ISBN 978-82-326-5784-1 (printed ver.)
ISBN 978-82-326-5402-4 (electronic ver.)
ISSN 1503-8181 (printed ver.)
ISSN 2703-8084 (electronic ver.)

NTNU
Norwegian University of
Science and Technology

Doctoral thesis

Kathrine Vestues

Doctoral theses at NTNU, 2021:172

NTNU
Norwegian University of
Science and Technology
Thesis for the degree of
Philosophiae Doctor
Faculty of Information Technology
and Electrical Engineering
Department of Computer Science

NTNU

Doctoral theses at NTNU, 2021:172

Kathrine Vestues

Using digital platforms to promote value co-creation: A case study of a public sector organization

NTNU
Norwegian University of
Science and Technology

Kathrine Vestues

Using digital platforms to promote value co-creation: A case study of a public sector organization

Thesis for the degree of Philosophiae Doctor

Trondheim, June 2021

Norwegian University of Science and Technology
Faculty of Information Technology
and Electrical Engineering
Department of Computer Science

NTNU
Norwegian University of
Science and Technology

NTNU
Norwegian University of Science and Technology

Thesis for the degree of Philosophiae Doctor

Faculty of Information Technology
and Electrical Engineering
Department of Computer Science

© Kathrine Vestues

ISBN 978-82-326-5784-1 (printed ver.)
ISBN 978-82-326-5402-4 (electronic ver.)
ISSN 1503-8181 (printed ver.)
ISSN 2703-8084 (electronic ver.)

Doctoral theses at NTNU, 2021:172

PRINTED IN
NORWAY
NO - 1598

NORDIC SWAN ECOLABEL
Printed matter
2041 0731

Printed by Skipnes Kommunikasjon AS

Abstract

This thesis explores how digital platforms can improve value co-creation in large public sector organizations characterized by large, interconnected information systems. This research is based on a two-year longitudinal case study of the Norwegian Labour and Welfare Administration (NAV). This thesis is especially focused on the way in which digital platforms enable collaboration and co-creation across time and space.

This thesis has three goals. First, it investigates the role of agile development in the context of value co-creation in public sector organizations. Second, it explores how the inertia of existing systems and practices affect an organization's ability to co-create value. Third, it analyses the way in which digital platforms can be used to overcome the challenges of existing systems and practices, enabling co-creation at scope and scale.

Theoretically, this thesis aims to contribute primarily to the field of information systems and secondarily to the field of software engineering. By developing a framework based on service-dominant logic, these two fields are connected, and insight is provided into the relationships between agile development practices, digital platforms, and the organizational context in which development practices and platforms exist.

This thesis contributes in at least three distinct ways. First, this thesis contributes to the software engineering literature by conceptualizing agile development as the activities and processes that underlie resource integration. Second, this thesis contributes theoretically to the information systems literature by proposing platformization as a strategy for reintroducing and maintaining flexibility in existing infrastructures. Third, this thesis contributes empirically to the information systems literature by proposing digital platforms as a means for scaling value co-creation across time and space in public sector organizations.

Preface

This thesis is submitted to the Norwegian University of Science and Technology (NTNU) in partial fulfilment of the requirements for the degree of Philosophiae Doctor (PhD). The PhD work was performed at the Department of Computer and Information Science, NTNU, Trondheim, under the supervision of Professor Eric Monteiro (main supervisor), Professor Torgeir Dingsøyr (co-supervisor), Associate Professor Elena Parmiggiani (co-supervisor), and Associate Professor Knut Rolland with the University of Oslo (co-supervisor). This work was supported by the Agile 2.0 project, which is sponsored by the Research Council of Norway through grant 236759 and by several companies, namely, DNV GL, Equinor, Kantega, Kongsberg Defence & Aerospace, Sopra Steria and Sticos.

Acknowledgements

I would like to thank Eric Monteiro (main supervisor) for many inspiring discussions. Eric changed the way I view the world. I also want to thank Torgeir Dingsøyr (co-supervisor) for providing me with funding and resources, for including me in the SINTEF research group, and for commenting on my work. I thank Elena Parmiggiani (co-supervisor) for her comments and guidance and Knut H. Rolland (co-supervisor) for his co-writing and support. Knut gave direction and momentum to my work.

In addition, I want to thank Marius Mikalsen and Bjørn Olav Bjørnson with the SINTEF research group for writing and discussing with me and Thomas Østerlie at NTNU for his co-writing and comments.

I would also like to thank NAV for granting me access to the examined case and informants within NAV for taking the time to talk to me. I especially want to thank Petter Hafskjold for welcoming me into the IT Architecture division and Ida Kristine Dørum for guiding me through the organization. I also thank Unni Garnes and the regional NAV administration in Trondheim for giving me access to their premises and including me in their work environment.

In addition, I want to thank my colleagues in Konik AS for their economic and moral support, and for keeping me connected to the real world.

Last but not least, I want to thank my husband Tore for his unconditional support and my children Anna and Emil for their patience and understanding.

Table of Contents

1 Introduction

1.1 Background and motivation

During recent years, there has been growing recognition among scholars and practitioners regarding the need for a more service-oriented, reliable and innovative government at all levels, which is essential to developing a dynamic, productive European society. Motivated by supranational agreements such as the "Tallinn Declaration on eGovernment" (Comission 2017), governments are changing the way they develop and disseminate services. At the centre of this transformation is a shift from designing and delivering public services solely based on an internal policy-driven logic of public administration to adopting an open and collaborative approach where services are co-created in collaboration with citizens (Mergel et al. 2018). The ambition behind this shift has been the goal to develop more efficient and effective services and increase government transparency and interoperability as well as citizen satisfaction.

Within the software development context, this need has been addressed through the use of agile development methods. These methods are based on iterative approaches, where users' opinions are continuously collected and reintegrated into subsequent versions of software. The concept of a user-centred public administration is not new. In contrast, it has been part of the digital strategies of the OECD member countries over the past two decades (OECD 2017).

However, involving users in the development of public services has proven difficult in practice. Most public sector organizations are characterized by large, integrated systems and rigid structures. These structures hamper the flexibility needed to achieve efficient value co-creation. As a consequence, service delivery in public sector organizations tends to follow a staged delivery model where user input and feedback are employed to measure user satisfaction but not to inform or drive the design of public services. This seems to be leading to "a government-centric culture and approach where citizens' needs are inferred and, as a result, not widely met" (OECD 2017, p. 36). To address this challenge, public sector organizations must find alternative delivery strategies where citizens' input is continuously collected and reintegrated.

