parallel design of regulations as challenging; however, one can argue that if the regulations were set before the development, it would be harder to follow the agile concepts.

When a law is passed, it is not only non-negotiable, but also the effective date is absolute. Many informants saw this as a challenge in relation to being agile,

as the hard deadline implied that they should follow a strict and detailed plan. Despite this, we found no evidence in the literature on hard deadlines counteracting agile practices. However, hard deadlines initiated parallel development of communicating systems, a situation encouraging agile development as the counterpart under development was subject to continuous change. The regulations were also subject to change throughout the project, which is another characteristic that aligns with using agile practices. All the identified agile practices in table 10 are in part used to handle the situation of continuous change; these will be discussed further in section 5.6. However, not all changes were regulatory; some were also a result of the industry collaboration, as mentioned in section 5.4.

Projects evoked by regulatory demands have a different purpose than most other projects. The end goal of most projects is to satisfy some customers. This was also the main goal stated in the *Agile Manifesto* (Beck et al., 2001). The legislators can nevertheless not be seen as regular customers. They have some common features in certain situations, one of which was stated in the previous paragraph, but there are other interpretations of their role. In the EPK project, the legislators set the overall goal of a project. Still, it was the industry actors and the individual business departments who actually acted as customers in this project. They develop a requirement specification together with a hired consultancy, order the PKR from a software company, and pay for the solution. Although possessing these customer characteristics, are this solution not driven by revenue incentives but rather to comply with demands and avoid penalties and loss of reputation. This issue was presented in section 5.5, but no informants in this study implied that this counteracted the use of agile practices.

Based on the discussion in this section we propose the following proposition:
Proposition 6:
Establishing a ministry task force gives the project a way of influencing the government, which increases the quality of the product by continuously collaborating with the demanding party.

5.6 Agile practices

We discovered several successful agile practices used in the project, which can be found in 10. The consultation rounds and ministry task force have been mentioned in 5.5, and continuous testing, sprint demos, and domain-specific commu-

nication channels were mentioned in 5.4, but the other practices didn't fit directly under one of the challenging categories.

Backlog grooming is a well-known agile practice that helps the whole team understand the work to do, the priorities, and the customer's needs. This was used by both the external and internal projects at the beginning of the sprints. Together with the sprint planning and sprint demonstrations, this formed the core of the agile ceremonies used in the project. These ceremonies lay the foundation for the project's iterative development processes and give both internal and external people involved chances to inspect and adapt according to the project's progress and results. They did not face any problems with the inclusion of the agile ceremonies, and the only feedback from the informants was positive about the effect they had on the project.

The user story workshops in the planning phase worked to get a shared understanding of what needed to be included in the solution and how the processes should work. In addition, it allowed the industry to work face-to-face with the software provider and transfer their knowledge of pension processes to the software provider.

The simple design of the solution was something both the developers and the central management agreed on. Keeping the solution simple simplifies further development and makes inspect and adapt cycles shorter to avoid misunderstandings and unnecessary work. It is a core part of the agile mindset and was supported by the principle minimum solution as discussed in 5.4

The project had its code standards to ensure the quality of the written code. Technical excellence enhances agility according to the agile manifesto(Beck et al., 2001). The code standard also makes it easier for developers to work together on the same code base and continue each other's work.

6 Conclusion

In our thesis, we have investigated the use of agile software development methods in a project evoked by regulatory demands. First, we have thoroughly described the project, from the planning phase to the deployment of the finished product. In our analysis of the results, we discovered two new categories of challenges in addition to the ones we found among the reviewed literature on agile methods in regulated environments. Then, in each phase, we have categorized the mechanisms connected to challenges with being agile in regulatory environments. Finally, we identified several agile practices implemented in the project and discussed how they work in a project evoked by regulatory demands.

