A dynamic perspective on the large-scale agile implementation in a high tech developing firm
A Case Study

Corstjens, F.P. (Stijn)

Award date:
2023

Eindhoven University of Technology

Innovation Management

A dynamic perspective on the large-scale agile implementation in a high tech developing firm: A Case Study

Author: F.P. Corstjens

Student №: 1633155

Reviewing Assessors:
Dr. Bob Walrave
Dr. Alex Alblas
Dr. Gizem Korpeoglu

May 8, 2023

Contents

1 Abstract

The increasingly turbulent market dynamics and the inability to respond to them have driven hardware-developing companies to adopt principles of the agile project management (APM) approach. Since these hardware-developing companies also require the benefits of the traditional project management approach, they strive to combine the benefits of both project management approaches and execute a so-called hybrid project management (HPM) approach.

However, the execution of such an HPM approach turned out to be challenging since it would require the smooth integration of both APM and TPM approaches. While established APM methodologies provide processes and roles, they do not provide guidance on how to integrate these processes and roles with existing organizational structures applying a TRM approach. In addition, since the APM approach was initially designed for the software industry, it lacks guidance on how it should be practiced in companies operating in the hardware industry. Despite these implementation challenges, the hardware industry is motivated to increase its project planning flexibility and adaptability to deal with the increasingly turbulent market dynamics.

To provide practitioners with insight into the implementation of an APM approach within a company developing hardware products using a TPM approach, this study set out to understand the dynamics of such an implementation. The mechanisms underlying these dynamics were captured and theorized into a causal loop diagram by conducting a case study. Based on a variety of data, including interviews with multiple organizational roles, company documentation, and participatory observation, this study has identified four dynamic mechanisms that describe the implementation of a large-scale agile approach into a high-tech company developing its high-tech products using a traditional V-model approach.

The findings show that the desired outcome of this APM implementation is mitigated by multiple unintended consequences. These unintended consequences include a hybrid project management loop, a conflicting planning requirements loop, a focus issue loop, and the effect of the high-tech environment. These dynamics are illustrated in a causal loop diagram that can provide guidance for practitioners on what intended consequences to expect during this implementation and how to mitigate these consequences.

2 Introduction

For decades, projects have been managed using a sequential and plan-driven approach (Gemino et al., 2021; Mule et al., 2020). This traditional project management approach (TPM) is defined by linear and predictable project planning practices designed to achieve a well-understood, achievable set of objectives (Gemino et al., 2021; Špundak, 2014). This approach aims to determine the project's scope, cost, and time at the beginning of the project (Gubinelli et al., 2019). Well-known methodologies for TPM are the waterfall and V-model method (Bricogne et al., 2016; Kasauli et al., 2020; Mule et al., 2020). Since TPM was designed for long and complex projects in a stable and predictable environment, the project approach aimed to minimize changes and uncertainty during the project phases by identifying potential future problems (Boehm and Turner, 2004; Vinekar et al., 2006).

While TPM was designed to minimize project uncertainty, its strict upfront planning would diminish the possibility of changing certain aspects of the project during its execution. As markets become more turbulent due to technological and socio-economic transformations, projects become more complex, and meeting project requirements is becoming more difficult (Eklund et al., 2014; Elia et al., 2021). If certain project aspects would require change due to particular market dynamics, then the inability to execute this change would impact project success (Böhmer et al., 2017). As a consequence, the ability to be adaptable during project execution becomes especially important for companies operating in such turbulent market conditions (Costantini et al., 2021).

The inability to act on changing market dynamics during product development and the increased product complexity were reasons for a group of software practitioners to develop the agile project management approach (APM) (Beck et al., 2001). This APM approach is characterized by iterative, flexible, transparent development and incremental product delivery (Špundak, 2014; Zasa et al., 2020). Combined with constant customer involvement, these characteristics enabled software developers to constantly adapt their projects to the market (Böhmer et al., 2017). While these APM methodologies were initially designed for small intra-team software practices, their success led to the introduction and application of large-scale agile frameworks and the adoption of APM in hardware environments.

To enable APM for large projects and organizations, large-scale agile methodologies such as the Scaled Agile Framework (SAFe) and Large Scale Scrum (LeSS) were introduced to manage concerns related to multiple teams, inter-team coordination, and customer involvement. As the adoption of these frameworks became more popular in this industry, multiple scholars investigated the implementation of these frameworks (Bick et al., 2017; Conboy and Carroll, 2019; Dikert et al., 2016;

Ebert and Paasivaara, 2017; Putta et al., 2018; Uludağ et al., 2022). As a result, the research maturity provided software practitioners with valuable insights and practical implications for adopting large-scale agile frameworks.

As the increased turbulent market dynamics also started to become a challenge for the hardware industry, hardware-developing organizations started to adopt agile methodologies. The ability to be the first on the market is an important factor in determining the competitive advantage of hardware-developing organizations. Therefore, it is crucial for these companies to learn as quickly as possible to stimulate the rapid development of new products. This becomes especially challenging for high-tech companies developing high-tech products as they are mostly operating in dynamic markets and are prone to long and complicated development processes (Alblas and Notten, 2021). To facilitate this rapid development, agile project management approaches in the hardware industry were considered.

The increased adoption of the agile project management approach within the hardware industry attracted wide scholarly attention (Eklund et al., 2014; Garzaniti et al., 2019a; Schmidt, 2018). Despite the promising success stories from the software industry, it became evident that implementing the APM approach in the hardware industry introduced additional challenges to the ones encountered within purely software development. These additional challenges could largely be attributed to the physical nature of hardware products (Atzberger and Paetzold, 2019; Drutchas and Eppinger, 2022; Ovesen, 2012).

Besides the additional implementation challenges in the hardware industry, the APM approach has also been criticized for its poor resource planning and lack of accuracy in time planning (Reiff and Schlegel, 2022). As these characteristics are critical for hardware development, which is characterized by long lead times, hardware companies started to apply practices of both the TPM and APM approach (Eklund and Berger, 2017). Since the goals are often unclear at the start of a project, this so-called hybrid project management (HPM) approach could use TPM principles to concretize the objective steps. This would allow for planning the 'big picture', even with long-term planning times and milestones often required for complex products (Reiff and Schlegel, 2022; Thesing et al., 2021). On the other hand, implementing APM principles in HPM would enable transparency and flexibility (Bick et al., 2017). As changes would occur during the project, implementing APM principles in the HPM approach would allow for reacting more flexibly to these changes than TPM (Copola Azenha et al., 2021).

Although combining the benefits of TPM and APM principles into the HPM approach seems

promising, tension exists between both approaches (Walrave et al., 2022; Zasa et al., 2020). Since both approaches have a different philosophy, they have contrasting approaches toward matters such as planning & execution, communication, hierarchy, and consequently, the perception of how a project should be executed (Cooper and Sommer, 2016; Zasa et al., 2020). This tension raised the question of how the combination of these contrasting project management approaches could be managed effectively (Kasauli et al., 2020).

To find a way to effectively manage the execution of both an APM and TPM approach simultaneously, multiple scholars have investigated organizations implementing a hybrid project management approach and addressed their challenges, successes, and best practices (Cooper and Sommer, 2016; Garzaniti et al., 2019b; Reiff and Schlegel, 2022; Zasa et al., 2020). Despite giving insight into these factors, research on the dynamics underlying the integration of such an HPM approach seems to be lacking. Investigating and describing these dynamics would enable practitioners to better understand the mechanisms accompanying such an HPM integration and therefore facilitate its implementation process. To address this gap, the following research question is derived:

What mechanisms are underlying the implementation of an agile project management approach within an organization developing high-tech products using a traditional project management approach and how do these mechanisms facilitate or hinder the task management of engineers?

To answer this question, an in-depth case study is conducted on the implementation of a large-scale agile framework in a company developing high-tech products. Since this company is applying a traditional V-model for product development, it is applying a hybrid project management approach. To theorize the dynamic behavior of the large-scale agile implementation in this high-tech developing company, a causal loop diagram is developed from qualitative data (Sterman, 2002). The model describes the effect of implementing large-scale agile on the task management of engineers within an organization that develops high-tech products. Furthermore, the causal loop diagram shows how this implementation is prone to unintended dynamics that undermine the effectiveness of the large-scale agile implementation on the task management of engineers.

Table 1 summarizes multiple studies that researched APM and HPM for physical products. In this research, physical or hardware products refer to products that do not consist of software solemnly and thus have a physical nature. This can be either a physical product with or without integrated software elements.

As shown in table 1, the research context of these exemplar studies differ in multiple aspects. While the illustrated studies in Table 1 have focused their research on organizations that implement APM for the development of physical products, thirteen studies actually researched the application of

APM for the development of the hardware part of the physical product. The remaining studies focused mainly on the APM implementation of the software part of the organization and studied how they interacted with the hardware part that was executing a traditional project management approach. Other studies in this hybrid project management context focused on the interplay between the development teams executing an agile project management approach and the traditional higher-level project management approach.

Four conclusions can be drawn from table 1: first, only two studies researched the use of APM for hardware development in large-scale projects. Second, while some studies addressed the high product complexity of the product developed, no studies have investigated the implementation of APM for large-scale projects developing high-tech products. Third, only one other study researched the use of APM for multiple projects simultaneously. Fourth, while one other study developed a dynamic model to describe the interplay between a software and hardware product development team with contrasting project management approaches, other studies have solely focused on the identification of challenges, benefits, and best practices. This indicates that the literature lacks guidance for organizations that want to implement APM for large-scale projects to develop high-tech products.

This study contributes to the literature in multiple ways. First, while some studies mentioned the high complexity of the physical products in the particular case setting they were researching (Böhmer et al., 2017; Garzaniti et al., 2019b; Schmidt, 2018) and other studies addressed the impact of product complexity on the implementation of agile (Atzberger and Paetzold, 2019; Zasa et al., 2020), research on the effects of high-tech products consisting of multiple physical sub-modules on the implementation of agile project management is missing. By researching a high-tech company that is implementing APM for its activities, this study contributes to the literature.

Second, most studies in the agile hardware development literature have focused on case settings that applied the APM approach to a single project (Drutchas and Eppinger, 2022; Eklund et al., 2014; Garzaniti et al., 2019a; Weichbroth, 2022). Since physical product development is prone to waiting times and hardware-related factors, engineers are mostly working on multiple projects simultaneously to increase efficiency (Cooper and Sommer, 2016). While Ovesen (2012) investigated three case settings that applied an APM approach for multiple projects simultaneously, further research on this subject is limited.

Lastly, this study contributes to the hybrid project management literature by providing a dynamic understanding of the implementation of a large-scale agile approach within a company developing

high-tech products using a traditional project management approach. As studies have investigated companies in a hybrid project management context and identified challenges, benefits, and best practices, the mechanisms underlying such an implementation seem to be lacking.

Table 1: Exemplar studies

Authors (year)	Industry	Research design	Organization developing physical products	Hardware development using APM	Hybrid project management context	Use of agile in large-scale projects	High product compexity	Use of agile for multiple projects simultaneously	Developing dynamic model
Drutchas and Eppinger (2022)	Wearable electronics, automotive, consumer electronics and more.	Multiple case study	✓	✓					
Garzaniti et al. (2019a)	Space industry	Case study	✓	✓					
Atzberger and Paetzold (2019)	n.a.	Embedded design study	✓	✓			✓		
Böhmer et al. (2017)	Automotive industry	Case study	✓	✓			✓		
Schmidt (2018)	Mechanical engineering, automotive industry	Cross-sectional study	✓	✓			✓		
Ovesen (2012)	Healthcare, building industry,infrastructure, audiological equipment, and more	Multiple case study	✓	✓				✓	
Eklund et al. (2014)	Automotive industry, circulation pump industry	Multiple case study	✓	✓		✓			
Cooper and Sommer (2016)	Toy industry	Case study	✓	✓	✓				
Cooper and Sommer (2018a)	Building industry, Toy industry	Multiple case study	✓	✓	✓				
Zasa et al. (2020)	n.a.	Multiple case study	✓	✓	✓	✓			
Weichbroth (2022)	Mobile process management system	Case study	✓	✓	✓				
Sommer et al. (2013)	Industrial manufacturing industry	Multiple case study	✓	✓	✓				
Garzaniti et al. (2019b)	Space industry	Case study	✓	✓	✓		✓		
Kasauli et al. (2020)	Automotive industry	Multiple case study	✓		✓	✓			
Eklund and Berger (2017)	Mechatronic industry	Comparative case study	✓		✓	✓			
Kasauli et al. (2021)	Telecom, automotive, medical industry	Multiple case study	✓		✓	✓	✓		
Walrave et al. (2022)	Automotive industry	Case study	✓		✓				✓
This study	**Semiconductor industry**	**Case study**	✓	✓	✓	✓	✓	✓	✓

3 Theoretical Background

3.1 Agile project management for software development

As markets become more turbulent due to technological and socio-economic transformations, projects become more complex, and meeting project requirements is becoming more difficult (Eklund et al., 2014; Elia et al., 2021; Walrave et al., 2022). To deal with this increasing market turbulence, a group of software practitioners developed the agile project management approach (Beck et al., 2001). This APM was characterized by iterative, flexible, transparent development and incremental product delivery (Cooper and Sommer, 2016; Zasa et al., 2020). Combined with constant customer involvement, these characteristics enabled software developers to constantly adapt their projects to the market and cope with market uncertainties (Atzberger and Paetzold, 2019; Böhmer et al., 2017).

As most agile methodologies were designed for intra-team practices, the software industry lacked coordination and communication mechanisms for large-scale projects and organizations. In response, consultants and software practitioners introduced multiple scaling agile frameworks such as the Scaled Agile Framework (SAFe), Large Scale Scrum (LeSS), and Disciplined Agile Delivery (DAD) (Uludağ et al., 2022). These frameworks introduce workflow patterns to manage multiple teams, inter-team coordination, and customer involvement (Uludağ et al., 2022). Among these frameworks, SAFe is the most applied scaling framework in large enterprises (Digital.ai, 2022).

Since SAFe builds on agile principles such as Scrum, Extreme programming, Kanban and integrates lean best practices, it enables scalability at the team, program, large solution, and portfolio levels for large organizations and projects (Koehnemann, 2022; Leffingwell, 2016). As the team level consists of agile teams, the concept of an Agile release train (ART) is introduced to scale a large number of teams at a program level. This ART then follows an 8 to 12 weeks lasting planning interval (PI) which is developed during the PI planning, Within this PI, the work is managed by the teams in multiple sprints to create a cadence-based time box. Furthermore, SAFe introduces the scrum master and product owner role to enable the teams, Moreover, SAFe introduces processes such as the PO sync, scrum of scrums, and sprint review to enable transparency and alignment within the ART (Leffingwell, 2016).

As the adoption of these large-scale agile frameworks became more popular in this industry, multiple scholars investigated the implementation of these frameworks (Bick et al., 2017; Conboy and Carroll, 2019; Dikert et al., 2016; Ebert and Paasivaara, 2017; Putta et al., 2018; Uludağ et al., 2022). As a result, the literature stream on large-scale agile for software matured and provided software practitioners with valuable insights and practical implications for adopting scaled agile frameworks.

The actual benefits of implementing large-scale agile in software were improved transparency, increase productivity, and improved ability to respond to changes (Putta et al., 2018; Schmidt, 2018).

Despite the benefits, implementing large-scale agile methodologies was found to be complicated and prone to multiple challenges. First, the large-scale agile methodologies would increase the need for inter-team collaboration (Edison et al., 2021; Uludağ et al., 2022). This collaboration required the work coordination of multiple agile teams. According to Dikert et al. (2016), challenges arose when the agile teams had to work with other teams from the surrounding organization. While the team level had become more flexible due to the agile implementation, the surrounding organization was not responsive enough. The reasoning for this was that agile did not remove dependencies, which made managing the development difficult (Dikert et al., 2016).

Second, Edison et al. (2021) identified organizational challenges when implementing large-scale agile. The definition of new roles and their responsibilities were not always clearly defined. This was especially the case in organizations that created their agile teams within a traditional project management organization. In addition, the newly introduced roles could lead to a complex organizational setup that would cause a large number of information handovers. Furthermore, Dikert et al. (2016) found that in some cases the initial organization had specialized knowledge in silos that caused problems in implementing agile. The use of scrum in these cases showed internal segmentation in which teams were operating with different priorities and agendas.

Third, a lack of investment in training or coaching was a challenge mentioned by Dikert et al. (2016). The reluctance of management to invest in training for the teams and provide coaching on agile in the real work environment resulted in the agile practices being executed improperly. Moreover, there were cases in which there was a too high workload which prevented the overloaded people to be able to change their behavior towards the new way of working.

3.2 Agile project management for hardware development

To deal with increasingly turbulent market conditions, companies in the hardware industry started to adopt the APM principles from the software industry (Böhmer et al., 2017). According to the research of Schmidt (2018), actual benefits as a result of implementing large-scale agile in hardware development are improved communication, reduced reaction time to changes, increased transparency, increased flexibility, increased project-related commitment, and increased project effectiveness. Since agile project management originates from the software industry, agile methods and principles were designed based on software characteristics. As multiple scholars researched cases of agile implementation in a hardware environment, it became evident that implementing

the APM principles in these companies introduced additional challenges to the ones encountered within purely software development. These additional challenges could largely be attributed to the physical nature of hardware products, referred to as the constraint of physicality (Atzberger and Paetzold, 2019; Drutchas and Eppinger, 2022; Garzaniti et al., 2019a; Ovesen, 2012).

Besides this constraint of physicality hindering frequent prototyping due to longer production times (Schmidt, 2018), other challenges related to the implementation of agile in hardware development were found. Ovesen (2012) was one of the first to research the implementation of agile for hardware development when he researched seven physical product-developing companies. In this research, five general challenges were identified: constraint of physicality, the integration of agile methods with the already existing project management structure in place (paradigm perplexity), motivating mechanical and industrial engineers to work with agile (designers dissent), constructing the teams into agile development teams due to required specific knowledge (team distribution dilemma), and changing company culture (education and maturation). Drutchas and Eppinger (2022) further added the challenges of sprint length and backlog decomposition to this.

Atzberger and Paetzold (2019) mentioned that complexity in the field of hardware development has a much broader range than in software. To describe this, they introduced a 3-axed graph with on the axis the product architecture (ranging from a few to several components from all domains), the company size (ranging from a small to large organization), and size of the project team (ranging from one small to several large teams). Based on this 3-axed graph they argued that the optimal conditions for implementing agile in hardware development were for products with a low product architecture, small company size and small team size. They further argued that highly sophisticated and therefore complex physical products are predominantly developed by large-size companies due to the necessity of incorporating different domains. They then concluded that adaptions to agile in the field of hardware development are inevitable and therefore principles and practices are necessary for these companies.

Moreover, while APM is among other things designed to promote focus, Ovesen (2012) found that some case studies were experiencing challenges related to 'maintaining necessary focus'. This was caused by the agile teams having to 'fire fight' in the operation of the production and due to the teams participating in multiple projects simultaneously. According to Wheelwright and Clark (1992), the participation in two or more projects significantly reduced the time spent on value-adding tasks. Monsell (2003) and Tregubov et al. (2017) describe this reduced task efficiency as switching costs.

Furthermore, while Kasauli et al. (2021) researched multiple hardware-developing companies implementing large-scale agile, they identified multiple challenges and practices related to requirements engineering. Zasa et al. (2020) researched multiple organizations in hardware environments and found that the transition of large-scale agile frameworks from the software industry to non-software environments was subject to two problems. First, as non-software environments typically divide their products into independent modules and the development requires interactions among different organizational functions, it challenges the basic premises of agile. Second, large-scale organizations are more likely to have traditional and consolidated project management practices in place that they do not want to abandon completely. When organizations decide to implement agile, they must find the agile practices to coexist with the existing project management practices.

3.3 Hybrid project management

As agile would offer flexibility and transparency, it also introduced challenges. Due to its short-term focus, keeping the focus on the long-term goal could be more difficult. Especially in the case of hardware development which is characterized by its long lead times (Cooper and Sommer, 2018a; Eklund et al., 2014). Additionally, due to the constant re-prioritization of tasks, the overview of the project can become confusing. As a result, the agile approach can contain a certain inaccuracy in time planning and budget scheduling (Reiff and Schlegel, 2022).