In this thesis, I explore the way in which digital platforms can be used as a means for achieving increased user involvement and improved value co-creation in public sector organizations. Digital platforms have become a popular approach to closing the gap between service providers and customers and are viewed as "the answer" to transforming public service delivery (Brown et al. 2017, p. 168).

To gain insight into the way in which digital platforms can enable value co-creation in public sector organizations, I draw on a longitudinal study done by the Norwegian Labour and Welfare Administration (NAV). NAV is a crucial public service provider in Norwegian society and is principally responsible for a wide variety of unemployment benefits, pensions, and sickness benefits. NAV is a large organization, with almost 19 000 employees, 2.8 million active users, and close to 300 ICT systems. The societal significance of NAV together with its considerable size makes this organization an interesting case setting for analysing the general trends related to shifts in eGovernance.

During the past few years, NAV has made radical changes to the way they develop and disseminate services. They have transitioned from utilizing a staged delivery model with limited user involvement to employing a network-oriented approach where multidisciplinary teams engage in continuous value co-creation with citizens. During the course of only a few years, monolithic systems and the outsourcing of software development activities have been replaced by relatively agile approaches emphasizing user involvement and the co-creation of services.

1.2 Theoretical approach

To investigate the role of platforms in enabling improved user involvement in large public sector organizations, I draw on three strands of theorizing. First, I use *service-dominant logic* (Vargo and Lusch 2004) to theorize the relationships between digital platforms, software development practices, and governance structures. Service-dominant logic provides an alternative view on value creation, where value is thought to be *co-created* in the interaction between service providers and users. While traditional goods-oriented logic maintains that value is created by a producer and delivered to customers who assume the role of passive consumers, service-dominant logic holds that public sector organizations cannot create value for citizens. They can only make value propositions that citizens might choose to accept.

Furthermore, service-dominant logic emphasizes that value propositions and their potential to create value for citizens depend on the social context. As a context changes, for instance, as citizens acquire new knowledge or appropriate new technologies, individuals' preferences and needs will change. To reflect this time- and context-dependent understanding of value, public sector organizations must therefore adopt more agile approaches (Dybå and Dingsøyr 2008; Mergel et al. 2020). By continuously collecting and reintegrating feedback into the service delivery process, public services will be perceived as valuable *over time* (Vestues et al. 2021).

However, value co-creation has proven to be difficult in large-scale and distributed settings (Dingsøyr et al. 2019; Roland 2018). To explore a principal source of these difficulties, I draw on *path theory* (Garud et al. 2010; Singh et al. 2015; Sydow et al. 2009), which is the second strand of theorizing. Path theory, or path constitution theory, attempts to integrate the somewhat competing theories of path dependence and path constitution. In short, path constitution holds that, over time, the inertia of existing systems might reduce the flexibility of an organization but that mindful actors are able to intervene and increase the organization's technical and organizational flexibility (Rolland and Vestues 2020).

The third strand of theorizing relates to *digital platforms* and their ability to enhance the efficiency and effectiveness of value co-creation. Such platforms provide a venue where development teams and end-users can meet and exchange services, forming an ecosystem of loosely coupled actors that collaborate in the co-creation of value (Lusch and Nambisan 2015). By facilitating the collection and reintegration of feedback (Krancher et al. 2018), digital platforms have the potential to enable value co-creation across time and space.

Together, these perspectives are used to derive a framework of *platformization,* where platformization denotes the sociotechnical process of establishing a platform structure across existing systems and practices. Through the decoupling of legacy systems into platform-oriented infrastructure and the recoupling of an organization into an ecosystem of loosely coupled development teams, an organization is able to increase its organizational and technical flexibility and improve the efficiency and efficacy of its value co-creation (Vestues and Rolland 2021). A platform thus provides a fluid structure where teams, domains, and services can be combined and recombined in response to the evolving needs of users, enabling organization-wide value co-creation across time and space (Vestues et al. 2021).

The theoretical framework of this study is described in more detail in chapter 2.

1.3 Research setting and approach

The fieldwork on which this thesis is based was conducted within the IT department of NAV. The NAV IT department has approximately 700 employees and 400 consultants who develop, maintain, and operate nearly 300 information systems. The organization's application portfolio is made up of several generations of solutions, from mainframe systems to newer web-oriented applications, as well as standard systems that support operations such as accounting, payroll, and document production.

In 2017, the organization initiated a shift in the way they developed and delivered digital services, replacing staged deliveries and coordinated releases with a distributed governance model where multidisciplinary teams were responsible for developing and delivering software in close collaboration with citizens and other stakeholders. However, as the organization had a history of failed IT projects and a reputation of engaging in inefficient and expensive software development, this shift was met with scepticism both externally and among internal employees. Critics claimed that the complexity of the organization, combined with the interconnectedness of its applications, made the transition both unrealistic and irresponsible. It was therefore with amazement that I observed the speed and efficiency with which the transition took place. During the two years of fieldwork, NAV was able to restructure its IT department, increase its responsiveness, and reduce the number of critical system errors. Staged deliveries were gradually replaced by more collaborative practices where users and other stakeholders were engaged in the value co-creation process.

Agile approaches and processes of value co-creation are often antithetical to bureaucratic line organizations (Mergel et al. 2020). Therefore, NAV provides a unique opportunity for studying value co-creation in a large public organization. Although digital platforms in the private sector have transformed the global economy (Parker et al. 2016), the adoption of digital platforms in the public sector has been much slower. Both researchers and practitioners are therefore in need of increased insight into how public sector organizations can adopt co-creation practices across existing infrastructures.

To investigate the way in which digital platforms enable improved value co-creation in public sector organizations, I draw on a longitudinal case study that employs an IT department as its unit of analysis. This research approach was inspired by Pan and Tan (2011)'s structured pragmatic situational (SPS) approach to conducting case studies. The research process of the SPS approach is described in terms of a "framing" cycle and an "augmenting" cycle where a researcher iterates between data collection and data analysis until theoretical saturation is reached. Central to this study was obtaining an understanding of the beliefs and views of stakeholders in change processes and the context in which these changes take place. Empirical studies that gather this type of data are often described as "interpretive" (Walsham 1995). Interpretive studies are based on the assumption that people have subjective interpretations of the world. A researcher must thus reconstruct a given phenomenon by accessing these interpretations. "What we call our data are really our own constructions of other people's constructions of what they and their compatriots are up to" (Geertz 1973).