We have discussed the implications of our findings in the different categories, which contribute to the discussion of agile software development methods in regulated environments. These discussions and the reviewed literature have led us to six propositions regarding using agile software development methods in regulated environments. Our findings show that using agile software development methods is possible in regulated environments. However, there are some parts where you need more traditional approaches, amongst them pre-planning and documentation.

Our research has focused on the financial industry, but several other industries are similarly impacted by regulatory demands for which our thesis might have implications. For example, in our review, the health sector was frequently mentioned as a sector of interest, as were other safety-critical industries such as automobile and aviation. Our findings can also help in these fields, but the differences in regulations must be accounted for.

As for future research, we hope other researchers could test our propositions in different contexts to see if they still hold. Secondly, we suggest investigating how coordinating mechanisms changes when going from inter-team to inter-company on a large-scale industrial project. Finally, another interesting topic is the different categories of challenges in regulated environments; these could be further investigated and tied to challenges in large-scale agile projects.

6.1 Limitations

The case study was limited to only a semester, which set some constraints on the number of interviews we could do. We ended up with 17 participants, which can be seen as a small sample size. The studied case included ten companies in the industry. Still, we only had access to two participating companies, so other actors may have other perspectives on the project than the ones who participated. However, the two companies we had access to had different roles and views on the project, as one of them were a prominent and influential actor, while the other was one of the minors. We also got the perspectives of the two neutral parties and their thoughts on the industry as a whole.

As this was a historical case study, the project happened some years ago. This might affect the informant's thoughts of the project as they might have forgotten certain things, or their memory isn't an accurate description of how things happened.

Bibliography

Abbott, A. P. (1995). Things of Boundaries. *Social Research*, 62(4):857–882.

Abrahamsson, P., Conboy, K., and Wang, X. F. (2009). 'Lots done, more to do': the current state of agile systems development research. *European Journal of Information Systems*, 18(4):281–284.

Armour, J., Awrey, D., Davies, P. L., Enriques, L., Gordon, J. N., Mayer, C. P., and Payne, J. (2016). *Principles of Financial Regulation*. Oxford University Press.

Arner, D. W., Barberis, J., and Buckley, R. P. (2015). The Evolution of Fintech: A New Post-Crisis Paradigm? *Social Science Research Network*.

Atlassian (2023a). Confluence.

Atlassian (2023b). Jira.

Badampudi, D., Fricker, S. A., and Moreno, A. (2013). *Perspectives on Productivity and Delays in Large-Scale Agile Projects*. Springer Science+Business Media.

Baham, C. and Hirschheim, R. (2021). Issues, challenges, and a proposed theoretical core of agile software development research.

Baker, A. (2014). Transnational technocracy and the macroprudential paradox.

Barroso, M. and Laborda, J. (2022). Digital transformation and the emergence of the fintech sector: Systematic literature review.

Bass, J. M. (2019). *Future Trends in Agile at Scale: A Summary of the 7th International Workshop on Large-Scale Agile Development*. Springer Science+Business Media.

Batra, D., Xia, W., VanderMeer, D., and Dutta, K. (2010). Balancing Agile and Structured Development Approaches to Successfully Manage Large Distributed Software Projects: A Case Study from the Cruise Line Industry. *Communications of the Association for Information Systems*, 27.

Beck, K. (1999). *Extreme Programming Explained: Embrace Change*.

Beck, K., Beedle, M. J., Van Bennekum, A., Cockburn, A., Cunningham, L., Fowler, M. M., and Thomas, D. (2001). Manifesto for agile software development. Technical report.

Bick, S., Spohrer, K., Hoda, R., Scheerer, A., and Heinzl, A. (2018). Coordination Challenges in Large-Scale Software Development: A Case Study of Planning Misalignment in Hybrid Settings. *IEEE Transactions on Software Engineering*, 44(10):932–950.

Bjørnson, F. O., Wijnmaalen, J., Stettina, C. J., and Dingsøyr, T. (2018). *Inter-team Coordination in Large-Scale Agile Development: A Case Study of Three Enabling Mechanisms*. Springer Science+Business Media.