To be able to reap the benefits of the agile project management approach while avoiding its challenges, the hybrid project management approach emerged. As this project management approach integrates practices of both the agile project management approach and the traditional project management approach, it aims to combine the benefits of both (Brandl et al., 2018; Eklund et al., 2014). By applying this hybrid project management approach, higher management is able to track the progress and performance of the project from a strategic and long-term perspective, while the execution of this project can be executed flexibly (Zasa et al., 2020).

The promising advantages of the hybrid project management approach led to the development of multiple hybrid methodologies. Reiff and Schlegel (2022) conducted a systematic literature review of articles researching hybrid project management settings and categorized four hybrid methodologies: Water-Scrum-Fall, Waterfall-Agile, Hybrid V-model, and Agile-Stage-Gate. Additionally, they mention that besides these systematic methodologies, companies often cherry-pick individual methods and practices from different methodologies in their project management style. When a company applies the HPM, the traditional approach is often supplemented by the agile approach in practice (Reiff and Schlegel, 2022).

As more companies started to implement hybrid project management approaches, scholars investigated their advantages. One of the advantages mentioned by Cooper (2017) was that the hybrid approach could be applied within the hardware industry, from toys to heavy industrial equipment. Furthermore, a hybrid project management approach could be adopted for projects with long deadlines as it woukd allow for integrating and managing different stakeholders and maintaining strict project planning and control (Copola Azenha et al., 2021). In addition, if changes occurred, the agile component would allow for reacting more flexibly to these changes than a pure traditional project management approach (Brandl et al., 2018; Špundak, 2014). This would enable the organizations to manage new requirements or changing priorities without rescheduling the whole project (Reiff and Schlegel, 2022).

Despite the promising success stories of firms implementing a hybrid project management approach (Cooper and Sommer, 2018a), effective integration of the traditional and agile project management approach in practice remains challenging. Zasa et al. (2020) investigated companies implementing a hybrid project management approach within non-software companies and found that it is not easy to adopt. They identified transformation challenges that arise due to the required alignment between the project teams, organizational objectives, and project implementation. Furthermore, Kasauli et al. (2021) investigated the challenges related to the interplay of agile and plan-driven methods within the automotive industry. They found challenges in communication between the newly established agile interface and the traditional plan-driven interface, awareness of development teams about high-level requirements, and ability to change requirements. Challenges related to communication and coordination were also encountered by Garzaniti et al. (2019b) who researched a hybrid project management approach for product development in the space sector. Moreover, Bick et al. (2017) explored coordination challenges in a hybrid setting and found effective coordination between agile teams and traditional inter-team levels to be prevented by misaligned planning activities of specification, prioritization estimation, and allocation.

While theoretical prescriptive hybrid project management models have been proposed (Conforto and Amaral, 2016; Cooper and Sommer, 2016) and multiple studies have defined hybrid project management challenges and successes (Reiff and Schlegel, 2022; Zasa et al., 2020), an actual understanding of how and why hybrid project management approaches fail is lacking (Copola Azenha et al., 2021; Reiff and Schlegel, 2022). This can partly be attributed to the many possibilities to apply a hybrid project management approach. Since the integration of HPM has an intrinsic need to adapt to the needs of the environment, combining agile and traditional approaches is a trade-off between the need for agility and project control. As a consequence of these adaptions to specific environments, HPM approaches lack standardization (Copola Azenha et al., 2021).

While an effective integration of an HPM approach requires overcoming the challenges related to co-existence of the APM and TPM approach, the solution remains an open issue, especially in hardware environments (Zasa et al., 2020). Therefore, to provide insights and guidance on the combined use of agile and traditional project management approaches, research should focus on the implementation dynamics of APM with TPM in the hardware industry.

4 Research Methodology

In response to the lack of research on the dynamics underlying the implementation of an agile project management approach in an environment that is also executing a traditional project management approach, this study sets out to investigate these dynamics. More specifically, this study aims to understand the mechanisms underlying the dynamics encountered during the implementation of a large-scale agile framework within a company developing high-tech products using a traditional project management approach.

To build this dynamic understanding, an in-depth case study was conducted on the industrialization department of a large firm producing complex products (Yin, 2015). As this research has an exploratory nature, and the goal is to build theory inductively, applying a case study is appropriate for answering the research question (Pratt, 2009). This implies studying a real-world setting and systematically collecting and analyzing qualitatively generated research data to build a theory (Yin, 2015).

4.1 Case Setting

According to Eisenhardt and Graebner (2007), theoretical sampling of single cases is straightforward as they are chosen because of their unusual revelatory, extreme exemplars, or opportunities for unusual research access. Following this logic, the industrialization department of a company developing high-tech lithography systems was selected as a case study, further referred to as LithoCo. These high-tech lithography systems require the integration of highly complex hardware and software and consist of thousands of hardware parts. While LithoCo is developing its high-tech products using a traditional V-model approach, the industrialization department has implemented a large-scale agile framework to manage its tasks. More specifically, the industrialization department has implemented the scaled agile framework (SAFe). The suitability of this case for answering the research question is twofold. First, the implementation of the large-scale agile framework is in a setting that develops high-tech products. Second, large-scale agile is being applied in combination with a traditional project management approach.

The function of the industrialization department is to industrialize the manufacturing of newly introduced lithography systems developed by the so-called Design and Engineering (D&E) department. This means that the industrialization department receives a proven prototype and will guide the process of building this product again from scratch. As the work preparations for these new systems are not always ready and potentially contain technical disturbances before the system build starts, they will have to be redesigned or updated. To industrialize and transfer the manufacturing

of a new product introduction to the factory for volume production, the industrialization department has to validate and update work preparations and solve potential technical disturbances within five pilots. Each pilot produces one product from scratch and delivers it to the customer. The five pilots for each newly introduced product are done sequentially. The goal is to identify and solve manufacturing disturbances within each pilot, moving towards a smooth manufacturing process of the new lithography system within these five pilots. Besides the new system introductions defined as value stream 1, the industrialization department also manages two additional value streams that provide projects. Value stream 2 is concerned with process improvements and value stream 3 is concerned with refurbishing older systems.

Since the industrialization department receives a working product from the D&E department and has to transfer the production of the system towards volume production after five pilots, it is operating in the design-manufacturing interface. As the D&E department transfers the responsibility to the industrialization department after the first proven prototype, the industrialization department is responsible for the industrialization of the product and aims to reduce the cycle time of the product development within the next five product iterations. After these five newly manufactured products, the responsibility is transferred from the industrialization department to the volume production department if the product is ready enough. At this point, the cycle time has to be further reduced until the desired level.

The project portfolio of LithoCo is managed by an overarching portfolio layer. For each project, a dedicated project team is responsible to guide a particular project through the D&E department, the industrialization department, and finally the volume production department. Both the business line and the product team are further categorized in this study as the portfolio layer. Within the industrialization department, there is split between the program layer and the team layer. The program layer consists of the project leaders and project managers that are responsible for guiding a project through the department. The team layer consists of engineering teams responsible for the engineering tasks related to industrialization. A graphical representation of the lithoCo's organization can be seen in figure 12 in Appendice A.

4.2 Data Collection

Qualitative data was gathered between September 2022 and April 2023. To ensure triangulation and strengthen the validation of this study (Yin, 2015), three different data collection methods were applied: (1) Semi-structured interviews, (2) reviewing company documentation, and (3) observing company processes. Interviewing employees with different roles within the department would enable qualitative data generation from all organizational points of view. To ensure data saturation, multiple individuals per role were interviewed when possible. Table 2 summarizes the data collection methods and corresponding details.

Table 2: Used data sources, their data characteristics, and added value.

Data source	**Data characteristics**	**Data added value**
Semi-structured interview	22 interviews with 22 informants: Line manager (3) Project manager (3) Project leader (3) Release train engineer Production engineer/ Scrum master (5) Production engineer (2) Technical support engineer (2) Product owner (2) Project leader/Product owner	Collecting detailed information on how the actors were interacting with the information flows, department processes, organizational structure, roles, task executions to understand the causal and endogenous dynamics underlying the implementation of large-scale agile in the department.
Participatory observation	Formal meetings: daily stand-up meetings, scrum of scrums meeting, PO sync meeting, sprint review meeting, sprint planning meeting, ticket refinement meeting, retrospective meeting, PI planning event, PO-SM study session, splicing ticket session,validation meeting, team meeting, alignment meeting Informal talks taking place during and after meetings and/or interviews	Observe the transfer of information, interaction of actors with processes, experience challenges regarding the large-scale agile implementation Confirm information from semi-structured interviews
Company documentation	Meeting minutes, department handbook, presentation slides	Expand contextual knowledge to understand actors experiences. Confirm information from semi-structured interviews.

To identify the underlying mechanisms of the dynamics of implementing large-scale agile in such an environment, interviewees were asked about their experiences with the large-scale agile processes, encountered information flows, planning, function roles and the structure of the department and company (Appendice C). Since documentation about processes, projects, meetings, and planning details were also documented in company documentation, this allowed for verification. Interviewees were asked about particular projects or situations from which they gained their experiences. This was done to minimize the retrospective bias since particular events are easier to recall more accurately (Chell, 2004).

As an observer-participant within the company, observing the field setting allowed for gathering data from the process. These observations were done during daily meetings of the scrum teams, bi-weekly meetings of the product owners, weekly meetings of the team, sprint meetings every two weeks, and the quarterly meetings of the project interval meetings. Additionally, alignment, validation and study session meetings were attended. Being present during these parts of the process would allow for identifying organizational dynamics and verifying the data gathered from company documentation and interviews.

4.3 Data analysis

To stimulate the scholarly rigor of qualitative research, this research adopted the qualitative analysis approach of Gioia et al. (2013). Collected qualitative data from the interviews, company documentation, and field notes made during the observation of company processes were analyzed by applying open coding. Firstly, the goal was to generate first-order concepts to make sense of the data. During this first-order analysis, the focus was on identifying the key topics from the perspective of the interviewees that related to the project management approach. To reduce the first order concepts to a manageable number, similarities and differences among the categories were sought (Gioia et al., 2013). Secondly, as more similarities and difference among the qualitative data were found throughout the research, more theory-driven second-order themes could be established that categorized multiple first-order concepts. After the second-order themes were established, they would undergo further theoretical saturation to develop aggregate dimensions (Gioia et al., 2013). These aggregate dimensions represent a combination of multiple second-order themes. This process of data analysis and coding resulted in a so-called data structure depicted in Figure 1

The first-order concepts, second-order themes, and aggregate dimensions form the basic building blocks for the data structure. This data structure allows for transparency since it clearly illustrated the transition from raw qualitative data to concepts and themes. The derived data structure is illustrated in Figure 1. As can be seen in this data structure, multiple second-order themes are

connected to multiple aggregate dimensions. This illustrates that the second-order themes do not necessarily characterize one specific aggregate dimension but are also interrelated. This is because the aggregate dimensions describe dynamic behaviors that share certain second-order themes. Note that constructing the data structure entails continuous iteration between the raw data, concepts, themes, and dimensions. Furthermore, to further elaborate on transparency, Table 6 in Appendice D shows how exemplary quotes led to the construction of the first-order concepts and second-order themes.

Finally, to transform the static data structure into a dynamical representation that illustrate the mechanisms underlying the implementation of an agile project management approach within an organization developing high-tech products using a traditional approach, this study developed a causal loop diagram (CLD) originating from Sterman (2001). By illustrating the causal and endogenous relationships between the constructed themes, the CLD can generate a feedback-driven system that models complex and dynamic behavior. This CLD would then, in a graphical way, illustrate potential bottlenecks in the causal and endogenous relationships between the constructs, which is beneficial for investigating particular mechanisms within an organization. The generation of the CLD is done by representing key variables and indicating their causal relationships using arrows. These arrows can indicate either a positive or negative relationship.

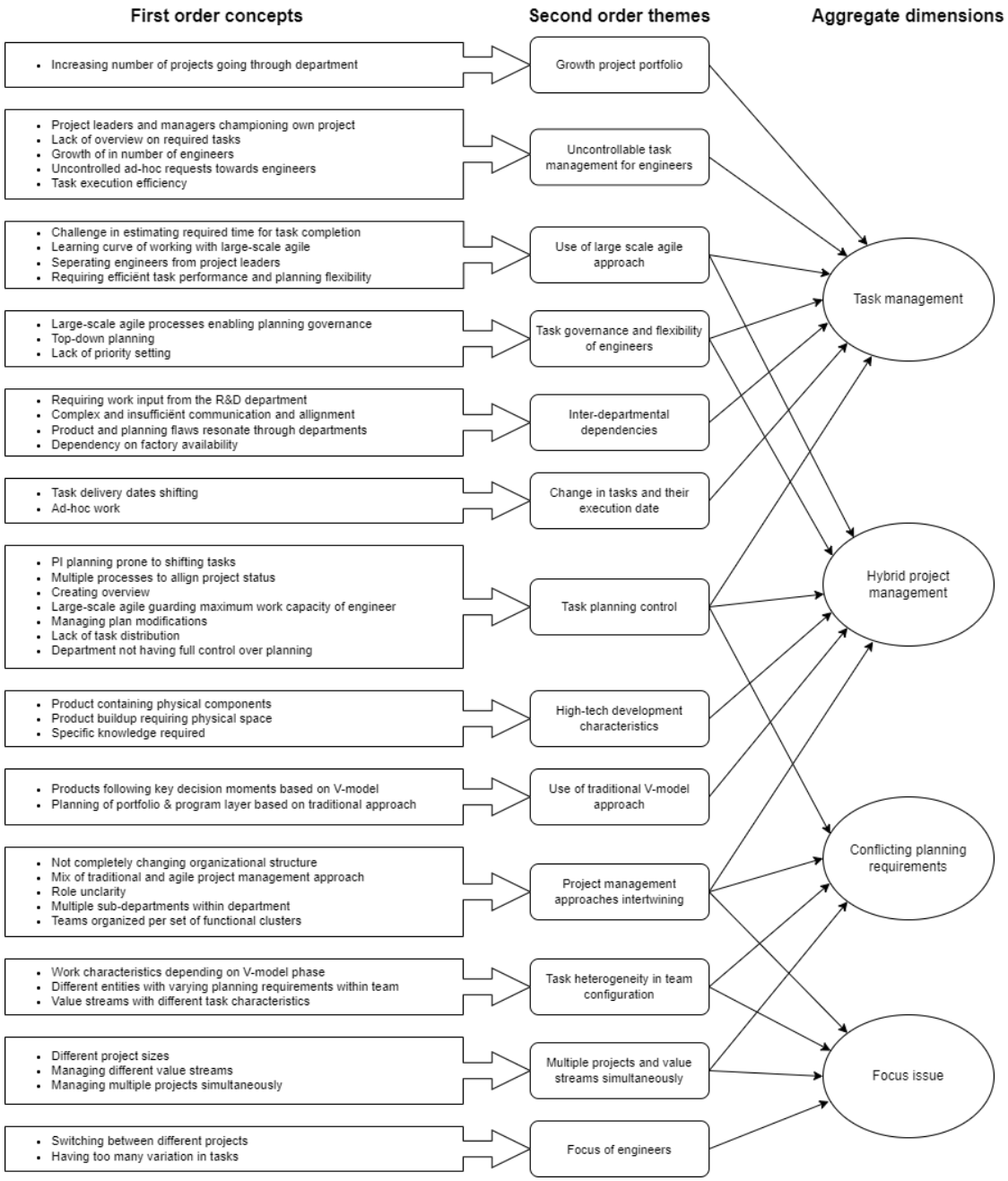

Figure 1: Data structure

5 Findings

Figure 2 represents the CLD that illustrates the causal and endogenous relationships between the derived variables. The dynamic behavior of this feedback-driven system and how it explains the mechanisms underlying the implementation of a large-scale agile framework within a company that is developing its high-tech products using a traditional V-model approach is thoroughly explained by means of a detailed case narrative.

5.1 Developing a dynamic perspective on large-scale agile implementation in a high-tech environment

In the following sub-sections, the key variables and dynamics that characterize the large-scale agile implementation are explained through four feedback loops. First, the task management loop explains the intended action to fix the problem symptom of an uncontrollable task management environment for engineers, consequently balancing this negative effect by implementing a new project management approach. Second, the hybrid project management loop explains the unintended reinforcing dynamics caused by the implementation of the agile project management approach within a hardware industrialization case setting that simultaneously executes a traditional project management approach. Third, the conflicting planning requirements loop explains the unintended dynamics of dealing with contrasting task characteristics within a project team, resulting in a reinforcing effect on the uncontrollable task management environment. Fourth, the focus issue loop captures the consequences of having a high variety and quantity of tasks, causing a reinforcing effect on the uncontrollable work environment.

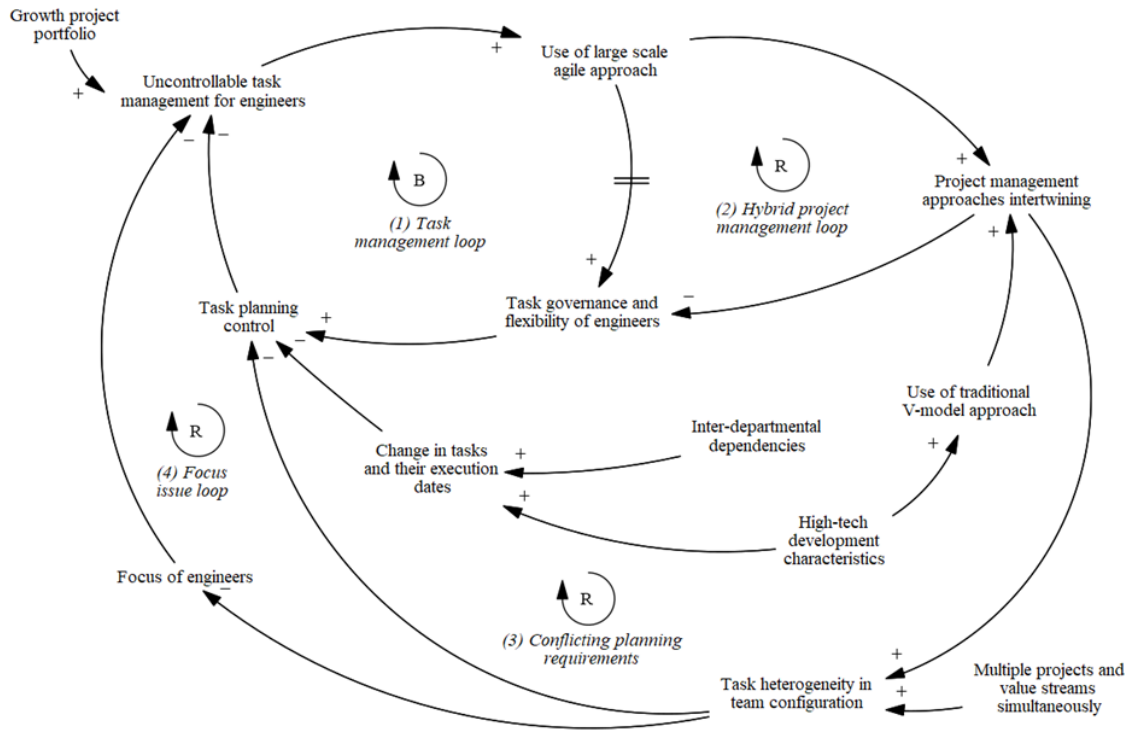

Figure 2: Causal loop diagram of the large-scale agile implementation within the case setting. The 'B' and 'R' loops illustrated in the Figure define the loops nature: either being balancing (B) or reinforcing (R). The '+' and '-' indicate the polarity of the causal link. The double line intersecting the arrow denotes a delay.

5.1.1 Balancing loop 1: The task management loop

Initially, the project teams within the industrialization department were organized per project. The amount of work could be managed using a matrix organizational setup since the number of projects being processed by the department was limited. In this setting, a project leader or manager could form a team of production engineers and technical support engineers to execute the concerning project. When the number of projects going through the department increased, the work efficiency of the engineers decreased. This is indicated in the CLD with the exogenous variable **growth project portfolio** (Figure 3). As one of the line managers mentioned: *"The portfolio of projects grew enormously. As a result, the engineers started participating in all those projects for a particular amount of their time. Plus, there was a lot of inefficiency in it, those engineers were more in meetings than they were busy with engineering work"*.

This growth of the project portfolio triggered multiple dynamics. First, the department experienced substantial growth in the number of engineers. Second, a lack of overview of the required project

task deliveries caused scenarios where engineers missed important task delivery dates. Third, uncontrolled ad-hoc task requests towards the engineers decreased their work efficiency. Fourth, project leaders and managers championing their own projects decreased the project priority overview for the engineers. These four dynamics created **uncontrollable task management for engineers** which is indicated as a variable in the causal loop diagram (Figure 3).