As illustrated in Figure 1, the research questions of this study were motivated by input derived from empirical data and a literature review. The longitudinal case study enabled me to observe the examined phenomenon over time and adopt an evolutionary perspective. Data were collected though semi-structured interview, participant observations, and document reviews and were analysed using different sensemaking strategies for processing data (Langley 1999). Although I illustrated these tasks as a linear process, the theory development, data collection, and data analysis of this paper were performed interactively, enabling me to respond to emergent themes as my knowledge of the case and the literature matured.

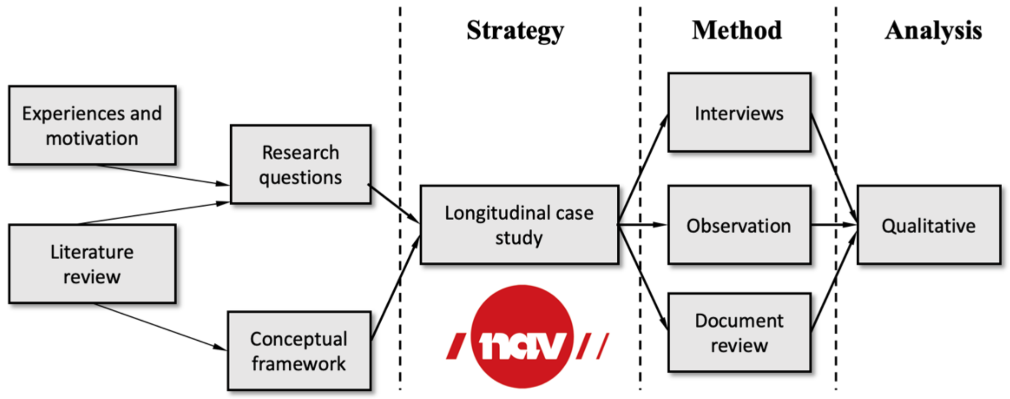


Figure 1. Research approach

The research setting and approach are elaborated upon in chapters 3 and 4.

1.4 Research Goal

The goal of this research is to provide insight into the way digital platforms and platformization processes might improve collaboration and value co-creation in public sector organizations. Digital platforms have transformed the global economy and are changing the way people and organizations interact. However, little is known about the transformative potential of these platforms in the public sector. Gaining improved insight into the use of digital platforms in public sector organizations and the way in which platforms can be used to increase value co-creation might improve both the efficacy and the efficiency of public services.

The research goal (RG) is formulated as follows: *How does platformization contribute to value co-creation in public sector organizations?*

The main research question is refined into the following three research questions:

RQ1 What is the role of agile development in value co-creation?

RQ2 How does inertia affect an organization's ability to co-create value?

RQ3 How do digital platforms enable value co-creation across time and space?

1.5 Contributions

This thesis contributes primarily to the information systems literature and secondarily to the software engineering literature in at least three ways:

C1 First, this thesis contributes to the software engineering literature by conceptualizing agile development as an expression of a *service-dominant logic*, where value creation happens through the alignment of governance strategies, development practices, and digital platforms.

C2 Second, this thesis contributes theoretically to the information systems literature by proposing platformization as a strategy for reintroducing and maintaining flexibility in existing infrastructure.

C3 Third, this thesis contributes empirically to the information systems literature by proposing that digital platforms are a means for scaling value co-creation across time and space in public sector organizations.

These three research questions, which are posed in section 1.4, are answered in the following way:

1) Agile development ensures that feedback from users and other stakeholders is collected and reintegrated into the service delivery process through iterative and collaborative practices, thereby ensuring value co-creation over time.

2) The inertia of existing systems and practices reduces an organization's ability to co-create value. The organization must therefore find strategies for reintroducing flexibility if efficient value co-creation is to occur.

3) Digital platforms provide a means for indirect and mediate value co-creation, where feedback from large and homogeneous user groups can be rapidly and continuously reintegrated into software, enabling improved value co-creation across time and space.

1.6 Included papers

Five research papers are included in this thesis. The implications of their contributions are discussed in section 6. The relationships between various contributions and these research papers are outlined in Table 1 below.

Paper 1. Vestues, Kathrine; Bjørnson, Finn Olav. (2016). *Agile requirements work in a digital transformation project: Managing diverse and dispersed user needs*. Paper presented at the International Research Workshop on IT Project Management (IRWITPM).

Paper 2. Dingsøyr, Torgeir; Mikalsen, Marius; Solem, Anniken; Vestues, Katherine. (2018) *Learning in the Large: An Exploratory Study of Retrospectives in Large-Scale Agile Development*. Agile Processes in

Software Engineering and Extreme Programming, 19th International Conference, XP 2018, Proceedings.

Paper 3. Rolland, Knut; Vestues, Kathrine. (2020) *Inertia and change in transformation of the IT-function in large organizations: A path theory lens*. Accepted to NOKOBIT 2020.

Paper 4. Vestues, Kathrine; Rolland, Knut, *Platformizing the Organization through Decoupling and Recoupling: A longitudinal Case Study of a Government Agency*, (2021) Initially submitted to Scandinavian Conference of Information Systems (2019) and later fast-tracked and accepted to Scandinavian Journal of Information Systems (2021).

Paper 5. Vestues, Kathrine; Mikalsen, Marius, Eric Monteiro (2021), *Using digital platforms to promote a service-oriented logic in public sector organizations: A case study*, Accepted to the 54th Hawaii International Conference on System Sciences. Nominated for the "Best paper award" in the "Digital Government" track.

Table 1. A detailed overview of various contributions and their relationships with the examined research questions and papers.