Boehm, B. O. and Turner, R. C. (2005). Management Challenges to Implementing Agile Processes in Traditional Development Organizations. *IEEE Software*, 22(5):30–39.

Boland, D. (2004). Transitioning from a co-located to a globally-distributed software development team: a case study at Analog Devices Inc.

Brown, S. (2018). The C4 Model for Software Architecture.

Cawley, O., Wang, X. F., and Richardson, I. (2010). *Lean/Agile Software Development Methodologies in Regulated Environments – State of the Art*. Springer Nature.

Ceps (2019). Study on the costs of compliance for the financial sector.

digital.ai (2022). 16th state of agile report.

Dikert, K., Paasivaara, M., and Lassenius, C. (2016). Challenges and success factors for large-scale agile transformations: A systematic literature review. *Journal of Systems and Software*, 119:87–108.

Dingsøyr, T., Bjørnson, F. O., Schrof, J., and Sporsem, T. (2022). A longitudinal explanatory case study of coordination in a very large development programme: the impact of transitioning from a first- to a second-generation large-scale agile development method. *Empirical Software Engineering*, 28(1).

Dingsøyr, T., Falessi, D., and Power, K. (2019). Agile Development at Scale: The Next Frontier. *IEEE Software*, 36(2):30–38.

Dingsøyr, T., Fægri, T. E., and Itkonen, J. (2014). What Is Large in Large-Scale? A Taxonomy of Scale for Agile Software Development. *Product-Focused Software Process Improvement*, 8892.

Dingsøyr, T. and Moe, N. B. (2013). Research challenges in large-scale agile software development. *ACM Sigsoft Software Engineering Notes*, 38(5):38–39.

Dingsøyr, T., Moe, N. B., Fægri, T. E., and Seim, E. A. (2018). Exploring software development at the very large-scale: a revelatory case study and research agenda for agile method adaptation. *Empirical Software Engineering*, 23(1):490–520.

Dingsøyr, T., Nerur, S., Balijepally, V., and Moe, N. (2012). A decade of agile methodologies: towards explaining agile software development.

Dybå, T. and Dingsøyr, T. (2008). Empirical studies of agile software development: A systematic review. *Information Software Technology*, 50(9-10):833–859.

Economist, T. (2017a). Remaking american financial regulation.

Economist, T. (2017b). The right way to redo dodd-frank.

Edison, H., Wang, X. F., and Conboy, K. (2021). Comparing Methods for Large-Scale Agile Software Development: A Systematic Literature Review. *IEEE Transactions on Software Engineering*, 48(8):2709–2731.

EU (2015). Payment services directive 2.

EU (2023). Overview of financial services legislation.

Fitzgerald, B., Stol, K.-J., O'Sullivan, R., and O'Brian, D. (2013). Scaling agile methods to regulated environments: An industry case study.

Flowable (2023). Confluence.

Flyvbjerg, B. (2006). Five Misunderstandings About Case-Study Research.

Guellec, D. and Paunov, C. (2017). Digital Innovation and the Distribution of Income. Technical report.

Hanssen, G. K., Brataas, G., and Martini, A. (2019). Identifying Scalability Debt in Open Systems.

Hoda, R., Salleh, N., and Grundy, J. (2018). The rise and evolution of agile software development.

Juuse, E., Raudla, R., Cepilovs, A., and Mikheeva, O. (2019). The Europeanization of financial regulation and supervision on the Baltic–Nordic axis: the perspective of national bureaucracies. *Journal of Baltic Studies*.

Jönsson, H., Larsson, S., and Punnekkat, S. (2012). Agile Practices in Regulated Railway Software Development.

Krasonikolakis, I., Tsarbopoulos, M., and Eng, T.-Y. (2020). Are incumbent banks bygones in the face of digital transformation? *Journal of general management*, 46(1):60–69.

Linux-Foundation (2023). Openapi.

Lovdata (2019). Lov om endringer i innskuddspensjonsloven.