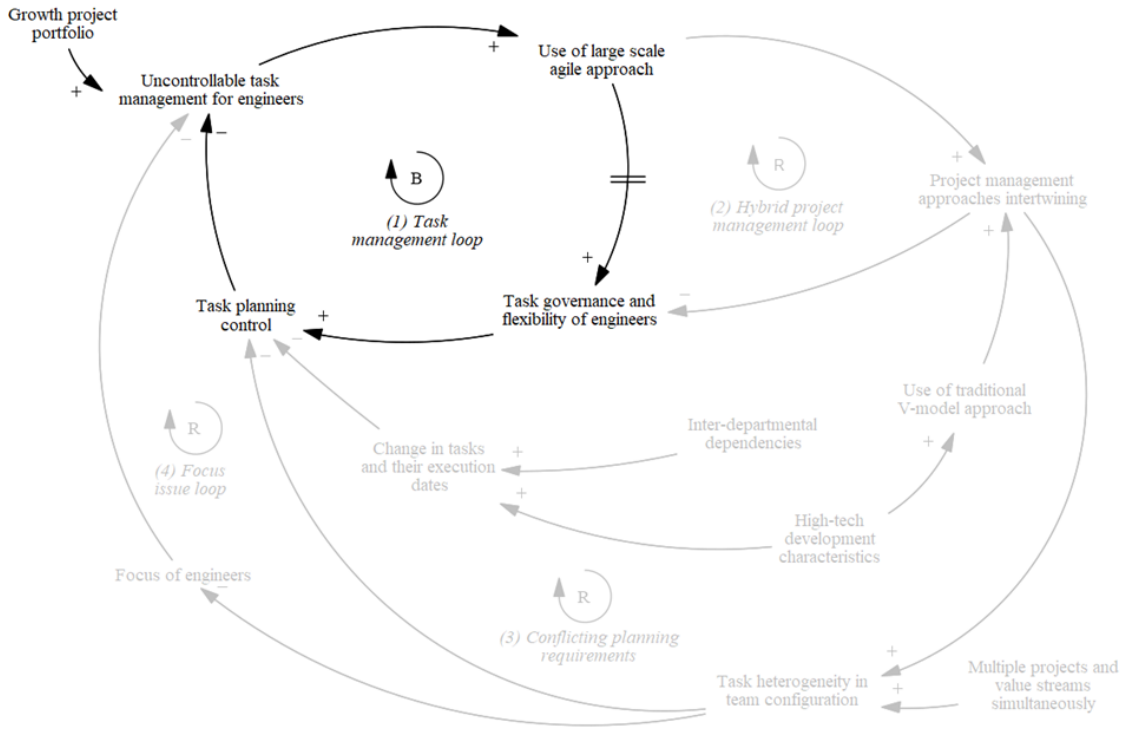

Figure 3: Causal loop diagram of the large-scale agile implementation with emphasis on the task management loop. The highlighted variables and arrows describe the task management loop. The 'B' and 'R' loops illustrated in the Figure define the loop's nature: either balancing (B) or reinforcing (R). The '+' and '-' indicate the polarity of the causal link. The double line intersecting the arrow denotes a delay.

This uncontrollable task management for the engineers was the incentive for management to implement a large-scale agile approach, indicated in the CLD as the variable **use of large scale agile approach** (Figure 2). As one of the line managers stated: *"That was the moment we said, how can we organize this more efficiently? What methods can we use in the dynamic world of this company where plans change often? We considered different project management structures and concluded that we had to centralize the engineering community and bring the work prioritized to them".*

As the work-demanding project leaders and work-delivering engineers would be decoupled, this would promote the **task governance and flexibility of engineers**, which is indicated as a variable in Figure 2. One of the line managers explained: *"Looking per product, what work do we have in the coming quarter? Visualizing the work in blocks of work, called features, and bringing these features to the engineering teams who will detail out what is necessary to complete the feature successfully"*.

The use of the large-scale agile approach would enable the engineers to plan their own work, providing planning governance. Having a clear overview of the amount of requested work upfront would allow for a clear planning overview, consequently enabling the engineers to be flexible in constructing their planning. In addition, the large-scale agile processes would provide guidance to the plan execution of the teams. As one of the line managers explained: *"The power of this process is that once a quarter, we gain an overview on what work is coming in the next quarter, breaking this work into working blocks of 2 weeks. The teams have their daily stand-up meetings to guide and discuss the progress of the work they planned. That you have the PO sync meeting that, when priorities change, management gets the chance to guide this"*.

As the incentive of using a large-scale agile approach was to provide more task planning governance and flexibility to the engineers, implementing such practices introduced organizational adjustments. First, large-scale agile implementation recommends the construction of scrum teams that are dedicated full-time to their team and organized around value to enable the continuous delivery of work. Second, the implementation of multiple processes is recommended to guide the teams on intra-team communication, inter-team communication, and interdepartmental progress alignment.

The processes of large-scale agile are designed to enable planning governance for the engineers by providing structural alignment processes, giving the engineers the autonomy to estimate how many tasks they can deal with, creating a planning overview by translating the tasks into features and tickets, visualizing team inter-dependencies on the program board and being able to react to changes. These factors were experienced by the engineers, as one of the production engineers mentioned the increased planning overview: *"It provides me with clarity in such a big organization. [...] Now I have a nice program board to plan together, it is nice to be able to fall back on that "*. Additionally, another production engineer mentioned the ability to react to changes: *"We have the daily stand-up. So if something new comes up during a sprint then you can ask people during this daily stand-up like: I don't have time for this, can someone take this up for me?"*. These factors would positively affect the engineer's **task planning control**, indicated as a variable in

Figure 2. As of the product owners explained: *"I like how we use the SAFe rituals, so the meeting structure, to have good discussions. [...] You are in a certain cadence talking to each other and also helping each other with the actions that need to be done. Whereas in a traditional way of project management, you are not all sitting together to discuss this, then it's often 1 on 1, and this happens or not"*. The ability of the engineers to have control over the planning would mitigate the uncontrollable task management environment for engineers.

The goal of the balancing task management loop was to improve the uncontrollable task management of the engineers by adopting an agile project management approach to provide planning governance and flexibility which would enable task planning control to consequently mitigate the uncontrollable task management environment of the engineers. Unfortunately, despite the expected and perceived benefits, the effect of the balancing task management loop was prone to a delay. This delay is illustrated in Figure 2 as a double line intersecting the arrow between the use of a large-scale agile approach and the task governance and flexibility of engineers. Since the use of a large-scale agile approach introduced the implementation of new processes and roles, the full benefit of the implementation was subject to a learning curve. Multiple factors initiated this learning curve.

First, requesting the work in work packages was found to be challenging due to the lack of experience. The split between the program layer, consisting of project leaders and project managers, and the team layer, consisting of the engineering scrum teams (Figure 12), required the program layer to request their work using features. The program layer perceived challenges in translating their project plan into sufficiently defined features. This was confirmed by one of the support engineers who mentioned that the so-called definition of done of these features was often missing: *"When project leaders and project managers come to us, these are all high-level requests: we want the procedures to be validated. But what do you mean by that, what do you want us to do? [...] The definition of done is missing there"*.

Second, a lack of guidance on large-scale agile implementation caused unclarity in the role fulfilment of the newly introduced scrum master and product owner role. As one of the scrum masters mentioned: *"I do not know if I really have been a scrum master. I did the exam, I have the certificate, and everybody calls me the scrum master but I don't know if I necessarily have been a scrum master"*. Next to the scrum master role, the product owner role implementation was subject to a lack of skill and knowledge on how to practice the role. While one of the tasks of the product owner's role was to manage and prioritize the backlog of the team, fulfilling this task sufficiently was found to be challenging. This lack of skills in practicing the product owner role was also experienced by one of the production engineers/ scrum masters: *"I felt that as a PO*

you need to understand the work of your team. Because if you go to the PO sync and they are requesting something from your team, you need to know how much that is. [...] we are missing somebody who can be like: wait it's not that easy, maybe this could be a risk." This indicated that there is a lack of skills and role clarity in practicing the scrum master and product owner roles.

Third, the lack of estimating how much time is needed for a particular task limits the engineer's ability to do priority setting and thus mitigates planning governance and flexibility. As one of the project managers explained: *"There is no transparency in how much effort it actually takes to do a task and how full the team actually is. [...] And if a PO receives an ad-hoc request then you should also say, or raise the question, what is the priority of this item versus the list of my backlog, and what will be dropped. I haven't heard these kind of questions in the last 6 months. So that is why it has a lot of impact, nobody really has a complete view of what fits, what doesn't, and what the priorities are"*. These factors mitigated the balancing effect of the task management loop (Figure 3).

Besides the delay in implementing the large-scale project management, the task management loop is prone to LithoCo's hardware case setting. As the company is developing high-tech products, it is subject to multiple factors. First, the product requires physical components that have to be built up. Second, in order to build up the large product, physical space, and assembly technicians are required. Third, as the product consists of highly complex technology, engineering on the product requires specific knowledge. These characteristics are constructed in the causal loop diagram as the variable: **high-tech development characteristics** (Figure 4).

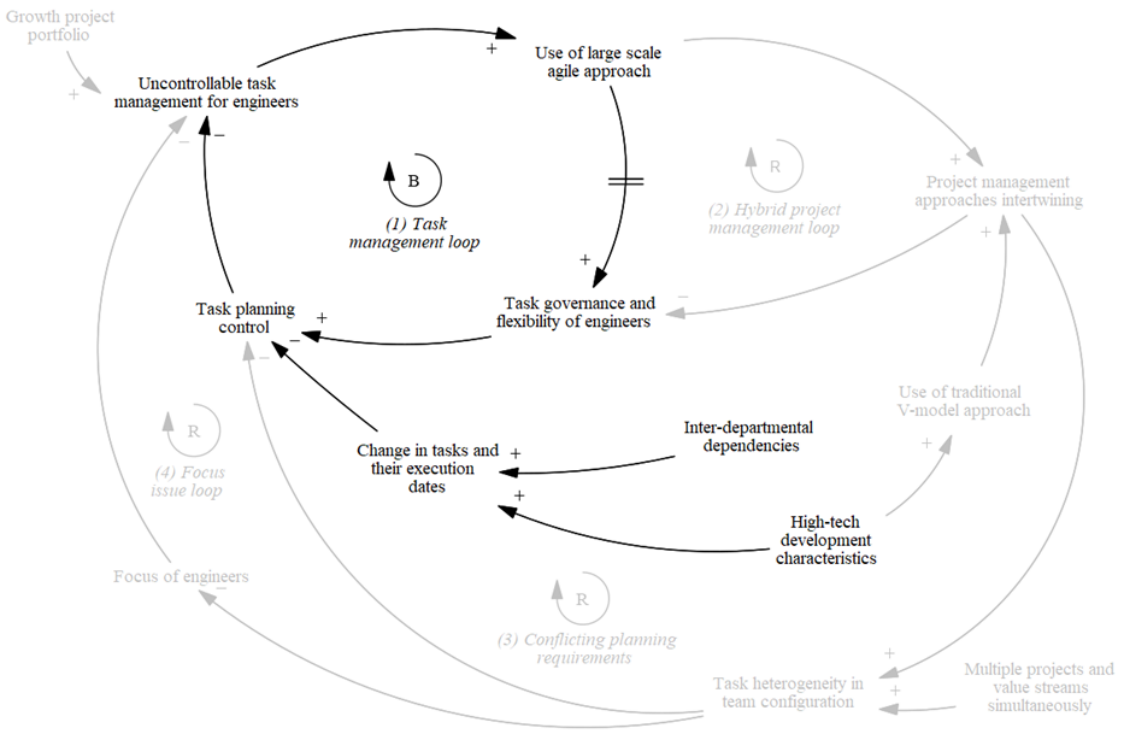

Figure 4: Causal loop diagram of the large-scale agile implementation with emphasis on the task management loop and the case setting. The highlighted variables and arrows describe the task management loop and the effect of the case setting on this loop. The 'B' and 'R' loops illustrated in the Figure define the loop's nature: either balancing (B) or reinforcing (R). The '+' and '-' indicate the polarity of the causal link. The double line intersecting the arrow denotes a delay.

As a consequence of the development of high-tech products, the industrialization department of LithoCo is operating in a multi-departmental setting. The product development process starts within the design department, after which the industrialization department will take over the process after the first prototype is developed (Figure 12). This means that the industrialization department is depending on the work input from the D&E department. The handover of product responsibility means that communication and alignment processes between these departments at multiple stages of the product development stage are important as product and planning flaws might resonate through the departments.

As each department has its own way of working, sufficient alignment of task transfer moments between the departments is complicated as one of the program managers explained: *"We have our delivering parties, which are many different functional clusters from the design department that all have their own PI events. They all have their own planning processes, most via SAFe, some not even that. The link between those functional clusters that have their own SAFe events, that*

alignment between them and us, there is such a mismatch, you are so dependent on it".

This mismatch in planning was also experienced by the scrum teams, as one of the production engineers explained: *"You always have a time frame that you are given a sprint during your PI planning that: oh I am gonna do it in sprint 2. You will never get this document in sprint 2. You will never get this document in sprint 1. And we made an agreement: we need these documents 2 weeks in advance for us to review them and come up with good feedback. You don't get them, and because you don't get them in time and there is so much more important stuff to do, this just gets pushed out. And you either just don't end up doing it and saying: I don't have feedback. And then you look at it much later when there is an actual problem on the floor".* This shows that the ability to work on tasks according to the derived planning is depending on input from other departments.

These **inter-departmental dependencies**, indicated as a variable in the CLD (Figure 4), and the high-tech development characteristics variables have an effect on task planning control. The delayed task deliveries of inter-departmental counterparts, delay in supplying materials, and lack of availability of space or operators in the factory can cause the task delivery dates to change. In addition, product flaws might initiate escalations, causing unplanned ad-hoc work that has a higher priority than the planned work since these escalations will hamper the development time of the product directly. One of the production engineers explained: *"There was a system in which part 1 crashed into part 2. Multiple parts of the system were broken, so that has to be repaired in the working center. At that point, materials need to be arranged: a repair order needs to be made so that the broken part is built out, work center capacity has to be there, the right tooling is there, an action plan is present, all those things have to be arranged. [...] So when it happens, it has to be managed immediately because the system is on hard down and already too late. So you need to drop everything to take that up".*

The occurrence of such escalations will result in the planning deviating from the initial planning, which is indicated as the variable **change in tasks and their execution dates** in the CLD (Figure 4). This change in tasks and their execution dates negatively impacted the task planning control, which could best be explained using the PI planning. The generated PI planning, which is constructed for a time period of three months, will be subject to shifting tasks. As a consequence, planned tasks will move out of sprints and even whole PI's. As explained by one of the production engineers: *"It is a little bit of a waste of time, in the sense that you can not get attached to your PI planning. I think some of the teams in the department make the mistake of saying: "this is my holy grail for the next 3 months." That is never true, it is not possible because you work in a factory environment, with output, and with breakdowns".*

As a result of the constant change of planned tasks during the 3-month schedule, the task planning control decreased (Figure 4). In response to these task-planning changes, engineers introduce a certain amount of time percentage in their PI planning that was reserved for unplanned ad-hoc work. As one of the product owners explained: *"if we plan 56% of our time with planned work and the remaining for support then we are usually fine"*. While the PI planning creates an overview for the engineers to stimulate planning control, the change in tasks and their execution dates mitigate this effect. As a result, the balancing effect of the task management loop is hampered.

5.1.2 Reinforcing loop 2: The hybrid project management loop

The industrialization department is applying a traditional project management approach to manage the projects. To develop its high-tech products and produce them at high volume, LithoCo is organized into multiple departments: the D&E department, the industrialization department, and the volume production department (Figure 12 in Appendice A). To guide the development of the large portfolio of high-tech products going through each department, an overarching portfolio layer monitors the progress of these projects (Section 4.1 & Appendice A). The nature of the high-tech products, containing complex hardware development characteristics, requires the portfolio layer to plan the development of these products linearly, predictably, and far upfront. As one of the project managers explained: *"We make a system for company X and they can build something with that. [...] our system engineers then make a roadmap like: if company X wants to be there in 2030, then we need to have our systems over there to meet their customer's wishes. Knowing that, system 1 has to be on that company's floor by 2024, there the product will ship. [...] Then we will plan back. If we ship there, we have to test it here, then we have to build there, then the design must be known at that time, then the first prototype must be ordered"*.

This traditional project management approach is executed using the product generation process, consisting of multiple key decision (KD) moments that help to determine the required content of the product, enabling alignment of stakeholders, and retrieving the right materials at the right time. These KD moments are milestones to guide the project through multiple departments. Within this product generation process, the V-model is integrated since the product contains a developing phase, consisting of the system design, design realization, and volume production preparation. The guidance of the V-model on the project management of the industrialization department is defined in the CLD as the following variable: **steering on a traditional V-model approach** (Figure 5).

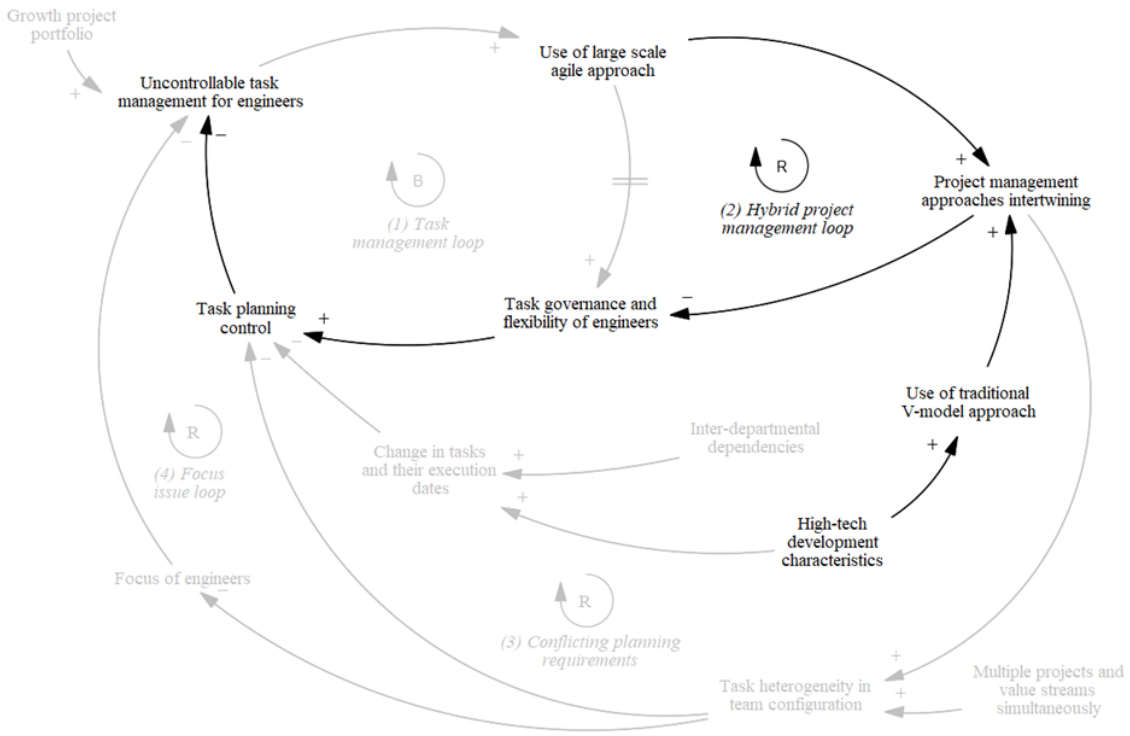

Figure 5: Causal loop diagram of the large-scale agile implementation with emphasis on the hybrid project management loop. The highlighted variables and arrows describe the hybrid project management loop and the effect in the task management loop. The 'B' and 'R' loops illustrated in the Figure define the loop's nature: either balancing (B) or reinforcing (R). The '+' and '-' indicate the polarity of the causal link. The double line intersecting the arrow denotes a delay

The intertwining of the traditional and large-scale agile project management approach gave rise to multiple challenges in smoothly integrating the large-scale agile approach. As the industrialization department was steering on a large-scale agile approach, consequently splitting its engineers from the project leaders and managers, a mix of traditional and agile project management approaches arose. While the team layer of the department started to manage its projects using large-scale agile, the program layer and the portfolio layer were using a traditional project management approach. As a result, these two contrasting approaches started to intertwine within the industrialization department. This intertwinement is indicated in the CLD as **project management approaches intertwining** (Figure 5). As one of the project managers explained: *"The teams here are trying to work on a scaled agile way of working. The project leaders and managers are only looking towards KDs saying: you need to do this otherwise you don't meet my timeline. But that is also because above them, the portfolio layer is also working like that. Even though they have all their sectors and the D&E department working with SAFe, they don't want to do it"*. Integrating the large-scale agile project management approach in an industrialization environment that applies

traditional project management approach gave rise to multiple challenges.