No	RQ	Key finding	Paper
C1.1	1	Agile development contributes to the co-creation of value by providing the roles, practices and processes that underlie resource integration.	1, 2, 5, Thesis
C1.2	1	Agile development provides iterative and collaborative work practices that enable value co-creation *over time.*	1, 2, 5, Thesis
C1.3	1	This thesis contributes to closing the gap between the software engineering and information systems fields by applying information systems theories in the context of the exploration of software development practices.	1, 2, 5, Thesis
C2.1	2	Public sector organizations can develop organizational inertia in terms of path dependence related to software development methods, sourcing strategies, governance strategies, and technology platforms.	2, 3, Thesis

C2.2	2	By introducing path-breaking mechanisms, organizations can break away from an established path.	3, Thesis
C3.1	3	Digital platforms enable co-creation at scope and scale by facilitating the continuous collection and reintegration of feedback from large, heterogeneous user groups.	4, 5, Thesis
C3.2	3	Platformization facilitates a cultivation strategy where silo-based organizations are replaced by platform structures.	4, Thesis
C3.3	3	The decoupling of legacy systems into platform-oriented structures enables improved innovation and value co-creation *inside* organizations.	4, Thesis
C3.4	3	The recoupling of an organization enables a dynamic reconfiguration of its value paths in response to emergent needs.	4, Thesis
C3.5	3	Decoupling and recoupling become mutually enabling, which ensures the continued and ongoing improvement of an organization's capacity to co-create value.	4, Thesis

2 Theoretical framework

The framework used in this thesis is based on three themes. First, *service-dominant logic* (Vargo and Lusch 2004) is used to conceptualize agile development as a collaborative activity where value is co-created through the interaction between software developers and citizens. Central to this logic is the idea that value is individualistic and context dependent and that only citizens can decide what constitutes value. For services to be perceived as valuable over time, public sector organizations must collect and reintegrate citizens' feedback throughout service delivery cycles (Normann and Ramirez 1993).

The second theme involves *path constitution theory* (Singh et al. 2015; Sydow et al. 2009) and the difficulties associated with value co-creation in large-scale settings. Path constitution theory opens up the analytical possibility of explaining both why public sector organizations can become locked onto paths of rigidity and inefficiency and how these paths can be broken.

The third and final theme refers to *digital platforms* and their ability to support value co-creation in large-scale settings. The modular structure of a digital platform provides a venue where users and developers can meet and co-create value across time and space.

In the following sections, each of the three themes is discussed in detail before being merged into a theoretical framework that is used for analysing the examined case.

2.1 Co-creating value

Service-dominant logic denotes a logic of value creation that holds that value is co-created through interactions between providers and users (Lusch and Nambisan 2015; Vargo and Lusch 2004). In this section, I begin by describing what service-dominant logic entails and how it differs from traditional goods-oriented perspectives. These differences are underscored for pedagogic reasons. I then introduce agile software development and describe the way in which agile development practices enable value co-creation in a software development setting.

2.1.1 Service-dominant logic

In private firms, value is measured in terms of annual income. If a firm is unable to provide shareholders with a profit, investors will most likely withdraw their financing and invest elsewhere. In addition, private firms must answer only to shareholders and the market. As long as customers continue to buy its products and services, a firm is said to generate value.

In the case of public organizations, the picture is more complex. The main source of income obtained by these organizations comes from public money for public purposes, which means that they rely on the goodwill and support of citizens and their elected representatives. To secure necessary resources, managers in public sector organizations must therefore sell a story of public value creation (Moore 1995, p. 5). Alford and Hughes (2008) argue for "public value pragmatism", where service delivery is adapted to the specific circumstances of a given situation, including the context, the nature of the task at hand, and the type of value being produced. Central to this argument is the view that value is context dependent and depends on the needs and wants of citizens; indeed, “citizens have different and often conflicting preferences about different issues, and moreover, these preferences change over time, sometimes quickly" (Alford and Hughes 2008, p. 133). It is therefore inherently difficult for managers to anticipate and define the factors that constitute value for citizens. Instead of striving towards finding a universal definition of what constitutes public value, public sector organizations must therefore embrace the perception that they hold and seek feedback on this perception from the public. “The main objective is therefore not to deliver a predefined conception of public value but to develop organizational capabilities which enable the organization to share their opinion of what constitutes public value and respond to the subsequent feedback” (Moore 1995). The problem for public sector organizations then becomes articulating a vision of public value and seeking rapid and continuous feedback on this vision; in the context of the public sector, ‘value’ is inherently transient and context dependent.

This insight contradicts views traditionally held in public sector organizations, where value is thought to be created by public officials and delivered to citizens who take the role of passive consumers. Central to this linear view on value creation, which is often referred to as a *goods-dominant logic* (Vargo and Lusch 2004), is an emphasis on increasing the internal efficiency of public administration, largely ignoring the opinions and needs of citizens. “In Norway, user input and feedback are relevant to measure user satisfaction (e.g., surveys) but not to inform or drive the design of public services. This seems to be leading, in general terms, to a government-centric culture and approach where citizens’ needs are inferred and, as a result, not widely met” (OECD 2017).

However, this goods-dominant logic has come under increasing criticism for failing to address the complex, fragmented, and emergent needs of citizens (Osborne 2018; Osborne et al. 2013; Osborne et al. 2016). As a consequence, researchers have identified an alternative logic, which holds that value is co-created through interactions between public sector organizations and citizens. Central to this *service-dominant logic* is that public sector organizations cannot create value for citizens—they can only make a “value proposition” that citizens might choose to use (“value-in-use”) (Osborne 2018; Vargo and Lusch 2004).

Furthermore, service-dominant logic emphasizes that value propositions and their potential to create value for citizens depend on the social context in which the service is offered (“value-in-context”) (Lusch et al. 2010). As the context changes, for instance, as citizens acquire new knowledge or appropriate new technology, individuals’ preferences and needs will change. If services are to be perceived as valuable *over time*, public sector organizations must therefore continuously seek feedback from citizens and improve their value propositions accordingly.

Within the software development context, agile software development has been proposed as a means for achieving value co-creation (Babb and Keith 2012; Kautz and

Bjerknes 2020). In the following section, I describe what agile development entails, how it aligns with a service-dominant logic, and how agile development practice contributes to value co-creation in public sector organizations.

2.1.2 Agile software development

Agile development methods can be seen as a reaction to traditional staged development approaches where activities such as design, development, and operations are performed by distinct parts of an organization and progress along a production line. Foundational to a staged approach is the idea that systems are fully specifiable and that optimal and predictable solutions exist for every problem. Goals of efficiency and predictability are achieved through planning and reuse. Development teams conform to predefined specifications and have limited ability to adjust these designs to meet emerging user needs. The staged development process resembles a traditional assembly line and is characteristic of a goods-dominant logic where value is defined and produced within an organization and delivered to users upon completion:

> *"As organizations continued to increase in size, they began to realize that virtually all their workers had lost sense of both the customer [...] and the purpose of their own service provision. The workers, who performed microspecialized functions deep within the organization, had internal customers, or other workers. One worker would perform a microspecialized task and then pass the work product on to another worker, who would perform an activity; this process continued throughout a service chain. Because the workers along the chain did not pay one another (reciprocally exchange with one another) and did not typically deal directly with external customers, they could ignore quality and both internal and external customers. To correct for this problem, various management techniques were developed under the rubric of total quality management [...]"* (Vargo and Lusch 2004, p. 8).