Mathiassen, L. (2017). Designing Engaged Scholarship: From Real-World Problems to Research Publications.

McCaffery, F., Trektere, K., and Özcan Top, (2016). *Agile – Is it Suitable for Medical Device Software Development?* Springer Science+Business Media.

Merisalo-Rantanen, H., Tuunanen, T., and Rossi, M. (2005). Is Extreme Programming Just Old Wine in New Bottles. *Journal of Database Management*, 16(4):41–61.

Miro (2023). Miro app.

Oates, B. J., Griffiths, M., and McLean, R. (2022). *Researching Information Systems and Computing*. SAGE.

Paasivaara, M., Behm, B., Lassenius, C., and Hallikainen, M. (2018). Large-scale agile transformation at Ericsson: a case study. *Empirical Software Engineering*, 23(5):2550–2596.

Paul, M. (2011). On Empirical Research Into Scrum Adoption.

Poth, A., Jacobsen, J., and Riel, A. (2020). Systematic Agile Development in Regulated Environments.

Preda, A. (2009). Framing finance: the boundaries of markets and modern capitalism. *Choice Reviews Online*, 47(04):47–2123.

Pries-Heje, L. and Pries-Heje, J. (2011). Why Scrum Works: A Case Study from an Agile Distributed Project in Denmark and India.

Quaglia, L. (2013). Financial regulation and supervision in the european union after the crisis.

Regjeringen (2022). Acts and regulations.

Reifer, D., Maurer, F., and Erdogmus, H. (2003). Scaling agile methods. *IEEE Software*, 20(4):12–14.

Reitz, A., Jentsch, C., and Beimborn, D. (2018). How to decompress the Pressure - The moderating Effect of IT Flexibility on the negative Impact of Governmental Pressure on Business Agility.

Schwaber, K. (2004). *Agile Project Management with Scrum*.

Scott, E., Milani, F., Kilu, E., and Pfahl, D. (2021). Enhancing agile software development in the banking sector—A comprehensive case study at LHV. *Journal of software*, 33(7).

Sharp, H. and Robinson, H. P. C. (2004). An Ethnographic Study of XP Practice. *Empirical Software Engineering*, 9(4):353–375.

SIKT (2023). Notification form for personal data.

SteelEye (2022). The state of Financial Services Compliance. Technical report.

Steghöfer, J.-P., Knauss, E., Horkoff, J., and Wohlrab, R. (2019). *Challenges of Scaled Agile for Safety-Critical Systems*. Springer Science+Business Media.

Stirbu, V. and Mikkonen, T. (2020). CompliancePal: A Tool for Supporting Practical Agile and Regulatory-Compliant Development of Medical Software.

Stol, K.-J., Goedicke, M., and Jacobson, I. (2016). Introduction to the special section—General Theories of Software Engineering: New advances and implications for research.

Stotts, P. D., Williams, L., Nagappan, N., Baheti, P., Jen, D., and Jackson, A. (2003). *Virtual Teaming: Experiments and Experiences with Distributed Pair Programming*. Springer Science+Business Media.

Strode, D. E. (2005). The agile methods : an analytical comparison of five agile methods and an investigation of their target environment : a thesis presented in partial fulfillment of the requirements for the degree of Master of Information Sciences in Information Systems at Massey University, Palmerston North, New Zealand.

Strode, D. E., Huff, S. L., Hope, B. G., and Link, S. (2012). Coordination in co-located agile software development projects. *Journal of Systems and Software*, 85(6):1222–1238.

Surendra, N. C. and Nazir, S. (2019). Creating "informating" systems using Agile development practices: an action research study. *European Journal of Information Systems*, 28(5):549–565.

West, D., Grant, T., and D'silva, D. (2010). Agile development: Mainstream adoption has changed agility. Technical report.

WorldBank (2020). How Regulators Respond To FinTech : Evaluating the Different Approaches – Sandboxes and Beyond. Technical report.