As the program layer and the team layer were now separated and both layers were applying a different project management approach, the communication between these layers perceived challenges. The portfolio layer, overarching the departments of the company (Figure 12 in Appendice A), finances and guides the projects high level using a traditional project management approach and determines what systems are developed and when. This planning is communicated from the portfolio layer to the program layer within the industrialization department. The program layer then defines its own detailed traditional planning based on this input to guide and guard the high-level planning of the project within the industrialization department. The program layer is then communicating its traditional planning to the agile planning of the team layer, which requires translation of the planning. As one of the project leaders explained: *"We can not plan in [digital scrum board], for us, it's a tool to bring work in a structured manner towards the teams. That is high level, we have to do it otherwise nothing will be planned. [...] So we have different master plans and different PGP processes. We have to bundle all that information and make features.[...] It works fine for the engineers as they can work structured, but for the project leaders, the added value is limited"*. As a result, the contrasting project management approaches complicated communication.

The dynamic industrialization environment of the department further complicated the communication between the program and team layer. While the program layer has to translate its traditional planning to the agile planning of the engineering teams, the traditional planning of the program layer is subject to changes. As one of the project leaders explained: *"Every two hours I could open my planning and see that the next start was delayed by a day again because of material issues or other things"*. The challenge with the separation of the program and team layer is that when changes occur at the program layer, they have to be communicated instantly to the teams so that they can act on these changes. This implies that the translation between traditional planning and agile planning has to be revised, as one of the project managers explained: *"What happens the most is that tasks shift. [...] And also often that it happens at another PI, so a quarter shift. It also happens that you have to ask things from the team layer that you did not foresee and all these matters must be discussed at the PO sync, where you discuss new plan changes and adjustments. That takes a lot of effort. As a result, you have to adjust all those features afterward and there is quite a bit of administration that is secretly attached to the whole SAFe principle. So you have to align and shift everything with a lot of people and you get a lot of questions from people like: how is that possible? It is also blamed on you sometimes while it actually happens to you. Because you have to plan based on what the portfolio layer assumes and that is the best-case scenario"*. As

a result, communication of plan changes and adjustments through the large-scale agile framework processes is complicated by the dynamic industrialization department.

Besides the challenge in communication between the program and team layer, not completely changing the organizational structure of the department according to the large-scale agile framework complicated smooth implementation of the framework. As the department was implementing the large-scale agile framework for its engineers, the project leaders and managers did not undergo this transition. The new practices and organizational structures related to the large-scale agile framework were implemented while old practices and organizational structures related to the initial way of working remained in place, resulting in a mix of processes and structures. As one of the product owners explained: *"If you look system specific, it just passes through the V-model. [...] The scaled agile framework has just been thrown over it "*. Not completely changing the organizational structure of the department complicated the smooth implementation of the framework in at least three ways.

Firstly, having a mix of initial roles and new roles implemented by the large-scale agile framework would cause role unclarity within the department. As the team layer was now implementing the roles and processes of the large-scale agile framework, the program layer was not applying large-scale agile at all. As a result, the project leaders of the program layer did not know how they would fit within the large-scale agile approach. As one of the project managers explained: *"In this department, PLs or epic owners/business owners, do not know what their role should be exactly. Definitely, they do not know what to do during the PI events. Why are we here? Why are we wasting our time? You can hear that back a lot. Why do we need sprint reviews? What is our role in the sprint reviews? So they don't understand the framework, they don't understand the way of working. That's from the PL's side"*.

Secondly, the number of meetings in the department increased. While the large-scale agile framework prescribes multiple processes for project management, these processes did not completely replace the old processes. As a result, the processes implemented by the large-scale agile framework are on top of the processes that were already in place. As one of the production engineers explained: *"The problem with scrum SAFe is, we introduce more meetings on top of the work we already have. So I gave all of my information about a product to that respective PL in my PL validation meeting. [...] Then my PO also has to summarize this information and keep it back in the PO sync. [...] Now I am giving the same information twice. [...] Same for the PL, who is hearing it twice. [...] What I find with the way of working now is, we keep saying the same thing over and over again with different audiences. And most of that audience overlaps"*. This increase

in meetings would decrease the effective time of engineers to work on their tasks.

Thirdly, implementing the product owner role in the department was perceived as a challenge that was caused by the inability of the product owner to successfully fulfil its set of tasks, as described by the large-scale agile framework. This inability was caused by the required adaption of the product owner's role to the industrialization department setting. The responsibilities of the product owner entail managing customer collaboration, contributing to the vision and roadmap, supporting the team in delivering value, and managing and prioritizing the team backlog. The necessity to adapt these product owner responsibilities had multiple causes. First, customer collaboration by the product owner is not possible due to the current organizational structure of the company. As there are multiple organizational layers between the engineering teams and the customer, customer collaboration will not be with the end user. As the release train engineer explained: *"The challenge that I see within LithoCo is, the customer is not near us and the customer is also not accessible. [...] There are multiple layers between me and my customer. And you can call those layers, PM's and PL's, and then PO's"*. As a result, customer collaboration will be with internal stakeholders. These internal stakeholders are the product leaders and product managers from the program layer and the volume production department.

Since multiple projects have to be managed by the team simultaneously, the product owner role should be able to align with all stakeholders of each project to contribute to the vision and roadmap, support the team in delivering value by solving impediments, and consequently manage and prioritize the team backlog. As the release train engineer explained: *"If there are no PM's and PL's , then the PO has to do what the PO does for a team, for the train, in the sprint review, PO sync, PI planning he does all of that. And on top of it, he has to talk to many customers in various parts of the world timing-wise, and then the volume production department and D&E department, I mean the lists of stakeholders, business stakeholders, customers, proxy customers, is so big. One PO can not do that scope"*. As the product owner would have to deal with all these tasks for each project moving through its scrum team, this would lead to an uncontrollable situation as one of the project leaders explained: *"The product owners, their head is just not big enough. They just can't keep track of the status of all those 30 different things. They really need a PL for that communication"*. This challenge of having too many tasks to handle for a single person was broken down by separating the tasks, as explained by the release train engineer: *"So the way the company has divided this challenge is that they say: let PM and PL do external communication and let PO do internal communication. So they just broke the problem down into easier chunks that can be done by people"*.

This split in external and internal communication between the product owner of the team layer

and the project leader of the program layer has multiple consequences. Since the project leader is responsible for the delivery of certain competences of a particular project and is doing external communication, he or she also adopts the responsibility of contributing to the vision and roadmap and supporting the team in delivering value by solving impediments. As a result, these two responsibilities will have to be interpreted differently by the product owner. Furthermore, the product owner will maintain the full responsibility of managing and prioritizing the team backlog. This reduction in product owner responsibilities resulted in the product owners feeling like a planner, as one of the product owners explained: *"If I look at the PO role, I feel as a department planner. Planner for the department while that doesn't actually entail the role. The PO should have a business vision and carry the product"*.

While the product owners are still having the responsibility of managing and prioritizing the team backlog, the product owner should acquire the information necessary for facilitating this. As the external communication contains team-relevant information and is managed by the program layer, the ability of the product owner to act on this information is depending on the degree of communication between the team and the program layer. As the program layer might receive beneficial information from external communication that is related to the team backlog, this information has to land at the product owner of the teams. As a result, the extent to which the product owner can manage and prioritize the team backlog is depending on the degree of communication between the program layer and the product owners of the team layer.

These perceived integration challenges, categorized under the variable project management approaches intertwining, affected the task planning governance and flexibility of the engineers negatively (Figure 2). First, the top-down planning of the program and portfolio layer limits the engineers planning governance and flexibility. As the KD moments defined in the traditional planning of the portfolio and program layer communicate milestones for the engineers to work towards, they are time-based instead of value-based. As a result, the engineers are required to fit their work according to this deadline instead of providing an estimate on when the tasks could be completed. This contrast in planning becomes challenging when engineers are dealing with multiple KD moments simultaneously, as a technical support engineer explained: *"The portfolio layer determines which products we will deliver in which time span. [...] And when that machine is there and the portfolio layer says: that has to continue, then we have to make sure that we do our work on that machine. And then you get all those weird peaks and valleys of workload. While scrum works well when you can decide for yourself when you are going to do something or not"*. As the portfolio and program layer define certain KD moments up front and thus plan top-down, the engineers have limited flexibility to construct their planning as they see fit, consequently decreasing their ability

to govern the planning.

Second, the high number of projects and thus features limit the engineer's ability to do proper priority setting. As mentioned earlier in section 5.1.1, the priority setting is limiting the planning governance and flexibility because of the lack of skills of the product owner and the ability of the team to estimate how much time is needed for a particular task. Besides these factors related to a learning curve with applying large-scale agile practices, there are other factors that limit the ability to do sufficient priority setting and consequently decrease task planning governance and flexibility. As the release train engineer explained: *"Priority setting is almost not there. [...] For us, at the industrialization department, it is just execution that has been left, everything else has been verified. We are just redoing some work and rechecking whether it really does work in a volume production environment. So we don't have 12 features then, we have those 12 features that the design department did in their quarter, multiplied by 50 maybe. Because there are that many steps, now we need to verify, so if you have 150 features and if you then ask any stakeholder please prioritize, it will be difficult. If I put myself in the place of a stakeholder, I can not prioritize 150. So I will also then make something like a burger and say: the top 20 are high and then there is medium and then there is the bottom at 50"*. This extended list of features makes it difficult to prioritize features since the top 20 features are considered equally important, consequently lacking a sufficient prioritized program backlog.

On top of this program backlog lacking sufficient prioritization, the usefulness of prioritization is diminished by the inability to reject incoming work. As one of the line managers explained: *"Eventually, we will do everything. The question is, do we have enough people? We have the program backlog but that is of no use. You can set priorities in what you find important, but in the end, we have to deliver everything to the volume production department. So I can say: I don't have the capacity so I deliver limited quality, but in the volume production department they don't have that capacity either. So then you move the problem and I don't want to do that"*.

The necessity to complete all the tasks within the given amount of time removes the incentive of prioritizing tasks. This lack of rejecting work was also mentioned by one of the program managers: *"Every Monday there is a meeting from the staff of the industrialization department. So the main group leads from the line and the product side where there is discussed what are the new products coming in. [...] So far I haven't heard: we don't do this. It is just: okay we understand the need and now we will have to see how and when can we accommodate it. [...] I had the same situation within D&E. It was really hard to say: we can't do it or we will not do it. Because not doing it doesn't mean it will stop. They will continue without you and later on they take the decision*

for you. Then you just need to deal with the consequences of it, good or bad. You need to take it from that moment in time. So I understand that they are not able to say no, and now they are really searching". This indicates that the industrialization department is unable to reject incoming projects from the portfolio layer. This inability to reject work was also observed during the PI planning event. As one of the teams communicated its overcapacity of 40 hours within the first sprint (2 weeks) of the PI, multiple project managers and project leaders communicated that their features could not be postponed and that the KD moments of the project will be taken with or without the input of the team.

Reasons for this inability to postpone tasks are top-down planning from higher management at the portfolio layer and the inter-departmental dependencies. In sum, the department is not operating in a vacuum and has to comply with other departments and higher management within the company. As all projects will have to be managed, prioritization of the program backlog is complicated. As a result, the challenge of prioritizing the program backlog decreases the task planning and flexibility of the engineers.

Third, while sophisticated communication between the program and team layer about changing tasks is required to enable task planning governance and flexibility for engineers, it seems to be lacking due to a lack of a prioritized backlog. As one of the engineers/scrum masters explained: *"So the project leaders will let you know that: hey this is shifting out. In time for you to stop the work and pick up something else. Or they will say to you in time but it will be communicated in a way that like, it just moved out and there is nothing to be said or done about it. But when it is moved in and your work is not ready, it's your fault. So the issue with that is that there is nobody who can take ownership of that problem. If parts are not in, parts are not in, it's not the PLs fault. It's just that the frustration will always be there because he wasn't tracking it for our FC only because he has 6 other FCs he has to look at"*. This illustrates that there is a lack of clear overview throughout the entire program layer that would enable the engineers to adjust their planning to the changing planning.

The lack of the program layer having a sophisticated prioritized backlog results in the task planning governance and flexibility of engineers to be mitigated. As the release train engineer explained: *"The only thing we are not doing well is that when something moves out, we do not have something to pull in, in its place. We don't have the stakeholders with the program backlog, that they have it prioritized and they say: okay because the date changed for feature number 10, we cancel it for this PI. Can you please look at feature number 11, we have all the documentation in it, can you spend some time if this can be planned instead?"*. To retain an equally divided planning, the teams

should be able to move tasks out or drag new tasks in when tasks get prioritized or postponed. In order to do this, product owners need to have a higher overview of what tasks they can drag in or postpone. Currently, as a sophisticated prioritized program backlog is lacking, the possibility of the teams to effectively prioritize tasks is mitigated. This results in the task governance and flexibility of engineers being mitigated.

The reinforcing hybrid project management loop tends to undermine the balancing effect of the task management loop. While the goal of the task management loop is to balance the uncontrollable task management of engineers by the use of a large-scale agile approach, the hybrid project management loop diminished this goal.

5.1.3 Reinforcing loop 3: The conflicting planning requirement loop

Since high-tech systems contain highly complex hardware and integrated software, being able to develop, adjust or understand the entire system requires highly specific knowledge. This is indicated as the variable **high-tech development characteristics** in the CLD (Figure 6). To deal with this highly specific knowledge, LithoCo has applied modularization to create functional clusters that divide the high-tech system into manageable sub-components of the system. To guide the development process of these high-tech systems consisting of functional clusters through the multiple departments, LithoCo is **steering on a traditional V-model approach**. The different KDs of the V-model define the work deliverables at each phase of the project and guide engineers in providing the required content at the specific project phase and trigger inter-departmental knowledge transfer and alignment.

As the department started to use a large-scale agile approach, the engineering scrum teams had to be designed. In defining the right scrum team configuration, the knowledge transfer and alignment between the departments that are based on the V-model were considered important factors. As one of the line managers explained: *"We looked at the volume production department structure and learned from the D&E department, who was doing SAFe and attempting to design it project-based, which did not succeed. [...] What you see is that the whole company is functional cluster-based, so competence-based. [...] So on a competence-base you have your alignment. You have your fixed counterparts on competence bases that you can keep stable"*. It became evident that inter-departmental alignment was an important factor in designing the scrum teams. In addition to considering inter-departmental alignment as an important factor in designing the scrum team configuration, continuous work delivery, team stability, and the ability to build collective knowledge were also considered. Subsequently, these factors led to engineering scrum teams being organized based on bundles of functional clusters of the high-tech system, further referred to as product modules. The consideration of designing the engineering scrum teams according to agile

and taking into account the initial organizational structure resulted in the **project management approaches intertwining** (Figure 6).

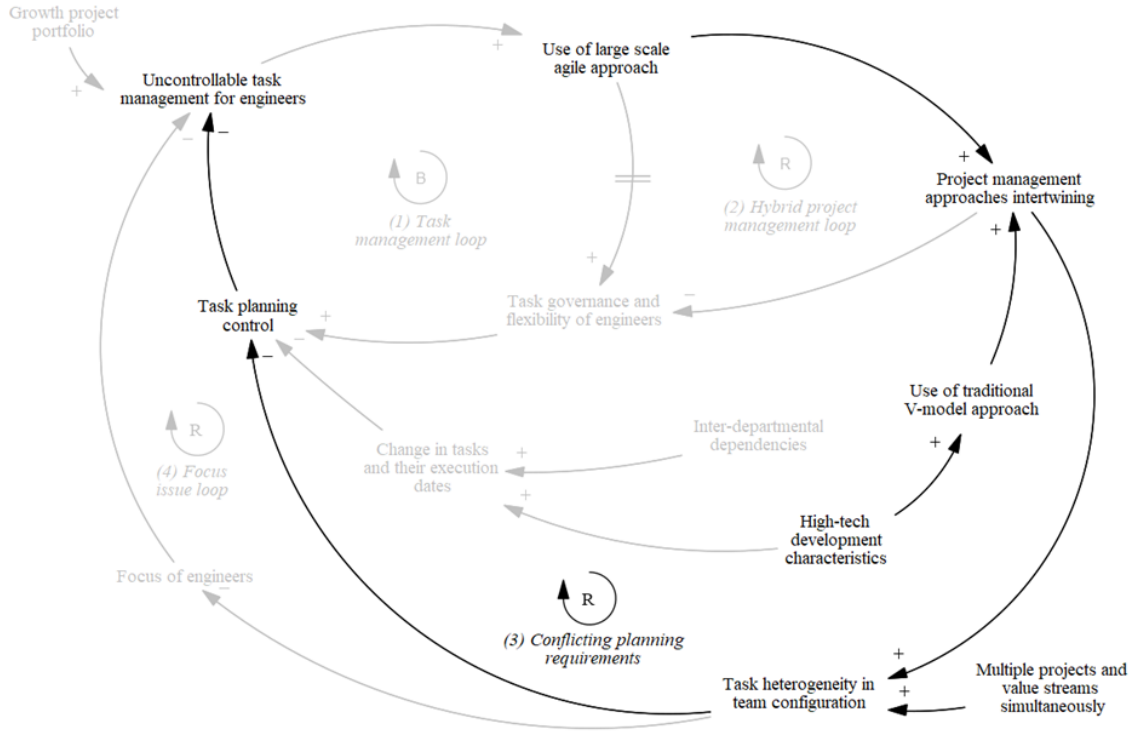

Figure 6: Causal loop diagram of the large-scale agile implementation with emphasis on the conflicting planning requirements loop. The highlighted variables and arrows describe the conflicting planning requirements loop and the effect on the task management loop. The 'B' and 'R' loops illustrated in the Figure define the loop's nature: either balancing (B) or reinforcing (R). The '+' and '-' indicate the polarity of the causal link. The double line intersecting the arrow denotes a delay

As the scrum teams were organized based on these product modules, they also included members from different sub-departments. This was caused by the organizational structure of the industrialization department not being completely restructured to the large-scale agile framework. Initially, the industrialization department consisted of two sub-departments: the production engineering (PE) department and the technical support (TS) department. When implementing the large-scale agile framework, both sub-departments, having different task characteristics, were combined within the scrum teams without merging both sub-departments. As one of the product owners explained: *"You have the scrum teams that have PE people and TS people, but they are actually two separate departments. So TS people have different managers than PE people. [...] So in that sense you have still that traditional structure of those departments"*.

The scrum team configuration is an important factor in the added value of the large-scale agile implementation as they are intended to execute the tasks going through the department. One of the factors that determine the effectiveness of the scrum teams is the variation in task characteristics that the team is dealing with, indicated as the variable **task heterogeneity in team configuration** within the CLD (Figure 6). This task heterogeneity is mainly affected by three factors.

First, as mentioned earlier, the implementation of the large-scale agile approach led to teams being structured around product modules. This implies that each project going through the department, that has tasks related to a specific product module, will have to be picked up by that specific scrum team. By doing this, the task heterogeneity within the team is constrained to particular modules of the machine. Constraining the team to the particular module enables the engineers to deal with the high level of required specific knowledge, enabling stable alignment with counterparts from other departments, keeping the team stable and consequently building collective knowledge within the team as all experiences related to the specific product module are handled by the team.