Agile methods, on the other hand, acknowledge that the world is inherently unpredictable and that this uncertainty must be addressed with incremental and iterative approaches and close customer interaction. Conboy and Fitzgerald (2004) suggest that agile development practices were inspired by similar trends in other fields, such as agile manufacturing (Sanchez and Nagi 2001; Tan 1998) and lean production (Womack and Jones 1997; Womack et al. 2007).

Agile software development constitutes a set of methods and practices for software development that were created by experienced practitioners (Dybå and Dingsøyr 2008). In this context, "agility" can be defined as "the continual readiness of an ISD method to rapidly or inherently create change, proactively or reactively embrace change, and learn from change while contributing to perceived customer value (economy, quality, and simplicity), through its collective components and relationships with its environment" (Conboy 2009, p. 341). This definition suggests that value creation happens through a process of feedback and learning that necessitates short "inspect-and-adapt" cycles with frequent feedback (Williams and Cockburn 2003).

The need for continuous development practices, where user feedback is reintegrated into service delivery processes, is further emphasized by movements such as DevOps (Krancher et al. 2018) and BizDev (Fitzgerald and Stol 2017), where practices of continuous integration are combined with collaboration practices that ensure integration across the departments of an organization. By removing the disconnects between actors who have previously been kept apart, organizations are able to rapidly respond to the emerging needs of users.

Recently, a growing number of public sector organizations have begun to adopt agile development approaches. For instance, digital service teams such as the U.S. Digital Service, the United Kingdom's Government Digital Service, and the Canadian Digital Service have paved the way for working in an "agile way". Similarly, state and local governments in the United States have adopted agile and related practices in the contexts of innovation labs and civic service design teams (e.g., Georgia Technology Authority, New York City) (Mergel et al. 2020). Barroca et al. (2019) provide insight into an agile transformation taking place in a district council in the UK.

However, although agile methods show considerable promise and have the potential to transform public service delivery, so-called "agile transformations" have proven difficult to implement in practice (Dingsøyr et al. 2019; Fuchs and Hess 2018; Paasivaara et al. 2018). Agility and responsiveness, which are required to respond to rapidly changing technologies and user needs, often contradict established structures that favour internal efficiency over external efficacy (Osborne et al. 2016). As stated by Mergel et al. (2020), agile practices are antithetical to typical bureaucratic line organizations and require consensual decision making and an acceptance of trial-and-error methods, which are poorly suited to risk-adverse environments. Agile practices also require new forms of contracting and procurement (Mergel et al. 2020).

Successfully implementing agile development in large public organizations therefore requires strategies for overcoming the rigidity and inefficiency that often come from the technical and organizational inertia in public sector organizations. In the next section, I theorize these challenges by drawing on *path constitution theory*.

2.2 Path theory

To conceptualize and explain the rigidity and inefficiency of public sector organizations and how this rigidity can be overcome, this thesis draws on *path constitution theory* (Garud et al. 2010; Singh et al. 2015; Sydow et al. 2009). Path constitution theory attempts to integrate the somewhat competing theories of *path dependency* and *path creation*. In short, path dependency implies that historical choices and events narrow actors' options in the present, while path creation suggests that some of these choices can be reinstated through the mindful intervention of organizational actors (Meyer and Schubert 2007).

In the following sections, I begin by discussing the concept of path dependence and how it can be used to explain the rigidity and inefficiency of public sector organizations. I then proceed to discuss path creation, attempting to theorize how public sector organizations can break away from existing paths to increase their flexibility and ability to co-create value.

2.2.1 Defining path dependence

The concept of path dependency was initially used in evolutionary economics as a theoretical basis for describing why actors chose less than optimal solutions when constituting the path of complex technologies (Arthur 1989; Arthur 1994; David 1985; David 1994). The concept was later extended by Sydow et al. (2009) to an organizational context, where it was used to explain how organizations can lose their flexibility and become inert or even locked in. The theory contends that path dependence can be conceptualized as a process that entails three distinct phases. In the first phase, the organizational path is relatively open. However, path dependence arises at a critical juncture. At this point, a specific pattern or organizational path starts emerging, and self-reinforcing mechanisms take effect. Because of these self-reinforcing mechanisms, the options for organizational actors to deviate from the organizational path are increasingly diminished. The resulting lock-in is precisely what threatens the required responsiveness to evolving demands from citizens in a public sector setting.

A self-reinforcing mechanism is a mechanism in which each subsequent step intensifies the effect of the previous step (Schreyögg and Sydow 2011). Sydow et al. (2009) identify four specific mechanisms that shape the trajectory of an organization. First, there are coordination effects caused by actors who adopt the same organizational routine or rule, making it increasingly attractive for others to adopt the same routine or rule. Thus, coordination becomes increasingly more efficient and less costly. The second element in the framework is complementary effects, which refer to self-reinforcing effects arising out of "the interaction of two or more separate but interrelated resources, rules, or practices" (Sydow et al. 2009, p. 699). Complementary effects are generated through combinations of multiple routines and practices, so that it becomes increasingly attractive to adopt all of them as an "institutional cluster" (David 1994). Third, there are self-reinforcing learning effects at different levels in organizations. Obviously, the more often a specific task is performed, the easier it becomes to perform it and the more efficiently it is executed. Arguably, all three of these variants of self-reinforcing mechanisms are relevant for public sector organizations. A fourth self-reinforcing mechanism is the adaptive expectation effect. Since individual preferences or choices are affected by other relevant actors' expectations, it becomes preferable to choose certain solutions that are perceived as 'right' by an increasing number of actors. This mechanism is described by Mehrizi et al. (2019) as "legitimization" in reference to information systems use. The more widely an information system is used, the more widely it will be accepted by organizational actors; this applies to the learning effects and coordination effects related to choices involving development approaches (i.e., staged or continuous), application platforms (i.e., JBoss or WebSphere), and programming languages (i.e., Java or .NET). Moreover, as illustrated in a case study by

Law (2017), sourcing models carry strong path dependencies. Once software development activities are outsourced, it becomes progressively harder to discontinue the use of this setup due to self-reinforcing mechanisms.

Path dependence does not in itself imply inefficiency. However, as an organization becomes locked onto a path, it becomes increasingly difficult for the organization to adapt to changing environments and customer expectations. Rigidity thus becomes a liability that often results in inefficiency over time (Sydow et al. 2009).