Yap, M. (2005). Follow the sun: distributed extreme programming development.

Appendices

A Interview guide

- Overall
  - Name
  - Role
  - How long have you been working here?
  - What roles have you had?
    - What was your role in the EPK project
- The EPK project
  - How was it to create a product with multiple customers?
  - How did you agree to create this solution?
  - How was it to share the role of product owner with others?
  - How was the tradeoff between ensuring the usability of the product and complying to the law?
  - How was the project structured?
  - Who was responsible for maintaining the solution?
  - Did you consider the possibility that new laws would affect the solution in the future?
  - How were the developers involved in the different parts of the project?
  - Where in the process were legal experts involved?
  - How was this involvement experienced?
  - When in the process is the finished solution controlled against the legal demands?
  - How much focus was it on producing documentation during development?
    - Was this amount of focus different from other projects you have been involved in?
    - Where there any demands for documentation from the customer?
  - Where any specific framework for development used or was it customized?
    - What was considered if customized?
  - Was there any continuous deployment in this project? If so, how was the continuous testing performed?
  - Agile development
    - What common meetings were arranged?
    - How was the communication between teams?
    - How were the teams organized?
  - What was the biggest challenge in your role?
  - How did you handle dependencies to other teams?

B Information letter

Vil du delta i forskningsprosjektet

Programvareutvikling i regulerte omgivelser i finans

Dette er et spørsmål til deg om å delta i et forskningsprosjekt hvor formålet er å undersøke hvordan reguleringer påvirker programvareutviklingen hos en finansaktør. I dette skrivet gir vi deg informasjon om målene for prosjektet og hva deltakelse vil innebære for deg.

Formål
En masteroppgave i form av en case studie som tar for seg hvordan en finansinstitusjon håndterer kontinuerlig endring av reguleringer i forbindelse med programvareutvikling. Formålet er å finne evidens på om regulerte omgivelser påvirker hvordan et utviklingsprosjekt blir lagt opp og gjennomført. Dataene vil bli hentet inn gjennom personlige intervju, spørreundersøkelser og tilstedeværelse hos case-organisasjonen.

Hvem er ansvarlig for forskningsprosjektet?
NTNU er ansvarlig for prosjektet. Prosjektet utføres i samarbeid med Storebrand.

Hvorfor får du spørsmål om å delta?
Vi ønsker å snakke med deg basert på din rolle i Storebrand, eller din tilknytning til egen pensjons konto(EPK) prosjektet. Vi har fått din kontaktinformasjon gjennom Roar Klokk Morset eller Kristin Normann

Hva innebærer det for deg å delta?
Hvis du velger å delta i dette prosjektet vil du enten motta et spørreskjema som tar ca 30 minutter å fylle ut, eller bli innkalt til et intervju som tar ca en time. Svarene fra et spørreskjema vil bli registrert elektronisk, og et intervju vil bli tatt opp og transkribert.

Det er frivillig å delta
Det er frivillig å delta i prosjektet. Hvis du velger å delta, kan du når som helst trekke samtykket tilbake uten å oppgi noen grunn. Alle dine personopplysninger vil da bli slettet. Det vil ikke ha noen negative konsekvenser for deg hvis du ikke vil delta eller senere velger å trekke deg.

Ditt personvern – hvordan vi oppbevarer og bruker dine opplysninger
Vi vil bare bruke opplysningene om deg til formålene vi har fortalt om i dette skrivet. Vi behandler opplysningene konfidensielt og i samsvar med personvernregelverket.

Underveis i prosjektet er det kun masterstudentene Andreas Saltom Rikheim og Eskil Helgesen Schjølberg, og veilederen deres Marius Mikaelsen som vil ha tilgang på dataen. Dataen vil lagres på krypterte forskningsservere for å sikre personopplysningene. Godkjenning av denne oppbevaringen har man fått via Sikt. Navnet og kontaktopplysningene dine vil jeg erstatte med en kode som lagres på egen navneliste adskilt fra øvrige data

Som deltaker i studien vil du ikke kunne gjenkjennes i publikasjonen.