Despite the benefits of this team construction, it also entails multiple challenges. The team consisting of production engineers and technical support engineers from different sub-departments, having different task characteristics, contributes to an increase in task heterogeneity within the team. The contrasting task characteristics between these two entities have an effect on the dynamics within the team in multiple facets. One of these dynamics is the task distribution within the scrum team, as explained by one of the line managers: *"PE has other deliverables than TS. So that makes it quite complicated. The teams want to be interchangeable but that is difficult"*. Similar challenges related to task distribution were explained by one of the production engineers/scrum masters: *"...what a scrum team is supposed to be is that it is diverse, any person can pick up any tickets. But PE and TS are very split in responsibilities. And PE people want to do TS work because we want to get technical knowledge. TS people do not want to deal with any of the PE work, they don't want the administration. That is where the struggle is for us"*. Next to the task distribution, the different task characteristics of the two entities result in them requiring contrasting planning characteristics, as one of the scrum masters explained: *" On scrum, it is difficult and can be frustrating because TS has only ad-hoc work, primarily. 90% of their job is dependent on whether they have to solve technical disturbances. The other 10% comes from the information of the product and saying: I want to review this document, I have to create this presentation, or whatever"*. This was also mentioned by one of the technical support engineers: *"You do not know when a machine has a big problem and you are stuck for three weeks. You just don't know that in advance. So we are going to support*

the system, reserve time for it, and when something happens the technical support engineer drops everything and will support".

Second, as the high-tech products are being developed according to the V-model, the task characteristics vary based on the project phase and consequently require different planning characteristics. As one of the line managers explained: *"What you notice is that the left side of the V, the designing phase, is reasonably plannable. [...] It can shift a bit now and then but it's reasonably planned. [...] The moment that you start to build the systems, that is also reasonably planned but you can not predict exactly what will go wrong, so you have to stand by. You can compare it with a fire brigade, they stand by and the moment someone presses the button, they have to extinguish it".* The effect of these different task characteristics, characterized by the project phase in the V-model, on task heterogeneity is reinforced by the number of projects being dealt with simultaneously.

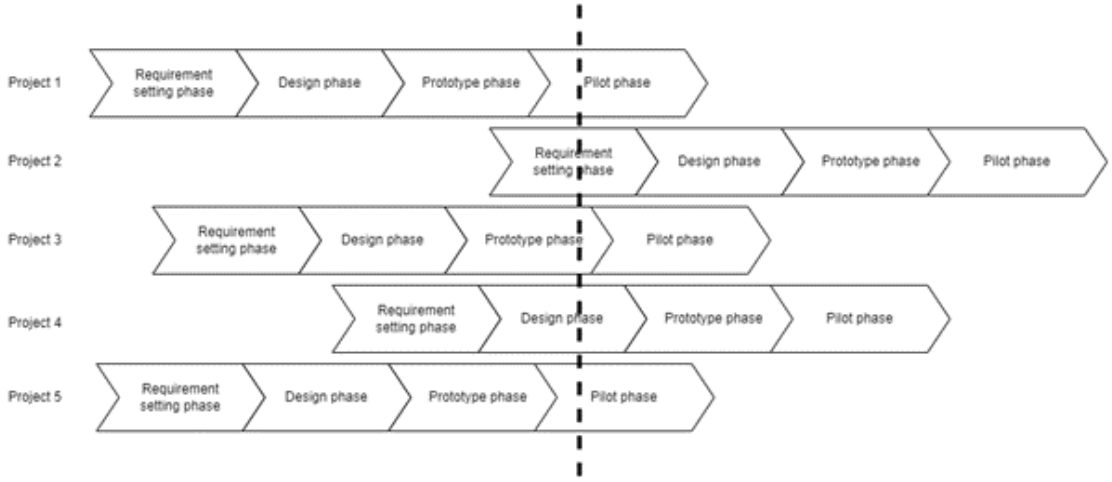

Figure 7: Graphical illustration of multiple projects and their project phase. The dotted line indicates a single moment in time.

As mentioned earlier, the organization of scrum teams based on functional clusters implies that all projects having an impact on that particular functional cluster will have to be handled by the specific scrum team. As each project can be in a different phase of the V-model, having multiple projects in different project phases will increase task heterogeneity and further contribute to the degree of conflicting planning requirements within a team. The overlap of different project phases with task characteristics is illustrated in Figure 7. As project two is in the requirement setting phase, it provides relatively up-front plannable work. On the contrary, projects one and five are in the pilot phase and therefore require the engineers to support the systems whenever disturbances occur. This occurrence of projects in different project phases was also explained by one of the engineers: *"We are still in the development stage and we have always machines in different stages*

of development and there will also be upturns and downturns. So if you have your people planned well, then the downturn will be the start of the upturn of another. So that you have that balance. Now we are in the tricky situation where there are so many parallel projects that yes, those up and down turns rarely match anymore just because the chance of you being in a complementary project like that is just a lot slimmer".

Third, the industrialization department is dealing with three different value streams. These three value streams also vary in their task characteristics, consequently increasing the task heterogeneity within the team. As one of the line managers explained: *"So for value stream 1 we have the PGP process in which you have KD moments. For these KD moments you have deliverables, so work packages in time. [...] There is a V-model in that, but in fact, you plan the work packages with key decisions over time. [...] For value stream 2 we have Hoshin Kanri. [...] So that is not related to the PGP process and the KD calendar but that actually hangs over it as a kind of annual process. Then you have value stream 3. [...] So they have a number of different system types and plan them out over time. And that is also just a year-planning with systems".* The value streams differ in task characteristics and are planned based on different time spans. As mentioned before, since value stream 1 is following the V-model, its project phases influence the required planning characteristics. Value stream 2 is not related to this PGP process and therefore can be planned differently, as one of the technical engineers explained: *"for example project X from value stream 2, almost all tasks can be executed within the team. So you can manage your dependencies. I think that is a project where you could do well with SAFe".*

These **multiple projects and value streams**, indicated as a variable in the CLD (Figure 6), the different project phases, and the multiple sub-departments with contrasting work characteristics within the scrum team are positively affecting the task heterogeneity within the team. The higher this task heterogeneity, the higher the variation in task characteristics requiring conflicting planning requirements. These conflicting planning requirements within the scrum team make it more difficult to implement a consistent way of managing tasks. Moreover, as task heterogeneity increases, the ability to distribute tasks within the team becomes more difficult. As one of the scrum master/production engineers explained: *"If you look at scrum, one team should be able to take over work from each other. We can't do that at all, everyone really has their own specialty. The idea is that you actually work on one product, but we are stuck with different value streams. And per value stream, you already have 10 different projects going on".* As a result, the task planning control of the team decreases, thereby mitigating the balancing effect of the task management loop (Figure 8.

5.1.4 Reinforcing loop 4: The focus issue loop

The focus loop is triggered by the **project management approaches intertwining** and the **multiple projects and value streams** being handled simultaneously within the scrum team. As a result of the scrum teams being organized per product module, every project going through the department with an impact on that specific product module has to be handled by that specific scrum team. Moreover, as the department has a project flow of multiple projects from different value streams, the scrum teams will deal with multiple projects simultaneously, consequently increasing the **task heterogeneity in team configuration**. As one of the technical support engineers explained: *"Per KD, from the first phase of the design of a new system until transferring the system to volume production, we have written down as a department what we want to do. [...] These could be complete systems, options, or end-of-life projects, it has all become more diverse. So if you multiply the tasks by the number of parallel systems, you get so many subtasks that you have to manage in your scrum team, the focus is no longer there. [...] You do not have an overview of what you have to work on and what is important. [...] So if you have 50 tickets in scrum, you can't remember what was on your board, so you have to look back like: what did we agree on here? What was the intention here? Can I postpone this or not? [...] So the number of tasks is too large to keep focus".*

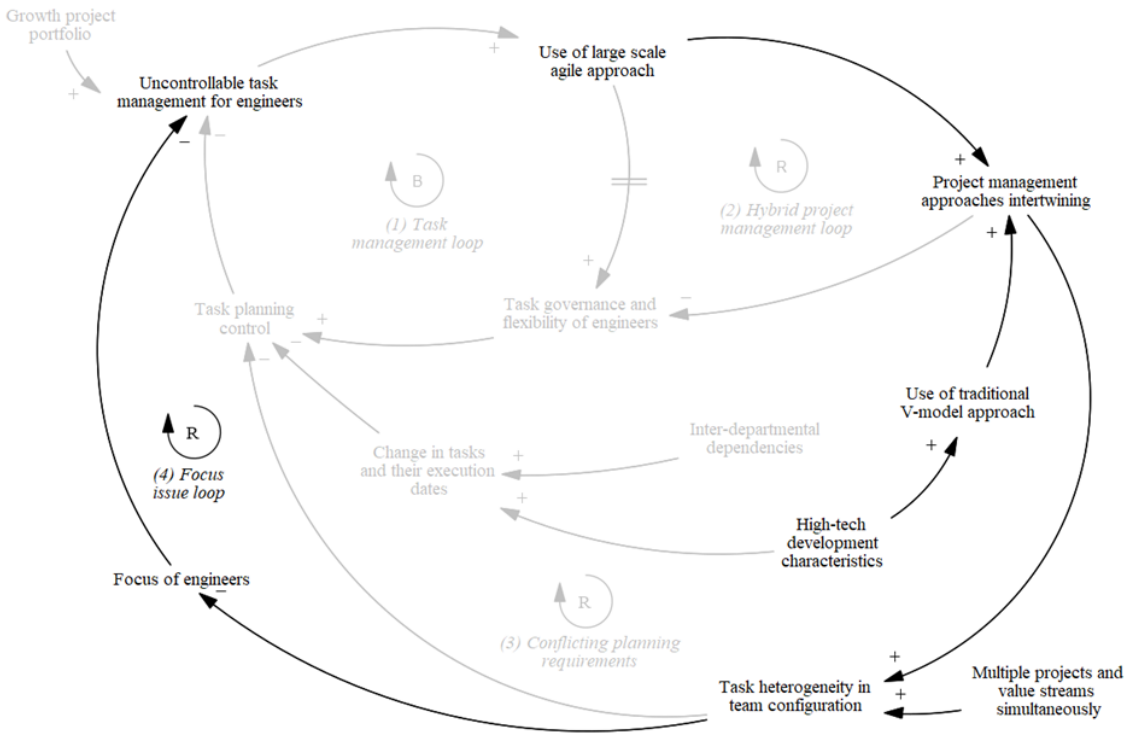

Figure 8: Causal loop diagram of the large-scale agile implementation with emphasis on the focus issue loop. The highlighted variables and arrows describe the focus issue loop and the effect on the task management loop. The 'B' and 'R' loops illustrated in the Figure define the loop's nature: either balancing (B) or reinforcing (R). The '+' and '-' indicate the polarity of the causal link. The double line intersecting the arrow denotes a delay

In addition to a lack of overview due to the high amount of tasks, the large number of projects enforces the engineers to switch often between these projects. As a consequence, switching between projects multiple times decreases the **focus of engineers**, indicated as a variable in the CLD (Figure 8). As one of the scrum master/production engineers explained: *"So during planning, we look at the whole backlog of 10 different projects like: 'okay where do we have to work on'. [...] The thing is that on a single day, you sometimes switch between 5 or 6 different projects and then you miss that focus. That is not ideal, but it is the reality"*. This large number of projects was also observed during the opening presentation of the PI planning event, which indicated twelve projects for value stream 1, seventeen projects from value stream 2, and six projects from value stream 3. As the projects differ in size and do not impact each scrum team, the variety clearly indicates the broad spectrum of different projects.

As the focus of the engineers decreases due to the high number and variety of tasks, this will affect the uncontrollable task management of engineers negatively. As the release train engineer

explained: *"Focus issues, and because of focus, the quality of what you are doing, our personal performance, team performance, it affects everything. [. . .] And that is just one distraction, that you were working on project A and you get distracted by project B, and then how much time it takes to go back to project A. Whereas we are working with 20 or so products and then on top of that we also have emails and chats and so on disturbing"*. The scrum team configuration, the number of projects and their diversity, and the resulting difficulty of focus for the engineers are characterizing the focus issue loop. This reinforcing focus loop mitigates the balancing effect of the task management loop on uncontrollable task management for engineers (Figure 8). In response to the high task heterogeneity, the engineers are specializing in particular value streams and projects within the teams, as one of the production engineers explained: *"In principle we have one production engineer who takes the lead in specific functional clusters. [...] If you want to know the division, engineer 1 is responsible for project 1. Project 2 is from engineer 2. Project 3 is from me. project 4, 5, and 6 are officially not divided yet"*. This was also explained by one of the product owners: *Some teams with other functional clusters, they are really like: I know a lot about functional cluster x but nothing from functional cluster y"*.

5.2 Commonality and robustness of the dynamic interactions within the CLD

The robustness of the CLD was tested by identifying the commonality of the interviewees on the interactions between the identified variables. In Figure 9, the commonality of each interaction is depicted using the thickness of the arrow. A thicker line indicates that more interviewees addressed the particular interaction during the interview. Table 3 further details which roles addressed which variable interactions. The CLD in Figure 9 clearly shows that the interactions from and towards task governance and flexibility of engineers, the effect of change in tasks and their interaction dates, and task heterogeneity in team configuration gained the most attention.

Table 3 clearly shows that the effects of the hybrid project management loop and the case setting on the task management loop are recognized by both the team layer and program layer. Furthermore, Table 3 shows that the effect of the conflicting planning requirements and focus issue loop on the task management loop is especially a team layer problem since these interactions are mainly discussed by the team layer and only by one person from the program layer.

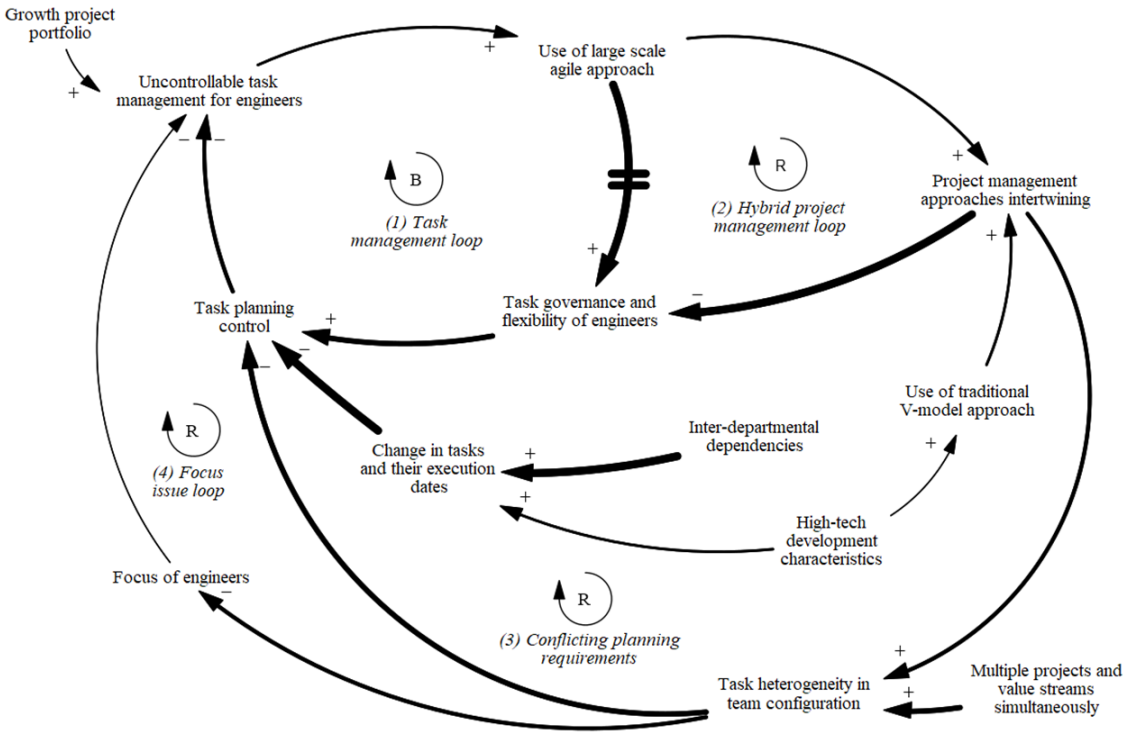

Figure 9: Causal loop diagram of the large-scale agile implementation and the commonality of the interaction dynamics between the variables. The thickness of the arrows corresponds to the commonality of the interaction. The 'B' and 'R' loops illustrated in the Figure define the loop's nature: either balancing (B) or reinforcing (R). The '+' and '-' indicate the polarity of the causal link. The double line intersecting the arrow denotes a delay

Table 3: Commonality table

Variable interaction	Count	Interviewee																					
		Team layer													Program layer						Line manager	Line Manager	Line Manager
		PE/SM	PE/SM	PE/SM	PE/SM	PE/SM	PE	PE	TS	TS	PO	PO	PO	RTE	PL	PL	PL	PM	PM	PM			
Task management loop																							
Growth project portfolio → Uncontrollable task management for engineers	6		✓					✓		✓						✓					✓		✓
Uncontrollable task management for engineers →Use of large scale agile approach	7								✓	✓						✓			✓		✓	✓	✓
Use of large scale agile approach →Task governance and flexibility of engineers	18	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓		✓	✓	✓	✓		✓	✓	
Task governance and flexibility of engineers →Task planning control	12	✓		✓	✓	✓	✓		✓	✓	✓	✓		✓		✓					✓		
Task planning control →Uncontrollable tasks management for engineers	9				✓	✓	✓			✓	✓	✓				✓			✓		✓		
Effect of case setting																							
Inter-departmental dependencies → Change in tasks and their execution dates	15	✓		✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓		✓		✓	✓			
High-tech development characteristics → Change in tasks and their execution dates	6				✓			✓	✓		✓				✓					✓			
Change in tasks and their execution dates → Task planning control	18	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓			✓	✓	✓	✓	✓	✓	✓	✓		
Task planning control →Uncontrollable tasks management for engineers	8		✓			✓			✓	✓	✓				✓		✓		✓				
Hybrid project management loop																							
Steering on a large scale agile approach → Project management approaches intertwining	7				✓						✓					✓		✓	✓		✓		✓
High-tech development characteristics → Use of traditional V-model approach	3																✓			✓	✓		
Use of traditional V-model approach → Project management approaches intertwining	8				✓	✓			✓		✓	✓				✓		✓	✓				
Project management approaches intertwining → Task governance and flexibility of engineers	19	✓	✓		✓	✓	✓	✓		✓	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓
Conflicting planning requirements loop																							
Project management approaches intertwining → Task heterogeneity in team configuration	9					✓	✓		✓	✓	✓		✓	✓								✓	✓
Multiple projects and value streams → Task heterogeneity in team configuration	12				✓	✓	✓	✓	✓	✓	✓			✓						✓	✓	✓	✓
Task heterogeneity in team configuration → Task planning control	12	✓		✓	✓	✓	✓			✓	✓	✓	✓	✓						✓			✓
Focus issue loop																							
Task heterogeneity in team configuration → Focus of engineers	9				✓	✓	✓		✓	✓	✓			✓								✓	✓
Focus of engineers →Uncontrollable task management for engineers	5				✓				✓		✓			✓									✓

6 Discussion and conclusion

In response to the lack of research on the dynamic mechanisms underlying the implementation of large-scale agile frameworks in a hardware setting, this study has set out to investigate these dynamic mechanisms underlying the implementation of a large-scale agile framework within a company developing high-tech products using a traditional V-model approach . By conducting an in-depth case study, the dynamics between the constructed themes were captured in a causal loop diagram (Figure 2). As this CLD model offers a theoretical explanation of the implementation of a large-scale agile framework in the particular case setting, it answers the research question of what mechanisms are at play and how they facilitate or hinder the task management of engineers. The findings of this study have various implications and are discussed in the following sections.

6.1 A dynamic perspective on large-scale agile implementation in a high-tech environment

Several findings stood out when researching the dynamic perspective. First, the CLD describes the balancing task management loop. This loop describes the intention to implement the large-scale agile framework, its perceived benefits on the task planning governance and flexibility of engineers, the effect of this planning governance and flexibility on task planning control which consequently reduces the uncontrollable task management for engineers. In addition, the model described a delay in the task management loop which is corresponding to a learning curve in the use of a large-scale agile approach. Furthermore, the model describes that factors such as the high-tech development characteristics and multi-departmental dependencies cause tasks and their execution dates to change which negatively impacts the effect of the balancing task management loop (Figure 2).

Second, it became evident that the implementation of the large-scale agile framework was prone to unintended consequences. To develop its high-tech products, LithoCo is executing a traditional project management approach. As the industrialization department also implemented the large-scale agile approach, both project management approaches started to intertwine. This hybrid project management loop, categorized by its contrasting planning philosophy, communication challenges, and mixed project management structures, was negatively affecting the task governance and flexibility of engineers. As a result, the reinforcing hybrid project management loop mitigated the balancing effect of the task management loop (Figure 2).