Path dependence can occur at different levels of organizations (Henfridsson et al. 2009). For example, in the context of new product development, researchers have noted the importance of cognitive and organizational structures (Boland Jr et al. 2007; Henfridsson et al. 2009), while Baldwin and Clark (2000) suggest that there is an interrelationship between physical products, conceptual design structures, and task structures in the context of organizational resources. A similar point is made by Conway (1968), who argues that system designs reflect communication structures in development organizations. If a multi-layered perspective is adopted, path dependence can be analysed within or across different layers.

In a software development setting, paths might be formed both on organizational and technical layers. On an organizational layer, path dependence can be induced by mechanisms such as coordination effects, complementary effects, learning effects, and adaptive expectations effects (Sydow et al. 2009), while a technical path will be related to a given system's evolvability. In the existing software engineering literature, the ability (or inability) to evolve is often described in terms of "technical debt". This term was initially introduced by Cunningham (1992) as a way to explain why software systems need "refactoring". The term was later used to describe the general inertia of existing software and the need for ongoing maintenance if system owners wish to avoid software rot. Ramasubbu and Kemerer (2016, p. 1487) describe technical debt as "a buildup of software maintenance obligations that need to be addressed in the future". Kruchten et al. (2012) use this concept to describe the evolvability of software architecture and the maintainability of software. This means that technical debt is not simply the result of having made a wrong choice originally but rather the result of an evolution where choices made yesterday are no longer appropriate today. Technical debt can thus be caused by technological obsolescence, changes in the environment, rapid commercial success, or the advent of new and better technologies (Kruchten et al. 2012, p. 19). The longer a system is left unattended, the more inflexible it becomes. Eventually, it becomes a "legacy system", where the term legacy system (or "installed base") simply denotes a system that has become difficult to change and maintain for some reason. Legacy systems that have accumulated throughout the years may significantly impact a corporation's freedom to improve and innovate. For instance, Mehrizi et al. (2019) suggest that of the annual cost involved with maintaining legacy systems, only 25% provides competitive value to organizations.

Public sector organizations and industries where interdependent technological objects, organizational routines, and actors have developed historical commitments are more prone to path-building forces than markets. Pierson (2000) argues that the corrective mechanisms are even less effective when one shifts from firms in private markets to the world of political institution. This happens because of the complexity of their organizational goals and the unclear link between their actions and outcomes. Therefore, practices, once they have been established in public sector organizations, will relatively easily gain momentum and create fertile ground for developing positive self-reinforcing feedback. In contrast, private firms and markets are subject to organic corrections of inefficient action.

However, the concepts of self-reinforcing mechanisms and path dependence do not explain how some organizations are able to break away from existing paths. While the original version of path dependency theory argued that paths could only be broken by external shocks or extraordinary events (Arthur 1990), Garud et al. (2010) introduce the concept of "path creation", arguing that an organizational path can be broken through the deliberate actions of organizational actors.

2.2.2 Creating new paths

From a path creation perspective, agency is seen as a largely distributed group of possibilities in which many different actors in different situations and contexts can deliberately "break" an existing path. Furthermore, path creation emphasizes that novel paths can stem from improvisation and bricolage, as well as through the active cultivation of serendipity, for example, the case of innovating Post-it Notes (Garud et al. 2010). Law (2017) shows how path dependencies can be broken by strategically mobilizing resources for creating new paths. Rolland et al. (2015) use the terms "architectural path dependencies" and "architectural hacking" to theorize that the evolution of enterprise architectures is influenced both by path dependencies and path creation.

Thus, the main difference between the concepts of *path dependence* and *path creation* concerns the role of agency and the idea that actors can deliberately reverse or reduce the casual mechanisms of path dependence. According to certain literature on organizational path theory, organizations can break away from an established path by "interrupting the logic and the specific energy of the self-reinforcing mechanisms" (Sydow et al. 2009, p. 702). Whether a path can be broken depends on the reversibility of the related process and the possibility of presenting the given organization with an attractive and viable alternative. "Major features here are resource commitment, reversibility, and transferability of experience" (Arthur 1989). A superior and viable alternative must be presented.

Henfridsson et al. (2009) suggest that path creation occurs across multiple layers. In the field of software design, innovation involves changes to both material properties and cognitive models. If software organizations wish to increase their flexibility and establish alternative paths, they must make changes to both their technology and their organization. On the technical level, path dependence must be addressed by breaking dependencies (Feathers 2005) and installing modular structures (Hanseth and Lyytinen

2010). Legacy systems are particularly important since they constrain rather than support the ability of an organization to respond to changing environmental conditions or to adopt new strategies.

Similarly, on an organizational level, flexibility and innovation are best achieved by moving towards decentralized structures. Organizations are, by definition, characterized by a central authority and some degree of control. However, Ciborra (2000) suggests that traditional management approaches based on top-down control are unsuitable in large organizational settings, advocating for a more humble, iterative and incremental approach. Later, such approaches were described as "cultivation", contrasting this approach to the mere "construction" mode, which has traditionally characterized large organizations. In a cultivation approach, monitoring and gradual adjustments are preferred over strict control and rigid preplanning (Aanestad et al. 2017).

Recently, digital platforms and processes of platformization have become attractive alternatives for organizations that wish to break away from drifting infrastructures, increase their flexibility and value co-creation (Törmer and Henningsson 2018), and make their digital infrastructure more flexible (Battleson et al. 2016). Digital platforms, then, have the potential for both the establishment of path dependencies (through network externalities) and path creation (through complementaries) (Parker et al. 2016; Tiwana et al. 2010). The potential of platforms to increase an organization's flexibility is discussed in the following sections.

2.3 Platformizing an organization

In this section, I describe how public sector organizations can increase the efficiency of their value co-creation through a process of "platformization". I begin by describing how digital platforms contribute to the scaling of value co-creation across time and space before elaborating on the role of platformization in the co-creation process.

2.3.1 Defining digital platforms

Digital platforms enable innovation (Yoo et al. 2010) and value co-creation (Cennamo and Santaló 2019) and have been studied as a means for increasing value creation (Ju et al. 2019). From an economic perspective, platforms create value by acting as mediators between two or more categories of users who would otherwise not connect (Eisenmann et al. 2006; Parker et al. 2016), while from an engineering perspective, they are seen as technology foundations that enhance generativity and innovation through their layered modular structures (Tiwana et al. 2010; Yoo et al. 2010). These technical innovations have opened up new possibilities for firms to gain user input throughout the whole innovation process (Bosch 2015; Bosch-Sijtsema and Bosch 2015).