Hva skjer med personopplysningene dine når forskningsprosjektet avsluttes?
Prosjektet vil etter planen avsluttes 30.06.2023. Etter prosjektet vil lydopptak slettes, og alle personopplysninger vil anonymiseres. Datamaterialet vil lagres for etterprøvbarhet. Det vil lagres på NTNU sin infrastruktur på ubestemt tid.

Hva gir oss rett til å behandle personopplysninger om deg?
Vi behandler opplysninger om deg basert på ditt samtykke.

På oppdrag fra NTNU har Sikt – Kunnskapssektorens tjenesteleverandør vurdert at behandlingen av personopplysninger i dette prosjektet er i samsvar med personvernregelverket.

Dine rettigheter
Så lenge du kan identifiseres i datamaterialet, har du rett til:

- innsyn i hvilke opplysninger vi behandler om deg, og å få utlevert en kopi av opplysningene
- å få rettet opplysninger om deg som er feil eller misvisende
- å få slettet personopplysninger om deg
- å sende klage til Datatilsynet om behandlingen av dine personopplysninger

Hvis du har spørsmål til studien, eller ønsker å vite mer om eller benytte deg av dine rettigheter, ta kontakt med:

- *NTNU* ved Marius Mikalsen, mariusmi@ntnu.no
- Vårt personvernombud: Thomas Helgesen, thomas.helgesen@ntnu.no

Hvis du har spørsmål knyttet til vurderingen som er gjort av personverntjenestene fra Sikt, kan du ta kontakt via:

- Epost: personverntjenester@sikt.no eller telefon: 73 98 40 40.

Med vennlig hilsen

Marius Mikaelsen (Forsker/veileder)

Andreas Saltom Rikheim og Eskil Schjølberg (Masterstudenter)

Samtykkeerklæring

Jeg har mottatt og forstått informasjon om prosjektet "Programvareutvikling i regulerte omgivelser i finans" , og har fått anledning til å stille spørsmål. Jeg samtykker til:

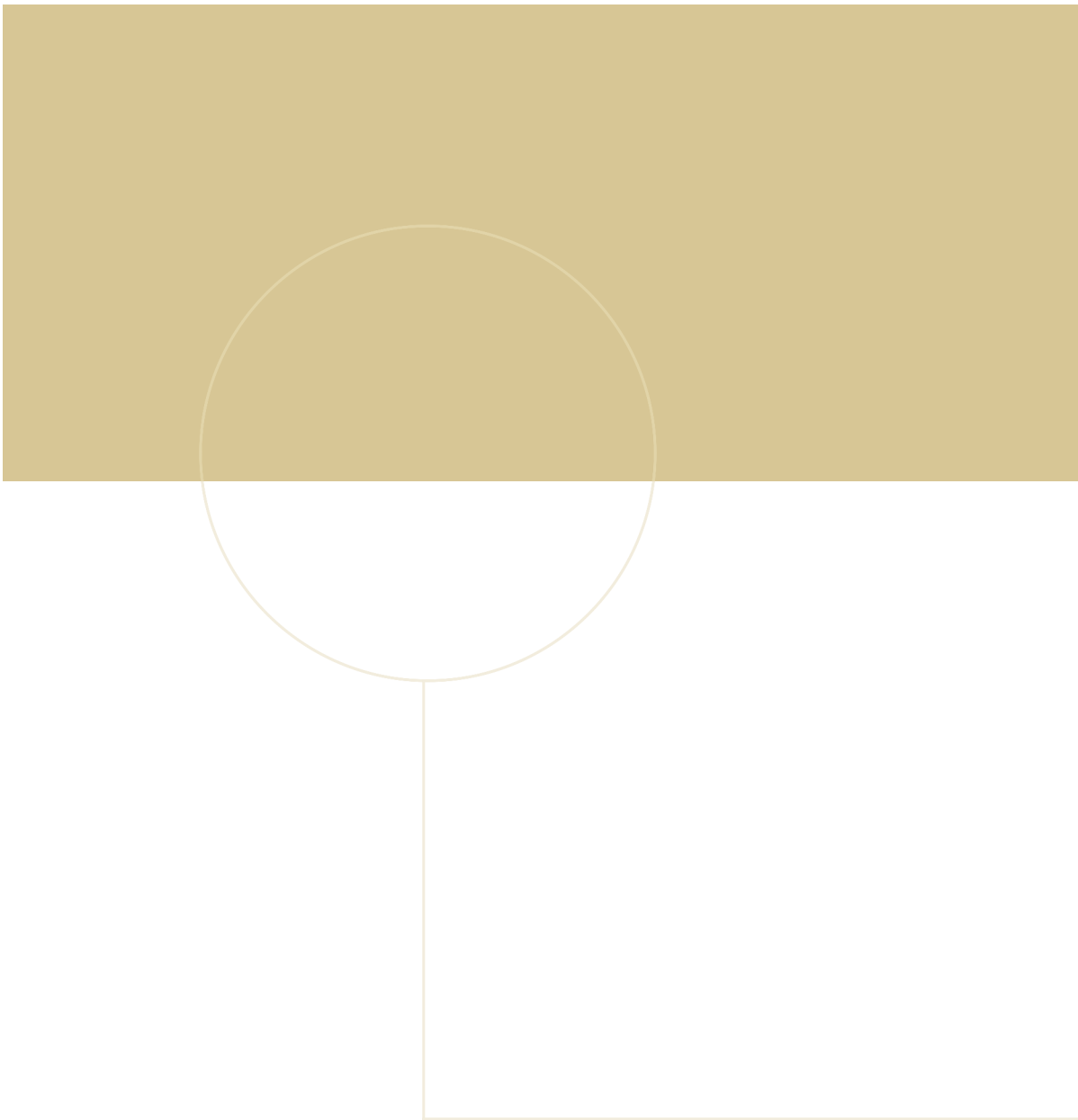
- å delta i intervju [ ]
- å delta i spørreskjema [ ]

Jeg samtykker til at mine opplysninger behandles frem til prosjektet er avsluttet

---
(Signert av prosjektdeltaker, dato)

C Presence at case organization

Dag	Tidspunkt	Tid	
Mandag 23.jan	09:00-13:00		4
Onsdag 25.jan	09:00-15:00		6
Fredag 27.jan	09:00-14:00		5
Fredag 3.feb	09:00-15:00		6
Onsdag 8.feb	09:00-15:00		6
Fredag 10.feb	09:00-14:00		5
Onsdag 15.feb	09:00-15:00		6
Fredag 17.feb	08:00-14:00		6
Onsdag 22.feb	09:00-15:00		6
Fredag 24.feb	09:00-15:00		6
Onsdag 1.mar	09:00-15:00		6
Fredag 3.mar	09:00-15:00		6
Onsdag 8.mar	09:00-15:00		6
Fredag 10.mar	09:00-15:00		6
Onsdag 15.mar	09:00-15:00		6
fredag 17.mar	09:00-15:00		6
Onsdag 22.mar	09:00-15:00		6
Fredag 24.mar	09:00-15:00		6
Onsdag 29.mar	09:00-15:00		6
Fredag 14.apr	09:00-15:00		6
Onsdag 19.apr	09:00-15:00		6
Fredag 21.apr	09:00-15:00		6
Onsdag 26.apr	09:00-15:00		6
Fredag 28.apr	09:00-15:00		6
Onsdag 3. mai	09:00-15:00		6
Fredag 5 .mai	09:00-15:00		6
Onsdag 10.mai	09:00-15:00		6
Fredag 12.mai	09:00-15:00		6
Fredag 26.mai	09:00-12:00		3



NTNU

Norwegian University of
Science and Technology