The increased level of project management approaches intertwining in combination with multiple projects and value streams in the company positively affected the task heterogeneity in the

team configuration. This increase in task heterogeneity within the team negatively affected the task planning control and focus of engineers. This reduction in the focus of engineers consequently affected the uncontrollable task management for engineers. As a result, these dynamics, defined as the reinforcing conflicting planning requirements loop and reinforcing focus issue loop, mitigated the intended balancing effect of the task management loop.

6.2 Theoretical implications

The findings of this study have multiple theoretical implications. First, as most research on hybrid project management is focusing on the identification of challenges, benefits, best practices, and guidelines (Reiff and Schlegel, 2022), this study contributes to the theory by identifying the mechanisms underlying the implementation of a large-scale agile framework in a company executing a traditional V-model. While multiple scientific articles are, despite the associated challenges, optimistic about the benefits of a hybrid project management approach (Cooper and Sommer, 2016,0), this study shows that implementing a hybrid project management approach is not that straightforward. By identifying and illustrating the dynamics of the intended and unintended consequences using a causal loop diagram, the complexity of implementing such a hybrid project management approach in a high-tech case setting is put into perspective.

Second, the intended effect of the balancing task management loop confirmed the findings of Schmidt (2018) who found that the implementation of an agile project management approach increased the planning overview, increased ability to react to changes, and increased transparency. Furthermore, it became evident that the expected benefits of the implementation of a large-scale agile approach are prone to a learning curve. This learning curve is depicted as a delay that was caused by challenges in requesting work between the program layer and the team layer, the difficulty in defining the 'definition of done' of features and tickets, insufficient role fulfillment, and the inability to estimate time to task completion. These challenges were also reported in research on software companies (Dikert et al., 2016; Edwards et al., 2019; Kasauli et al., 2020). This shows that this learning curve is not particularly hardware specific.

Third, this study shows how a large number of projects and different value streams increased the task heterogeneity within the team and consequently decreased the focus of the engineers. Literature shows that switching between different types of tasks is prone to switching costs (Monsell, 2003; Tregubov et al., 2017) and working on multiple projects decreases the time spent on value-adding tasks (Wheelwright and Clark, 1992). As a result, the teams being organized around sub-modules also negatively affects the task management of the engineers.

As can be seen in Table 3, nine interviewees mentioned the causal effect of task heterogeneity in team configuration on the focus of the engineers. Moreover, only five interviewees mentioned the effect of the focus of the engineers on the balancing task management loop. Although this effect did not have as much commonality as other dynamics within the CLD, it should not be neglected. Similar findings were reported by (Ovesen, 2012) who investigated a company applying scrum that was struggling to maintain the necessary focus as they were participating in multiple projects simultaneously. Solving this issue would require a change of the organizational setup.

While participating in multiple projects simultaneously conflicts with the principles of agile, it is also common in the hardware industry. Since physical products are prone to waiting times related to their physical nature and manufacturing companies often have many projects underway, engineers can work on other projects in the meantime to maintain a continious workflow (Cooper and Sommer, 2016). Since these waiting times do not exist in software development, the agile project management approach does not take this into account.

Fourth, this study contributes to the agile hardware literature by showing how the complexity of the product complicates the task management of the engineers.Atzberger and Paetzold (2019) mentioned that product complexity became more important with the development of hardware products in comparison to software. This was confirmed by the findings in this study since the high-tech products, consisting of multiple sub-modules and requiring high specific knowledge, complicated designing the agile teams. Ovesen (2012) categorized this challenge in designing the teams as the team distribution dilemma. Furthermore, the large size of the product and organization gave rise to inter-departmental dependencies.

Fifth, the findings showed that the effectiveness of the agile project management approach was mitigated by the changing tasks and their execution dates (Figure 4). This turned out to be an important subject since it was mentioned by 18 of the 22 interviewees (Table 3). An initiator of these changing tasks was the occurrence of ad-hoc work that would replace the planned work. Similar findings were done by Ovesen (2012) who found that companies experienced planning disturbances from the production facilities which forced the scrum teams to 'firefight'. To solve this, the company in the study of Ovesen (2012) established separate firefighting teams that could absorb most of the unforeseen disturbances.

A comparable concept was found in this study. The engineering scrum teams consisted of two engineering entities. While the production engineers were able to plan their tasks to some extend, the tasks of the technical support engineers were mainly based on 'fire fighting'. The difference

in this case study, in comparison to the case study of Ovesen (2012), was that those two engineering entities with contrasting task characteristics were combined in a single scrum team. The findings of this study show that the combination of these two engineering entities from different sub-departments increased the task heterogeneity in the team and resulted in limited task distribution between both engineering entities and contrasting planning requirements. This had a negative effect on the task planning control of the agile teams.

Sixth, the findings of the two project management approaches intertwining contributed to the hybrid project management literature stream by indicating a negative effect on the task governance and flexibility of the engineers (Figure 5). As can be seen in the commonality Table 3, 19 interviewees mentioned this relation, which makes it an important topic in this study. Findings related to this hybrid project management loop indicated communication challenges, the consequences of top-down planning on the engineers working agile, and the inability to do sufficient priority setting. While research of Bick et al. (2017) found that prioritization of tasks by the different teams complicated the inter-team dependencies, prioritization in this study was almost not done. Reasons for this lack of prioritization were related to the lack of skills of the product owners, the large number of projects, but more importantly, the lack of prioritization of these projects at the program layer.

Finally, this study answered to the call of Cooper and Sommer (2016) to research how the product owner's role would fit in with the traditional concepts of the program manager and project leader. This study elaborated on that by identifying role unclarity at the program layer and a challenge in implementing the product owner role. This latter was caused by the way of organizing the department, the structure of the scrum teams, the number of projects and value streams simultaneously, the high-tech nature of the product, and the size of the organization. As a result, the responsibilities and required tasks of the product owner required too much cognitive capacity to be handled by a single product owner. This resulted in the responsibilities and tasks of the product owner being shared with the product leader. This was done by giving the product owner the responsibility of internal communication with the team and the project leader the external communication with the stakeholders.

6.3 Managerial implications

In response to the findings of this study, multiple managerial implications are suggested. These implications are shown in the CLD (Figure 10). The implications are proposed to decrease the negative effects of the unintended dynamics that are accompanying the implementation of the large-scale agile framework.

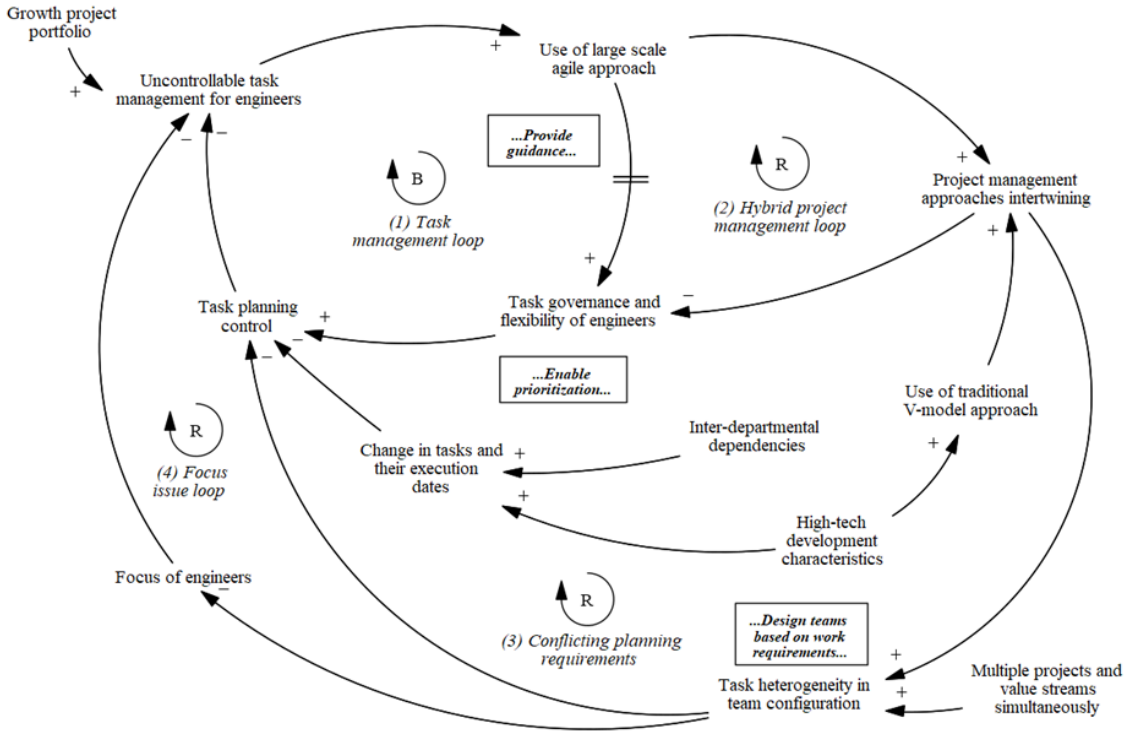

Figure 10: Causal loop diagram of the large-scale agile implementation within the case setting including managerial implications. The 'B' and 'R' loops illustrated in the Figure define the loops nature: either being balancing (B) or reinforcing (R). The '+' and '-' indicate the polarity of the causal link. The double line intersecting the arrow denotes a delay. The boxes indicate the managerial implication.

6.3.1 Provide guidance on large-scale agile implementation

The first proposed managerial implications are related to the learning curve of working with the large-scale agile framework (Figure 10). As challenges were experienced such as how to define work requests into features, defining the definition of done, role unclarity, and estimating how much time is needed for a task, they all have one thing in common: lack of experience and skills. Therefore, to deal with the lack of experience and skills, managers should focus on providing guidance on these topics.

First, defining work requests into features and defining a clear definition of done might be difficult since task specifications are not fully clear at that particular moment (Chapter 5.1.1). Nevertheless, the work-delivering team layer has to know what work is expected and the work-requesting program layer has to know if the delivered work meets the expectations. To smoothen this process of work request and delivery between the program and team layer, expectations have to be managed.

To facilitate this expectation management between both layers, emphasis should be put on the acceptance criteria of the features. When requesting the work, the program layer should define high-level acceptance criteria to communicate the expectations to the team layer. Furthermore, the team layer should also define its acceptance criteria for accepting a feature. This implies that certain things are required to be able to work on a particular feature. By using this two-way acceptance criteria communication, both parties learn to understand what the other party requires to complete their task. As most of the projects go through the same KD moments of the V-model, the type of tasks required for the projects can be considered comparable. As a result, defined acceptance criteria could be applicable to new projects.

Second, as the scrum master and product owner roles are subject to role unclarity, additional training on how to fulfill the product owner and scrum master role should be provided. Furthermore, training should contain an emphasis on how the high-tech multi-departmental environment affects the scrum master and product owner role. Furthermore, it is important that the implementation of the processes and introduced roles of the large-scale framework are fully supported by the higher management since the sufficient implementation of a new project management approach requires top-down support.

Third, as estimating the required time for a particular task is complicated, it can be improved by developing historical data on how much time was needed for a particular type of task and how much these time requirements could deviate. While tasks of different projects are never completely identical, collecting historical quantitative data on the time duration of certain types of tasks could provide valid estimates. Additionally, this data becomes more accurate as the amount of data grows.

6.3.2 Enable prioritization for the team layer

The second proposed managerial implication is to enable prioritization for the engineers (Figure 10). To provide guidance on estimating, prioritizing, and refining tasks that have to be completed somewhere in the future, the large-scale agile approach introduced a team and program backlog. This backlog is a tool that enables flexibility and governance in dynamics circumstances. Therefore, to be able to deal with such dynamic circumstances, the backlog should be carefully managed by the product owner.

As the inter-departmental dependencies and high-tech development characteristics of the industrialization department are causing tasks and their execution dates to change frequently, workload peaks and valleys occur, as described in chapter 5.1.1. While these changes mitigate the task planning control, efficient priority setting could mitigate this effect by filling the workload gaps and flattening the workload peaks. Unfortunately, effective priority setting is done insufficiently due to the lack of a sophisticated program backlog and therefore fails to provide planning control (Chapter 5.1.1 & 5.1.2).

To enable product owners to prioritize, the program layer should have a sophisticated backlog with the status of each project and the priority at that particular project stage. By doing this, the product owners can make choices on what tasks to work on first and keep the team backlog updates. Additionally, the program layer should provide features for future tasks upfront. When tasks move out of a sprint or even a PI, teams can drag these tasks in to fill the workload gap. Similarly, when additional tasks come in, the priority of these new tasks should be defined and the backlog refined accordingly. By doing this, product owners are enabled to manage and prioritize the team backlog and consequently keep the workload stable.

Furthermore, as the industrialization department is prone to multiple factors that can initiate changing tasks and their execution dates, planning updates are often required. Since the program layer is responsible for external project communication, as explained in chapter 5.1.2, it will receive critical information regarding project planning. Since the product owner is responsible for managing and prioritizing the team backlog, its ability to do so is depending on all information regarding project planning. Therefore the product owner's ability to be successful in its task is depending on the communication of changes in planning by the program layer.

To facilitate the product owner in managing and prioritizing the teams backlog, extra communication lines should be in place between the program layer and the product owners of the team layer. Since changes in tasks and their execution dates might occur more often than once a week,

these communications during the PO sync might be insufficient and reach the product owners too late. Therefore, these extra communication lines would enable the team layer to constantly update the product owner and vice versa. Additionally. these communication lines could be organized between the project leader or manager and the product owner, instead of with the engineers of the scrum team. In this way, the engineers of the scrum teams are released from the validation meetings with the team layer and can focus their time on executing tasks.

Finally, it is important to note that prioritization on the backlog can only work if management allows tasks to be delayed. As the delay of tasks might have consequences on the project's progress, teams lacking resources to complete particular tasks in time should be backed up by data providing evidence. Therefore, in order to play the prioritization game, the product owner and its scrum team should be able to estimate how much time they need for particular tasks. As this is currently lacking, teams should generate quantitative data on the task duration of tasks. By doing so, the teams can justify their work capacity towards the program layer and emphasize the need for prioritization when additional work comes in.

6.3.3 Design teams based on work requirements

As the conflicting planning requirements loop is having a reinforcing effect on the uncontrollable task management for engineers, managers should strive to mitigate this effect. As described in chapter 5.1.3, this loop is characterized by task heterogeneity in the team configuration. This task heterogeneity is characterized by multiple factors: having multiple roles from different departments in the team, project phase, value stream, number of projects, and scrum team configuration. To reduce task heterogeneity to a manageable level, multiple managerial implications are proposed related to the design of the teams.

First, the contradictory role of TS and PE within a single team should be considered. As the TS role is characterized by reactive work, they require a project management tool that facilitates this process such as Kanban. On the contrary, the PE role is characterized by more pro-active work, depending on the project phase. Therefore, they require a project management tool that facilitates this such as Scrum. Currently, while these two roles with contrasting planning requirements are now planning together within the same team on the same planning board, they appear to have limited shared deliverables 5.1.3. Managers should consider the benefits and challenges related to the combination or separation of PE and TS in a scrum team. As the separation of both roles would decrease the contrasting planning requirements, it would also decrease knowledge sharing between the roles. Therefore, if chosen for the separation of the roles from the teams, knowledge-sharing processes should be facilitated.

The scaled agile framework (SAFe) provides multiple team topologies to provide the teams to be organized around value (Koehnemann, 2022). One of these team topologies is the 'enabling team' which is responsible for supporting and assisting other teams in acquiring missing capabilities around a specific technical or product management area. Another team topology that is described is the 'complicated subsystem team', which is responsible for supporting and maintaining a part of the system depending heavily on specialist knowledge. Considering the contrasting planning requirements between both PE and TS but the strong need for knowledge sharing, a combination of the earlier mentioned enabling and complicated subsystem team could be made to split the two roles. In this case, TS would become the new enabling complicated subsystem team executing its own planning while still facilitating the required knowledge sharing between both teams (Figure 11).

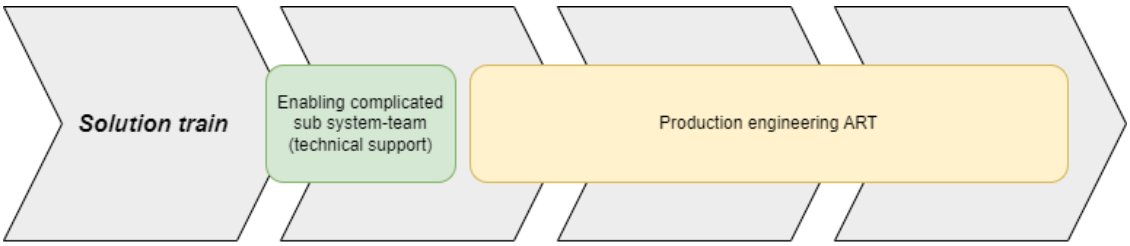

Figure 11: Graphical illustration of solution train consisting of a production engineering train with facilitating technical support team.

The second managerial implication regards the characteristics and quantity of tasks. As explained in chapter 5.1.3, the characteristics and quantity of tasks are defined by three factors. First, the scrum teams are dealing with three non-related value streams, each providing a number of projects with different characteristics. Second, as one of the value streams is following the V-model process, the phase of the project also affects the task characteristics. Third, simply the number of projects being managed. As multiple projects following the V-model are being managed, the project phases with contrasting task characteristics will overlap (Figure 7). This in combination with the number of projects provided by the other value streams is causing the contrasting planning requirements loop and the focus issue loop.

Mitigating the effect of the contrasting planning requirement loop and focus loop implies a split in tasks for the scrum teams. This can be either on the number of projects, the type of tasks within the project, or both. Realizing this split in tasks requires the scrum teams to be reorganized. While there are multiple possibilities to reorganize the scrum teams, each possibility provides its advantages and disadvantages. Table 4 shows the team construction options and their effect on

the project management variables. The effect of each team construction on a particular variable is indicated by either low, medium, or high. The exact description of each variable is further described in table 5, in appendice B.

As the teams can be organized based on competence, single project, project type, value stream, and process, each team construction contains its advantages and disadvantages. For example, organizing the teams based on competence would provide a highly continuous workflow since they have multiple projects to work on, the focus of engineers is low, and the diversity and number of projects are high since they will have to manage all projects for their competence. On the contrary, as organizing the teams based on a single project would provide a high focus and ability to adjust the planning to contrasting planning requirements, the continuous workflow of engineers is low since a particular project might not have work for each competence of the project. Moreover, whenever the project would be subject to a delay, engineers would not have other tasks to execute (low diversity and number of projects), consequently decreasing the task efficiency.

Organizing the teams based on the process of the project could be an option that is comparable with the organization based on competences. While the competence-based team organization is subject to the contrasting planning requirements of the different project phases, the process-based team organization mitigates this effect as it is focused on a specific phase of the project. The disadvantage of the process-based team organization is that, since the different project phases are only applicable to one particular value stream, additional teams will have to be formed for the other value streams. Another promising option could be to organize the scrum teams based on a value stream or specific project type. This option would balance the advantages and disadvantages of being organized around competences or per project (Table 4).

It should be noted that, while some variables might have a low rating within a particular team construction, additional actions can be taken to improve their effectiveness. For example, as the organization of teams based on project type and value stream would have a low to medium shared learning due to all the different competences within one team, additional cross-functional competences team could be formed to share experiences and discuss challenges. Furthermore, as one of the incentives to organize the teams based on value streams was to decrease the diversity and number of projects, they still might contain too many projects to handle. Therefore, a combination of a value stream and project-type organization could be made. This would imply the separation of value streams and simultaneously, within this value stream, a separation based on the project types.

As illustrated, selecting the most suitable scrum team configuration requires consideration of what

Table 4: Team construction options and its effect on multiple project management variables. Each effect is indicated with either low, medium or high.

Variable	**Team construction**				
	Competence based (Bundle of FC's)	**Project based**	**Project type based**	**Value stream based**	**Process based**
Diversity of projects	High	Low	Medium	Medium	High
Number of projects	High	Low	Medium	Medium	High
Focus of engineers	Low	High	Medium / High	Medium / High	Low
Shared learning	High	Low	Low / Medium	Low / Medium	High
Variety of required specific knowledge within the team	Low / Medium	High	High	High	Low / Medium
Continuous workflow	High	Low / Medium	Medium / High	Medium / high	High
Degree of contrasting planning requirements within team	High	High	High	High	Low
Ability to adjust planning to contrasting planning requirements	Low	High	Medium	Medium	High
Practicality of inter-departmental alignment	High	Low	Low	Low	High

variables are most valuable for the organization. Therefore, managers should carefully consider the options and their consequences. It has to be noted that the team organizations depicted in Table 4 assume that the team remains stable.