To understand the potential of platforms, it can be helpful to think about the way a business or organization creates and transfers value. As discussed in section 2.1, value creation has traditionally followed a step-by-step process with producers at one end and consumers at the other. In this pipeline arrangement, a product is first designed and then manufactured and offered for sale. Because of its single track shape, this manufacturing process is often described as a linear value chain (Parker et al. 2016).

Recently, an increasing number of businesses have adopted a platform model where they interact with users in a complex arrangement of service consumption and production. “Rather than flowing in a direct line from producers to consumers, value may be created, changed, exchanged, and consumed in a variety of ways and places, all made possible by the connections that the platform facilitates” (Parker et al. 2016). Although this shift might sound modest, its effect is astounding: The introduction of platform models is revolutionizing one industry after another (ibid).

Additionally, public sector platforms are receiving attention. They let governments tap into existing communication channels (Bonsón et al. 2015; Zavattaro et al. 2015) and engage citizens in the arenas they know. For instance, Hand and Ching (Hand and Ching 2020) examine how social media platforms such as Facebook and Twitter let citizens engage with police agencies, while Nam (Nam 2020) explores the way in which digital platforms enable discussions about rule making between citizens and other stakeholders. Similarly, other studies explore the challenges and opportunities related to open government data, focusing on issues such as innovation (Danneels et al. 2017), civic engagement (Kassen 2013), and the design of open data platforms (Ruijer et al. 2017). Public sector platforms can potentially increase both the transparency and efficiency of public sector organizations by exposing public sector data and engaging citizens in co-creation (O'Reilly 2011).

While the debate on digital platforms in public sector organizations has proven useful, much of the existing literature in this field has focused on digital platforms as a means for facilitating communication between public sector organizations and citizens. As an exception to this trend, Dunleavy et al. (2006) argue that we have entered an era of digital governance where public sector developments revolve around changes in digital technologies and alterations in information systems. By reintegrating public services, digital technologies are enabling a “needs-based holism” where end-to-end processes and agile practices are increasing public sector organizations’ ability to respond to emerging citizen needs (Dunleavy et al. 2006). Similarly, Fishenden and Thompson (2012) propose that digital platforms and open architectures enable a reaggregation of digital services, promoting a service-dominant logic where citizens become an integral part of value creation processes. Central to this transformative potential is platforms’ ability to mediate between different user groups and offer resources that can be recombined into new and improved services. Hence, platforms become venues where citizens and public sector organizations can interact and exchange services and information.

Digital platforms are thus important in public sector service delivery for at least two reasons. First, digital platforms facilitate the exchange of services and information between citizens and public sector organizations. Second, platforms enable a rapid and ongoing reintegration of this information into new and improved value offerings (Lusch and Nambisan 2015). Together, these elements enable innovation and value co-creation in ways and at a scale previously impossible.

However, for many public sector organizations, a legacy of ageing systems and rigid practices are making the adoption of platform models challenging. To tackle these challenges and capitalize upon the benefits of platform models, I propose a strategy of platformization. This strategy is described more fully in the following section.

2.3.2 Defining platformization

Most public sector organizations must deal with a legacy of existing systems and practices. In this thesis, I describe the process of introducing a platform structure to an established organization as the process of "platformization". Helmond (2015) uses this term to refer to the rise of platforms as the dominant infrastructural and economic model of the social web, defining platformization as the "extension of social media platforms into the rest of the web and their drive to make external web data 'platform ready'" (Helmond 2015, p. 1). A platform thus provides a computational infrastructure that others can build on.

Poell et al. (2019) understand platformization as "the penetration of the infrastructures, economic processes, and governmental frameworks of platforms in different economic sectors and spheres of life", whereas Bygstad and Hanseth (2018) describe platformization as the process of dismantling legacy systems into platform-oriented infrastructure. In this thesis, I build on these conceptualizations, viewing platformization as the dismantling of systems into a platform-oriented structures. However, I define platformization not only as the process of picking apart but also as the process of putting back together. This implies that platformization consists of two basic processes. The first process involves decoupling systems, information, and activities into modular components, and the second process entails recoupling components that would otherwise be difficult or expensive to reconnect (Normann 2001).

This conceptualization of platformization as consisting of two separate but related processes is consistent with service-dominant logic and its concepts of liquefaction and resource integration (Lusch and Nambisan 2015; Normann 2001). However, while service-dominant logic describes liquefaction as a decoupling of information from its physical media, I take a broader view and see liquefaction both as a decoupling of information and as a decoupling of legacy systems. The reason for this is related to the accessibility of the information stored in legacy systems: Although the information stored in a legacy system is accessible in principle (Kallinikos et al. 2013), poorly designed interfaces combined with tightly coupled systems often make information difficult or impossible to access and modify. By decoupling systems into smaller components with clearly defined interfaces, information can be made more easily accessible and available for recombination.

The decoupling of information from its physical media and the decoupling of systems into smaller components not only leverages resource density but also provides a fluid structure where applications and the work processes that they support can be redistributed to the most appropriate parts of an organization (Normann 2001). This allows for a dynamic reconfiguration of work practices and organizational arrangements where value paths and organizational arrangements are continuously shaped and reshaped through the decoupling and recoupling of technology and organizations. Eric Evans (2004) describes how the challenge of modularity can be addressed by an enterprise:

> *The goal of the most ambitious enterprise system is a tightly interconnected system spanning the entire business. Yet the entire business model for almost any such organization is too large and complex to manage or even understand as a single unit. The system must be broken into smaller parts in both concept and implementation. The challenge is to accomplish this modularity without losing the benefits of integration, allowing different parts of the system to interoperate to support the coordination of various business operations.*

2.4 Framework for analysing value co-creation

The purpose of this section is to summarize and synthesize the theoretical perspectives presented above into a theoretical framework that can be used to analyse the examined case.