6.4 Limitations and future research

To understand the dynamics underlying the integration of the hybrid project management approach within a hardware environment, an in-depth case study was conducted. While this kind of study is particularly suitable for building theory inductively, it is subject to some limitations (Eisenhardt and Graebner, 2007; Yin, 2015).

First, as this in-depth case study used theoretical sampling to select a real-world case setting to build theory, generalizing the findings to other case settings may be limited. Nevertheless, as this study aimed to understand the mechanisms underlying the integration of a hybrid project management approach in a particular context, these mechanisms might also be transferable to other contexts implementing such a hybrid project management approach. By providing a certain level of methodology transparency, future research could replicate this methodology to other contexts and consequently validate these findings (Aguinis and Solarino, 2019).

Second, as the effect of bias on the validity of the study was minimized by providing interview data from multiple points of view, targeting interview questions on experiences in particular events and using multiple data sources to provide data triangulation, this study might still be subject to some degree of retrospective bias.

Third, as this study aimed to research the effect of a large-scale agile framework in general, the case setting was implementing the scaled agile framework. As this is a specific type of large-scale agile framework, the implementation of another specific type of large-scale agile framework could potentially show deviating findings. Meanwhile, as the types of large-scale agile framework have similar principles, it can be reasonably assumed that different large-scale agile frameworks will show comparable dynamics. Nevertheless, it would be interesting for future research to investigate comparable case settings that implemented a different large-scale agile framework.

Fourth, the investigated case study organised its scrum teams based on the sub-modules of the high-tech product. As the team configuration was found to be an important factor in the dynamics of the causal loop diagram, it would be interesting for future research to investigate the dynamics of case settings with different team configurations, such as proposed in chapter 6.3.3. By revealing these dynamics, the knowledge of what mechanisms can be expected during the integration of a hybrid project management approach can be expanded, consequently providing better integration guidance for practitioners.

6.5 Concluding remarks

As organizations are interested in combining the flexibility and adaptability of agile project management with the predictability and long-term planning of traditional project management, effectively integrating both approaches turns out to be challenging. Moreover, as agile methodologies are initially designed for the software development industry, applying them in a hardware context introduces additional challenges. The findings of this in-depth case study put these challenges into perspective by showing what integration dynamics occur and how they affect the desired outcome of such a hybrid project management implementation in a high-tech case setting. More specifically, the dynamic causal loop diagram illustrates and theorizes how implementing a large-scale agile approach triggers a desired dynamic but is mitigated by the undesired consequences of the multi-departmental hardware context characteristics and the agile and traditional project management approach intertwining.

References

Aguinis, H. and Solarino, A. M. (2019). Transparency and replicability in qualitative research: The case of interviews with elite informants. *Strategic Management Journal*, 40(8):1291–1315.

Alblas, A. and Notten, M. (2021). Speed is significant in short-loop experimental learning: Iterating and debugging in high-tech product innovation. *Decision Sciences*, 52(6):1364–1402.

Atzberger, A. and Paetzold, K. (2019). Current challenges of agile hardware development: What are still the pain points nowadays? In *Proceedings of the Design Society: international conference on Engineering Design*, volume 1, pages 2209–2218. Cambridge University Press.

Beck, K., Beedle, M., Van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., et al. (2001). Manifesto for agile software development.

Bick, S., Spohrer, K., Hoda, R., Scheerer, A., and Heinzl, A. (2017). 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.

Boehm, B. and Turner, R. (2004). Balancing agility and discipline: Evaluating and integrating agile and plan-driven methods. In *Proceedings. 26th International Conference on Software Engineering*, pages 718–719. IEEE.

Böhmer, A. I., Hugger, P., and Lindemann, U. (2017). Scrum within hardware development insights of the application of scrum for the development of a passive exoskeleton. In *2017 International Conference on Engineering, Technology and Innovation (ICE/ITMC)*, pages 790–798. IEEE.

Brandl, F. J., Kagerer, M., and Reinhart, G. (2018). A hybrid innovation management framework for manufacturing–enablers for more agility in plants. *Procedia CIRP*, 72:1154–1159.

Bricogne, M., Le Duigou, J., and Eynard, B. (2016). Design processes of mechatronic systems. *Mechatronic futures: Challenges and solutions for mechatronic systems and their designers*, pages 75–89.

Chell, E. (2004). Critical incident technique'in cassell, c. and symon, g.(2004). essential guide to qualitative methods in organizational research.

Conboy, K. and Carroll, N. (2019). Implementing large-scale agile frameworks: challenges and recommendations. *IEEE software*, 36(2):44–50.

Conforto, E. C. and Amaral, D. C. (2016). Agile project management and stage-gate model—a hybrid framework for technology-based companies. *Journal of Engineering and Technology Management*, 40:1–14.

Cooper, R. G. (2017). Idea-to-launch gating systems: Better, faster, and more agile: Leading firms are rethinking and reinventing their idea-to-launch gating systems, adding elements of agile to traditional stage-gate structures to add flexibility and speed while retaining structure. *Research-Technology Management*, 60(1):48–52.

Cooper, R. G. and Sommer, A. F. (2016). The agile–stage-gate hybrid model: a promising new approach and a new research opportunity. *Journal of Product Innovation Management*, 33(5):513–526.

Cooper, R. G. and Sommer, A. F. (2018a). Agile–stage-gate for manufacturers: Changing the way new products are developed integrating agile project management methods into a stage-gate system offers both opportunities and challenges. *Research-Technology Management*, 61(2):17–26.

Cooper, R. G. and Sommer, A. F. (2018b). Agile–stage-gate for manufacturers: Changing the way new products are developed integrating agile project management methods into a stage-gate system offers both opportunities and challenges. *Research-Technology Management*, 61(2):17–26.

Copola Azenha, F., Aparecida Reis, D., and Leme Fleury, A. (2021). The role and characteristics of hybrid approaches to project management in the development of technology-based products and services. *Project Management Journal*, 52(1):90–110.

Costantini, S., Hall, J. G., and Rapanotti, L. (2021). Using complexity and volatility characteristics to guide hybrid project management. *International Journal of Managing Projects in Business*.

Digital.ai (2022). Digital.ai. Accessed on November 2, 2022.

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.

Drutchas, J. and Eppinger, S. (2022). Guidance on application of agile in combined hardware and software development projects. *Proceedings of the Design Society*, 2:151–160.

Ebert, C. and Paasivaara, M. (2017). Scaling agile. *Ieee Software*, 34(6):98–103.

Edison, H., Wang, X., and Conboy, K. (2021). Comparing methods for large-scale agile software development: A systematic literature review. *IEEE Transactions on Software Engineering*.

Edwards, K., Cooper, R. G., Vedsmand, T., and Nardelli, G. (2019). Evaluating the agile-stage-gate hybrid model: Experiences from three sme manufacturing firms. *International Journal of Innovation and Technology Management*, 16(08):1950048.

Eisenhardt, K. M. and Graebner, M. E. (2007). Theory building from cases: Opportunities and challenges. *Academy of management journal*, 50(1):25–32.

Eklund, U. and Berger, C. (2017). Scaling agile development in mechatronic organizations-a comparative case study. In *2017 IEEE/ACM 39th International Conference on Software Engineering: Software Engineering in Practice Track (ICSE-SEIP)*, pages 173–182. IEEE.

Eklund, U., Holmström Olsson, H., and Strøm, N. J. (2014). Industrial challenges of scaling agile in mass-produced embedded systems. In *Agile Methods. Large-Scale Development, Refactoring, Testing, and Estimation: XP 2014 International Workshops, Rome, Italy, May 26-30, 2014, Revised Selected Papers 15*, pages 30–42. Springer.

Elia, G., Margherita, A., and Secundo, G. (2021). Project management canvas: a systems thinking framework to address project complexity. *International Journal of Managing Projects in Business*, 14(4):809–835.

Garzaniti, N., Briatore, S., Fortin, C., and Golkar, A. (2019a). Effectiveness of the scrum methodology for agile development of space hardware. In *2019 IEEE Aerospace Conference*, pages 1–8. IEEE.

Garzaniti, N., Fortin, C., and Golkar, A. (2019b). Toward a hybrid agile product development process. In *Product Lifecycle Management in the Digital Twin Era: 16th IFIP WG 5.1 International Conference, PLM 2019, Moscow, Russia, July 8–12, 2019, Revised Selected Papers 16*, pages 191–200. Springer.

Gemino, A., Horner Reich, B., and Serrador, P. M. (2021). Agile, traditional, and hybrid approaches to project success: is hybrid a poor second choice? *Project Management Journal*, 52(2):161–175.

Gioia, D. A., Corley, K. G., and Hamilton, A. L. (2013). Seeking qualitative rigor in inductive research: Notes on the gioia methodology. *Organizational research methods*, 16(1):15–31.

Gubinelli, S., Cesarotti, V., and Introna, V. (2019). The evolution of project management (pm): How agile, lean and six sigma are changing pm. *The Journal of Modern Project Management*, 7(3).

Kasauli, R., Knauss, E., Horkoff, J., Liebel, G., and de Oliveira Neto, F. G. (2021). Requirements engineering challenges and practices in large-scale agile system development. *Journal of Systems and Software*, 172:110851.

Kasauli, R., Knauss, E., Nakatumba-Nabende, J., and Kanagwa, B. (2020). Agile islands in a waterfall environment: Challenges and strategies in automotive. In *Proceedings of the Evaluation and Assessment in Software Engineering*, pages 31–40.

Koehnemann, H. (2022). Applying safe to hardware development. scaled agile framework. Accessed on October 18, 2022.

Leffingwell, D. (2016). Safe® 4.0 reference guide: Scaled agile framework® for lean software and systems engineering. *Google Scholar Google Scholar Digital Library Digital Library*.

Monsell, S. (2003). Task switching. *Trends in cognitive sciences*, 7(3):134–140.

Mule, S., Plateaux, R., Hehenberger, P., Penas, O., Patalano, S., and Vitolo, F. (2020). A new agile hybridization approach and a set of related guidelines for mechatronic product development. In *Product Lifecycle Management Enabling Smart X: 17th IFIP WG 5.1 International Conference, PLM 2020, Rapperswil, Switzerland, July 5–8, 2020, Revised Selected Papers 17*, pages 618–633. Springer.

Ovesen, N. (2012). The challenges of becoming agile: Implementing and conducting scrum in integrated product development.

Pratt, M. G. (2009). From the editors: For the lack of a boilerplate: Tips on writing up (and reviewing) qualitative research.

Putta, A., Paasivaara, M., and Lassenius, C. (2018). Benefits and challenges of adopting the scaled agile framework (safe): preliminary results from a multivocal literature review. In *Product-Focused Software Process Improvement: 19th International Conference, PROFES 2018, Wolfsburg, Germany, November 28–30, 2018, Proceedings 19*, pages 334–351. Springer.

Reiff, J. and Schlegel, D. (2022). Hybrid project management–a systematic literature review. *International journal of information systems and project management*, 10(2):45–63.

Schmidt, T. S. (2018). Agile development of physical products: An empirical study about motivations, potentials and applicability.

Sommer, A. F., Slavensky, A., Nguyen, V. T., Steger-Jensen, K., and Dukovska-Popovska, I. (2013). Scrum integration in stage-gate models for collaborative product development—a case study of three industrial manufacturers. In *2013 IEEE International Conference on Industrial Engineering and Engineering Management*, pages 1278–1282. IEEE.

Špundak, M. (2014). Mixed agile/traditional project management methodology–reality or illusion? *Procedia-Social and Behavioral Sciences*, 119:939–948.

Sterman, J. (2002). System dynamics: systems thinking and modeling for a complex world.

Sterman, J. D. (2001). System dynamics modeling: tools for learning in a complex world. *California management review*, 43(4):8–25.

Thesing, T., Feldmann, C., and Burchardt, M. (2021). Agile versus waterfall project management: decision model for selecting the appropriate approach to a project. *Procedia Computer Science*, 181:746–756.

Tregubov, A., Boehm, B., Rodchenko, N., and Lane, J. A. (2017). Impact of task switching and work interruptions on software development processes. In *Proceedings of the 2017 International Conference on Software and System Process*, pages 134–138.

Uludağ, Ö., Philipp, P., Putta, A., Paasivaara, M., Lassenius, C., and Matthes, F. (2022). Revealing the state of the art of large-scale agile development research: A systematic mapping study. *Journal of Systems and Software*, page 111473.

Vinekar, V., Slinkman, C. W., and Nerur, S. (2006). Can agile and traditional systems development approaches coexist? an ambidextrous view. *Information systems management*, 23(3):31–42.

Walrave, B., Dolmans, S., van Oorschot, K. E., Nuijten, A. L., Keil, M., and van Hellemond, S. (2022). Dysfunctional agile–stage-gate hybrid development: Keeping up appearances. *International Journal of Innovation and Technology Management*, 19(03):2240004.

Weichbroth, P. (2022). A case study on implementing agile techniques and practices: Rationale, benefits, barriers and business implications for hardware development. *Applied Sciences*, 12(17):8457.

Wheelwright, S. C. and Clark, K. B. (1992). *Revolutionizing product development: quantum leaps in speed, efficiency, and quality*. Simon and Schuster.

Yin, R. K. (2015). *Qualitative research from start to finish*. Guilford publications.

Zasa, F. P., Patrucco, A., and Pellizzoni, E. (2020). Managing the hybrid organization: How can agile and traditional project management coexist? *Research-Technology Management*, 64(1):54–63.

Appendices

A Company organizational structure

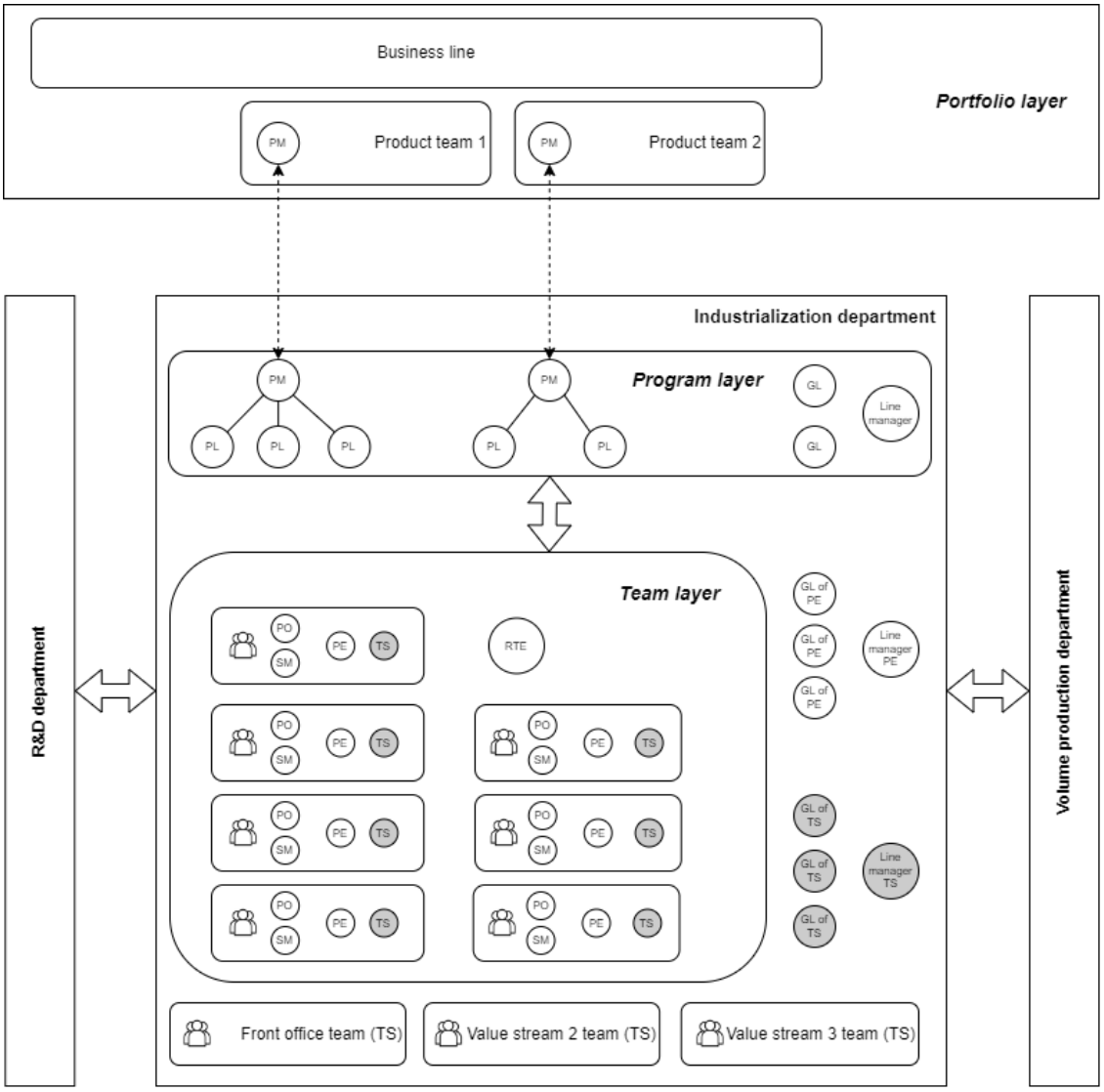

Figure 12: Graphical illustration of the company's organizational structure

B Definition of variables

Table 5: Explanation of the project management variables that are affected by the scrum team construction.

Variable	Explanation
Diversity of projects	The variation of projects in terms of required input that have to be handled by the teams.
Number of projects	The quantity of project that have to be handled by the teams.
Focus of engineers	The extent to which engineers perceive a focus when executing their set of tasks.
Shared learning	The ability of the engineers within the team to share experiences and consequently produce team knowledge.
Variety of required specific knowledge within the team	The degree of specific knowledge that is required within the team to deal with the allocated set of tasks.
Continuous workflow	The degree to which the scrum team is able to execute and complete a constant stream of tasks.
Degree of contrasting planning requirements within the team	The extent to which the scrum teams requires different planning characteristics to effectively deal with the set of tasks.
Ability to adjust planning to contrasting planning requirements	The ability of scrum teams to adjust their way of planning according to the planning requirements.
Practicality of inter-departmental alignment	The extent to which teams from the industrialization department can effectively align with their counterparts, considering the competence-based organization of teams throughout the company.

C Interview questions

C.1 Introduction

Before the interview started, the goal of the research and the objective of the interview was explained to the interviewee. Permission was asked to record the conversation and matters regarding confidentiality and anonymity were addressed. When the interviewee did not have questions, the interview started.

C.2 Interviewee information

1. What is your role within the department?
2. How much experience do you have within the department?
3. Could you tell me more about what your role entails?
4. How do you experience working with the large-scale agile framework?

C.3 Topic directions

Since this study executed semi-structured interviews, topic directions were used to guide the interview. The choice of topics was dependent on the interviewee. Since a high variety of roles was interviewed, the most suitable topic directions were chosen for the particular role. Follow-up questions were asked within each topic.

Topics:
- Large-scale agile processes
- Information flows
- Planning of work
- Function roles
- Structure of the department
- Structure of the teams

C.4 Wrap up

After completion of the topics, the interviews were ended by asking the interviewees if they wanted to share additional information regarding the topics that was missed during the questions.

D Codebook of data structure construction

Table 6: Codebook of data construction.

Second order concept	First order concept	Exemplary quote
Growth project portfolio	Increasing number of projects simultaneously	What you saw over time is that the portfolio of projects grew enormously. As a result, the engineers started participating in those projects with a fraction of their full time equivalent. Plus, there was a lot of inefficiency in it, those engineers were more in meetings than they were busy with engineering work.
		At some point, we started to introduce 29 products per year with an option here and an option there. And this is only for competence 1, and that is for all competences, and this is only for competence 2, and then it became too complicated to do it with project leaders and managers. That was the moment we started to do scaled agile.
		If we did not do it, we wouldn't have been able to do the work of today anymore. It is so much. We did not see it coming. A lot of work is coming. We have to deal with that somehow.
Uncontrollable task management for engineers	Inconsistent workload	And then you get all those weird peaks and valleys of workload. While scrum works well when you can decide for yourself when you are going to do something or not. Then you can spread the work out nicely. Because I think that we have enough people for the work that is being asked, but it is all asked at the same time, and then nothing is asked for a while. Then you do not get any input and you can't continue with your work.
	Growth in number of engineers	2018 we started with large scale agile. We first did a pilot in a relatively small product and that worked out so we started to roll it out across the whole department. At that point, the department had 100 people, now it is 250 I think. So we also experienced some growth spurts in the meantime.
		I think that we have grown to such an extent in terms of organization and in terms of the amount of products for which you still need help. That you need some kind of governance-like something. And I think this could work quite well honestly.
	Lack of overview on required tasks	Before, we had to know which work we had to do. Each week like, each day, you would have to choose what work to do on your own. You have no idea, you would check on the KD calendar, but it was a mess. People were missing deadlines. And there was actually nobody who would know if they were missing it or not. Because there was also no way of checking where all the teams were.
	Uncontrolled ad-hoc requests towards engineers	Sometimes I make the comparison between PE and mothers. A child needs its mother for everything and that is also a bit with PE. Be it for structural issues or this or that, they all need us. So what you had at a certain point is that those engineers could no longer work because every half hour, there was a project leader at someone's desk. And of course, the most at the best, and they did not get any work done at all anymore, they were distracted all the time.
		Well, we had a serious problem. The PL's were at the engineers' desk every day. Those engineers went crazy.
		Yes, it was sold at the time like: yes, then a product leader employee would come to your desk ad hoc and then you had to deliver some work at once and that was apparently a problem. I never experienced that as a problem, it actually worked fine for me. It was also less busy then so I don't know if that would still have gone well if we had still done it that way.
	Project leaders and managers championing own project	Plus, there was a second problem which was that every program manager or project leader thought that his own project was the most important. So it was often not clear to those engineers like: what is the most important project that I have to work on?
		At the moment, they don't operate in such a way. They are more allocated to a project and have to bring that project to a successful completion from a factory point of view. It is also because of this, if their project falls behind, they will scream louder to try to get their project done, still get done, and come to a successful completion. And as a result, their interests are not always, well often not, the same as those of the larger team and that also frictions on all sides.
Use of a large-scale agile approach	Requiring efficiënt task performance and planning flexibility	That was the moment we said, how can we organize this more efficiently? What methods can we use in the dynamic world of this company, where plans change often?
		To be competitive, that time-to-market is so important that you are always in that tension area. Planning well is not possible so whatever we do, we need a planning system in which we can be flexible and that we can adapt. SAFe should be suitable for that in particular.
		So basically you make a rough plan and you keep looking: where are we now? And then you adapt. [...] We have the same now, we are going to do validation. And then things are found like: we found this, this is good, this is not good, and based on that you have to make a new plan.
	Learning curve of working with large-scale agile	The hardest part was how we would make the translation from our project plan to what the engineers are asking. In the beginning, we requested too much and too detailed work. When you go the other way, you request too less. When you have an engineering team that is not so experienced, it can happen that things are missed. So as project leader, we are constantly looking for the balance to what level of detail of information we give and ask.
		When project leaders and project managers come to us, these are all high-level requests: we want the procedures to be validated. But what do you mean by that, what do you want us to do? [...] The definition of done is missing there.
		what I think what the struggle is for them is that they were just told one day that they were going to be PO without being told what necessarily the roles and responsibilities are and what they are actually supposed to do, what they are supposed to focus on, what their role is within the team. And then they were told to just figure that out within the team which I think is very unfair because we also don't know what appeals, what the role is going to be. So I think they really should have given proper training, and they should have been given like: this is your role, these are the tasks you are supposed to do, this is what you are supposed to do for the teams.
		I think that, as a team, we can improve by making good tickets. By making a clear definition of done so that the work becomes transferable. I think that will be of added value.
	Seperating engineers from project leaders	Looking per product, what work do we have in the coming quartile? Visualizing the work in blocks of work, called features, and bringing these features to the engineering teams who will detail out what is necessary to complete the feature successfully.
		Because we said, that engineer is getting questions from everywhere and the first thing we want to do is centralize those questions.
		We looked at different structures and came to the conclusion, let's centralize the engineering community and bring the work prioritized to them. Eventually, we did this by splitting the engineering teams
	Challenge in estimating required time for task completion	There is no transparency in how much effort it actually takes to do a task. And how full the team actually is.
		So my problem currently is, I don't have data or my scrum masters have qualitative comments, but they don't have data to back it up with and I would say to them, data is your weapon and your shield. That if you say to a stakeholder: our predicted velocity for the next sprint says we can do 70 points and if we look at just everything you have requested it is 120 points, so tell us what is the bottom priority and we will drop that.
		The downside is that it is always difficult to clearly define how much we can plan.
		As far as we are concerned, we are not yet ready to extract data from it and learn and be able to say; well, we get a new product, it's an product X alike product. Okay let's see how many story points and thus how much time did we actually spend on that project back then? Yes, then you can already make a prediction for the next project about how many resources you need. We aren't there yet.

Second order themes	First order concept	Exemplary quote
Task governance and flexibility of engineers	Large scale agile processes enabling planning governance	The power of this process is that once a quartile, we gain an overview on what work is coming in the next quartile, breaking this work into working blocks of 2 weeks. The teams have their daily stand-up meetings to guide and discuss the progress of the work they planned. That you have the PO sync meeting that, when priorities change, management gets the chance to guide this.
		'We have the daily stand-up. So if something new comes up during a sprint then you can ask people during this daily stand-up like: I don't have time for this, can someone take this up for me?
		It provides me with clarity in such a big organization. [. . .] Now I have a nice program board to plan together, it is nice to be able to fall back on that.
	Lack of priority setting	If a PO receives an ad-hoc request, then you should also say, or raise the question, what is the priority of this item versus the list of my backlog and what is going to go out. I haven't heard that kind of questions in the last 6 months happening
		We are not playing that game good enough. [...] I haven't had to make a choice anywhere in the past quarter whether we don't do A or B. No one has been to my desk. Not a PO has looked up escalation mode. So we don't play that game.
		And there is also something that, from the program, PMs and PLs sides, there is also wrong. Because there is a list of priorities, they are divided by four types of prio's. But underneath those priority types, you have 5, 6,7 or 8 products in between. So if you take all the prio ones, you are full. You will not do the rest. So also there, how the priority setting is done, it needs to be improved.
	Top-down planning	The portfolio layer determines which products we will deliver in which time span. [...] And when that machine is there and the portfolio layer says: that has to continue, then we have to make sure that we do our work on that machine. And then you get all those weird peaks and valleys of workload. While scrum works well when you can decide for yourself when you are going to do something or not".
		Yeah, your planning is filled in by the business line instead of being filled by the engineer that delivers you work. And he is filling in his planning also by the business line and those dates sometimes don't align. And what do you do then?
Task planning control	PI planning prone to shifting tasks	The PI event. We have to estimate 4 of 5 weeks upfront what has to take place at which moment in time. We have our delivering parties, which are different functional clusters from the D&E department who all have their own PI events. [. . .] Besides, you have a business line that basically plans everything in the best-case scenario. [...] That means that a lot can go wrong in planning at many different functional clusters before it reaches us. [...] So I have the feeling that we pretend to be very good at planning, but the PI event, all effort we put in there can be different within a month.
		So in that case, the only thing really that changes is the whole PI event. Which I think is nice because you get to see what is expected for the next three months but again, that kind of changes.
	Multiple processes to align project status	I like how we use the SAFe rituals, so the meeting structure, to have good discussions. [. . .] You are in a certain cadence talking to each other and also helping each other with the actions that need to be done. Whereas in a traditional way of project management, you are not all sitting together to discuss this, then it's often 1 on 1, and this happens or not.
	Creating overview	No, so what ends up happening. The PI planning is great in understanding the scope of the three months. Cause without it, I am sure that like, we complain about it but nobody has a better alternative right. Cause how else do I know what I am gonna work on in the next three months. Then I have to do it on 2 weeks basis, and trust me, the teams are not prepared for that. So that's great that you get that kind of overview.
		At least when we first introduced scrum, the first thing was the Kanban board. You finally were receiving what you had to do. That was the first big point.
	Large-scale agile guarding maximum work capacity of engineer	The engineers, they can now estimate themselves how much time is required for certain tasks and push back when too much work is asked.
	Managing plan modifications	That is often the discussion, should you plan ad-hoc work? Should you assume that 20% of your time is reserved for things you had not foreseen?
		If I am not mistaken we plan 56% of our capacity for plannable work and then we leave 44% for random issues that arise. So that is pretty much isn't it? So then you actually have 50% that we spend on issues that we encounter later and that immediately indicates how difficult planning is.
	Lack of task distribution	what a scrum team is supposed to be is that it is diverse, any person can pick up any tickets. But PE and TS are very split in responsibilities. And PE people want to do TS work because we want to get technical knowledge. TS people do not want to deal with any of the PE don't want the administration. That is where the struggle is for us work, they don't want the administration. That is where the struggle is for us
		Some teams with other functional clusters, they are really like: I know a lot about functional cluster x but nothing from functional cluster y.
		In principle we have one production engineer who takes the lead in specific functional clusters. [...] If you want to know the division, engineer 1 is responsible for project 1. Project 2 is from engineer 2. Project 3 is from me. project 4, 5, and 6 are officially not divided yet"
	Department not having full control over planning	You have to adjust constantly. People want to do what they are good at, their work, not constantly having to adjust their planning. Unfortunately, that it is the fact that everyone has to deal with.
		Every 2 hours you could open de planning and see that the next start was posponed again by a day because of material issues or other things. [. . .] Long story short, it takes a while to accept that you as a PL, or PM or even whole LAB, that you do not have influence.

Second order themes	First order concept	Exemplary quote
Project management approaches intertwining	Mix of traditional and agile project management approach	So, the teams here are working, or trying to work , on a scaled agile way of working. The program layer is only looking towards KDs and saying: you need to do this otherwise you don't meet my timeline. But that is also because above them, the portfolio layer is also working like that. Even though, they have all their sectors and the D&E department working on SAFe, they don't want to do it. So the top is still working in a waterfall way of working.
		Yeah, actually it all comes from a waterfall planning, which we've always done. And you can also see that it is actually partly managed that way. The D&E department is already applying SAFe in many places. But we don't manage the portfolio in the SAFe way. That's just how it used to be.
	Not completely changing organizational structure	I think that when they made the teams, they didn't want to make big steps on the changes. So they stick into at least knowledge-based teams. And the only thing they did was to mix PE and TS in one team. That is the only thing they did. But it's not like: with one team you will be able to deliver one product.
		We have a bit of a double system. You have the sprint review and validation meeting. You have the core team meeting and PO sync, which also have some overlap.
	Role unclarity	We have not yet propely defined what the differences are between the PO and PL role and who should do what. We encounter that regularly. What does a PL expect from a PO and vice versa.
		In this department, PLs or epic owners/business owners, do not know what their role should be exactly. Definitely, they do not know what to do during the PI events. Why are we here? Why are we wasting our time? You can hear that back a lot. Why do we need sprint reviews? What is our role in the sprint reviews? So they don't understand the framework, they don't understand the way of working. That's from the PL's side.
		In principle, our project leaders do not exist in SAFe. So there the discussion was: how is this going to work?
	Multiple sub-departments within department	You have the scrum teams that have PE people and TS people, but they are actually two seperate departments. So TS people have different managers than PE people. [...] So in that sense you still have the traditional structure of those departments.
		We have PE and TS in a team, that cooperation does not go well everywhere. [. . .] It is true that TS participates in all the rituals, but the need for them is much less to really be part of it. Collaboration is also less relevant than for PE. And that is of course difficult if you say you are from one team but still feel a kind of split.
		So their managers actually should get together and say: yes we have our people in all the teams but you make goals together, it's not that TS has that separate. TS goals and PE separate individual goals, okay, but team targets should be the same. So it's not only that the problem is existing because there are 2 very different roles in each team, its also the way management is approaching is, from the leadership, when they are saying; you are separate.
	Teams organized per set of functional clusters	What you see is that the whole company is functional cluster-based, so competence-based. [...] So on a competence-base you have your alignment. You have your fixed counterparts on competence bases that you can keep stable.
		We are constantly looking for a balance because there are also disadvantages. Because I mean certainly if you are in such a competence team, you get less of all the total product overviews
Inter-departmental dependencies	Requiring work input from the D&E department	Yes, we are not in the driver's seat of our dependencies. We receive documents for review, those documents. . . you know they are written by someone, but they arise spontaneously for our scrum team. You don't know when, and you don't know how big they are. And suddenly you get an email: 'Listen, I made this product, this document and you received a ticket that you have to review that document.
		We really depend on that. So in some cases we can't do our job because we don't have input from the D&E department yet. Yes that's a bit of a given.
	Complex and insufficiënt communication and allignment	While that if you look, we have our deliverers which are many different functional clusters from the D&E department that all have their own PI events. They all have their own planning processes. Most via SAFe, some not even that. But there is no link between all those SAFe events, so to speak.
		The link between all those FCs in the D&E department that all have their own SAFe events and all that coordination between them and us, there is just such a mismatch between them, you are so dependent on that. I don't see us solving that.
	Product and planning flaws resonate through departments	You work very much in the service of the product you as a PL are responsible for. And the teams you have, they also work with that same planning. So if something is delayed in the master plan, it usually slows down the planning of the industrialization department automatically.
		The real planning work is done by the people themselves who have to deliver things, for which they usually have the agile, safe way of working. I think that's fine. but the biggest problem I see is in the real timeline already at the beginning of the project, deliberately making the planning a bit too "wishful thinking' and consciously putting pressure on the organization, so to speak. And that pressure is getting to the D&E department and of course it seeps through to us.
	Dependency on factory availability	The start date of machines and modules, they change a lot. So the planning in the factory, that we depend on, does change a lot. That doesn't make it any easier for us.
		What you hear more often, is that it is agreed in advance: okay we build the machine in a day shift because then we as the industrialization department can validate. But then, two weeks before it starts, the factory says that they can't do it and then the validation has to go to another machine.
Change in tasks and their execution dates	Task delivery dates shifting	We try to plan as realistically as possible, of course, but yes, there are so many factors that influence those dates. That it often shifts or deviates.
		So we make a schedule for sprint 1 which is usually correct and sprint 2 is still reasonable but after that, it decreases. Then there are things that have been brought forward or that we can throw out. It is now almost sprint 4 so there are now quite a few tickets that we created during the PI event of which we say, throw it away, that won't happen anymore. Or they have been postponed to the next quarter and then other work will take its place.
	Ad-hoc work	There was a system in which part 1 crashed into part 2. Multiple parts of the system were broken, so that has to be repaired in the working center. At that point, materials need to be arranged: a repair order needs to be made so that the broken part is built out, work center capacity has to be there, the right tooling is there, an action plan is present, all those things have to be arranged. [...] So when it happens, it has to be managed immediately because the system is on hard down and already too late. So you need to drop everything to take that up.
		In the end, it was such an escalation that a particular module did not work, it was not usable and then you get into an escalation which means that people from your team are spending at least an hour a day in a meeting. If you then start thinking that you are in meetings for 1 hour and preparing for an hour with 2 persons, then you are already on 4 hours with your team per day. [...] So that eats a lot of your ad-hoc capacity.

Second order themes	First order concept	Exemplary quote
High-tech development characteristics	Product containing physical components	Highlevel the planning is right, but it is in the small nuances. Because its such a large system, such a complex machine with so much work preparation, there will always go something wrong.
		In that testing block, new parts are build in, software is installed and new tests are runned. It can be the case that parts aren't there yet, that the software isnt finished yet, or that the machine got delayed by a previous testing block.
	Product buildup requiring physical space	We are the industrialization department but we operate in a factory environment where dozens of systems start per month. This includes our pilots and say this are 5 pilots in six months' time. It's not like you can say: 'guys our pilot has priority over everything'.
	Specific knowledge required	Yes they do go on the floor but it's a lot. That really takes time. It is quite complex and some modules are even easier to oversee than if you really have a module X or something that is really throughout the system.
		If you consider a scrum team, for example, scrum team X, you will need someone with knowledge about functional cluster X, someone with knowledge about functional cluster Y, and someone with knowledge about functional cluster Z. To cross-train all of that so that everyone is knowledgable enough and to allocate one person to project 1 and one person to project 2, then you will lack sharpness everywhere.
		If you look at scrum, 1 team should all be able to more or less take over the work from each other. We can't do that at all, everyone really has their own specialty.
Use of a traditional v-model approach	Products following key decision moments based on V-model	At KD X you have a reasonably fixed masterplan, it will change, but a reasonable high level plan. You know how many pilots we are going to do, we know which protos and how much time that will cost. Together with the PGP process that will be your guideline. Every PI we use those information sources to check what features we have to define. We do this based on our handbook pages, so it's pretty good described what for which moments is required as delivery.
		So for value stream 1 we have the PGP process. The PGP process you have KD moments. So for those KD moments you have deliverables, so packages of work. And so they are in time. You plan a machine, we have an idea, then we have to have all the requirements. Then we translate those requirements into work packages for development. At system level and at component level. Then we really start designing those components. Then we will build those components once, then we will test them. So you have packages of work for all those key decisions. This is done via the KD calendar. So you have a whole plan. From an idea to a shippable system. And then you start planning away in time and we do that with the KD calendar. It contains a V-model, but in fact it is just your planned work packages with the key decisions over time. So then you have the pgp process.
	Planning of portfolio & program layer based on traditional approach	In the portfolio layer you just have a waterfall planning, you just have a timeline in excel where you just plan actions and when what needs to take place. KDs are put in. The starting point is clear, the end point is already clear. Here we have to ship the first machine, then those KD moments are planned in between. And then they're going to look at those activities and see if they can squeeze them into that whole plan with a lot of risk.
		I think it is because we also have the waterfall method that the program layer uses.
		Where the working is coming from, they are working waterfall. The portfolio layer is demanding you to work like that, but your team is working in a different way and you, as program layer, are in the middle.
Task heterogeneity in team configuration	Work characteristics depending on V-model phase	What you notice is that the left side of the V, the designing phase, is reasonably plannable. [. . .] It can shift a bit now and then but it's reasonably planned. [. . .] The moment that you start to build the systems, that is also reasonably planned but you can not predict exactly what will go wrong, so you have to stand by. You can compare it with a fire brigade, they stand by and the moment someone presses the button, they have to extinguish it.
		If you look at the projects, then we have our KD overview where you have a little bit of work in the beginning to input some specs and stuff from manufacturing. Then the project starts designing from KD X. We sometimes take a look at that, but that is mainly for the D&E department to do that piece of work. And by the time it gets here in the factory or here on the proto, then we start peaking and then we peak pretty high in the pilot phase.
	Different entities with varying planning requirements within team	Yeah and TS works very ad hoc. [...] All of their work is pretty much dealing with issues or picking up stuff. You always see on their board that they have like 5 or 6 tickets that they pretty much that they bring to every single sprint. So for them, it doesn't necessarily work.
		On scrum, it is difficult and can be frustrating because TS has only ad-hoc work, primarily. 90% of their job is dependent on whether they have to solve technical disturbances. The other 10% comes from the information of the product and saying: I want to review this document, I have to create this presentation, or whatever
	Value streams with different task characteristics	So for value stream 1 we have the PGP process in which you have KD moments. For these KD moments you have deliverables, so work packages in time. [. . .] There is a V-model in that, but in fact, you plan the work packages with key decisions over time. [. . .] For value stream 2 we have Hoshin Kanri. [. . .] So that is not related to the PGP process and the KD calendar but that actually hangs over it as a kind of annual process. Then you have value stream 3. [. . .] So they have a number of different system types and plan them out over time. And that is also just a year-planning with systems
		For example project X from value stream 2, almost all tasks can be executed within the team. So you can manage your dependencies. I think that is a project where you could do well with SAFe
Focus of engineers	Switching between different projects	So during planning, we look at the whole backlog of 10 different projects like: 'okay where do we have to work on'. [. . .] The thing is that on a single day, you sometimes switch between 5 or 6 different projects and then you miss that focus. That is not ideal, but it is the reality.
		And if you look at a higher level, it's kind of like ping pong sometimes. Now work on this, now work on that, I have to work on this. So the focus is not there.
	Having too many variations in tasks	And that is just one distraction that you were working on project A and you get distracted by project B and then how much time it takes to go back to project A. Whereas we are working with 20 or so products and then on top of that we also have emails and chats and so on disturbing. And our own phones disturbing. I think there we should do more for the focus of our engineers.
		Ideally, you would say: we are working on one product. Well, we've just discussed it, we're working on 7 systems, not including value stream 2 and value stream 3. That makes it more difficult to keep focus.