The first perspective relates to service-dominant logic and the notion of value co-creation (Lusch and Nambisan 2015). Service-dominant logic is used to theorize agile development as the activities and processes that underlie resource integration. Through iterative and collaborative practices, agile development ensures user involvement and value co-creation throughout a service delivery cycle. Although other works have explored value creation in software development settings, these studies have several limitations. First, the extant literature explores value creation from a business perspective (Boehm 2003; Ramesh et al. 2010), largely ignoring the role of users in the context of defining and creating value. Second, the extant literature on agile development focuses on value creation at the team level (Dingsøyr et al. 2018; Sharp and Robinson 2010; Stray et al. 2016), providing limited insight into the interaction between organizational structure, strategic decisions, and software development practices. Third, the agile literature does not consider the role of technology in enabling or inhibiting value creation. Fourth, in the extant studies on co-creation in public sector organizations, the examination of value co-creation is mostly limited to the early stages of specification and design (Voorberg et al. 2015), and these studies fail to view value co-creation as a continuous and ongoing process.

The second perspective presented in this thesis relates to the inertia of existing systems and practices. To draw on the benefits of value co-creation, large organizations must find strategies for overcoming rigidity and establish efficient feedback loops. By drawing on the concept of *path dependence* (Mahoney 2000; Sydow et al. 2009), this thesis explores the way in which legacy systems and established work practices reduce an organization's flexibility and value co-creation. In so doing, this thesis explores an empirical blind spot in the information systems literature by addressing the role of inertia in the co-creation process.

The third theoretical perspective concerns digital platforms and platformization as a strategy for overcoming technical and organizational inertia that can enable value co-creation across time and space. The extant literature discusses the role of digital platforms in facilitating communication and collaboration between governments and citizens (Hand and Ching 2020; Nam 2012) but provides limited insight into the transformative potential of digital platforms and the way in which platforms enable value co-creation at scale. The existing studies in this field also provide limited insight into the interplay between platforms and organizations and the way in which digital platforms and processes of platformization might enhance the flexibility and innovation of public sector organizations.

To explore these gaps in the extant literature and provide improved insight into the role of digital platforms in value co-creation, I performed a longitudinal case study within the IT department of the Norwegian Labour and Welfare administration. This case is described in detail in the following chapter.

3 Case

The Norwegian Labour and Welfare administration (NAV) was established in 2006 following the reform of the Norwegian welfare system. The reform was a response to long-held concerns regarding the disintegration of public welfare services and involved a merger of the formerly separate Employment Services and National Insurance Services. In addition, the reform involved a formal collaboration between NAV and municipal social services. In total, the organization employs approximately 19,000 people, of whom 14,000 are central government employees and 5,000 are municipality employees.

NAV forms the backbone of the Norwegian welfare state and is responsible for both increasing the work ability of the population and supporting citizens economically when they are unable to support themselves. To this end, NAV redistributes one-third of the national budget through schemes that include sick benefits, unemployment benefits, and old-age pensions. Providing reliable and efficient services is thus imperative to ensure both the individual wellbeing of citizens and collective trust in the welfare state.

The NAV IT department develops, maintains, and operates the information systems that support the organization. The IT department has approximately 700 employees and 400 consultants and maintains and operates close to 300 applications. The department's application portfolio is made up of several generations of solutions, from mainframe systems to relatively modern web-oriented applications, as well as standard systems that support operations such as accounting, payroll, and document production.

The service delivery strategy of the organization has evolved over the years. This evolution can be described in terms of two different value logics. Between 2006 and 2016, NAV was characterized by a goods-dominant logic. Between 2017 and 2019, NAV transitioned to a service-dominant logic. Table 2 summarizes how these periods differed in terms of sourcing strategy, governance strategy, and technical infrastructure, and each of these elements are described in more detail in the following sections.

Table 2. A comparison of the two different value logics of NAV

	Goods-dominant logic (2006 - 2016)	Service-dominant logic (2017 - 2019)
Sourcing strategy	Externally funded projects and outsourced software development methods. Large and unpredictable project financing makes the organization dependent on consultants.	Software development is insourced, and software development and maintenance are increasingly funded through the operating budget.
Technical platforms	Large and interconnected systems. Technical infrastructure and the	Platform structure with loosely coupled applications. Clearly

	application layer are tightly coupled.	defined interfaces between applications and between applications and the platform core.
Governance strategy	Staged software development methods with centralized decision models. Value is created in a “step-by-step” process where developers remain oblivious of user’s actual needs.	Iterative software development where software is developed by multidisciplinary teams. User feedback is collected and reintegrated throughout the service delivery cycle.

3.1 Pursuing a goods-dominant logic

Between 2006 and 2017, NAV was characterized by a goods-dominant logic. Central to this logic was the outsourcing of software development, large and interconnected information systems, and staged development practices with limited user involvement. The sourcing strategy, governance strategy, and technical infrastructure used during this period are described in more detail below.

3.1.1 Adopting an outsourcing strategy

When NAV was established in 2006, the organization adopted an outsourcing strategy where software development and maintenance were outsourced to consultant companies. In line with public sector procurement regulations, maintenance contracts were put out to tender every 4 – 8 years. This meant that the organization’s suppliers were replaced at regular intervals, resulting in discontinuity and loss of competence.

The use of target-price contracts required elaborate up-front specifications and formal handovers between NAV and its suppliers. To facilitate the management of the relationship between customers and suppliers, projects were designed to follow a staged development model. This staged development model implied that specification, development, and operations were performed by different teams and departments and that each stage had to be completed before the project could progress to the next. This meant that it could be years before a system that was planned and specified was available to end-users.

A significant part of the IT modernization within NAV was performed through large-scale projects funded through the national budget. Getting such funding required substantial effort. To reduce the associated administrative overhead costs, project proposals would include a large array of needs. Projects therefore became unnecessarily large and diverse, which increased both their complexity and their risk. The funding model also resulted in bursts of activity, where periods of intense activity were followed by periods of relative calm; this model made consultants an inevitable solution.

A prominent example of this funding model is the initiation of the Retirement pension program in 2006. The aim of this program was to develop a new pension system in support of legislative changes following a reform of retirement pensions in Norway. The pension system, known as PESYS, was NAVs' first online self-service solution, and it allowed citizens to plan and manage their own retirement pensions. The programme, which ran from 2006 until 2011, had an estimated cost of more than 2 billion NOK and was outsourced to consultant companies. NAV was responsible for eliciting the related requirements, approving the programme's outputs, and operating the system once complete. The programme operated independently, and the related technology decisions were based on the competence and preferences of individual suppliers without involving NAV's IT department.

> *"NAV was not involved in the development at all. They [the Retirement pension programme] just sat at Storo [a geographic location in Oslo] in a separate building, in a separate organization. And what hit NAV in 2011 was a gigantic solution that they [NAV] had no insight into. They did not know how it worked, and they did not know which licences were selected" (Former consultant, NAV).*

Although the Retirement pension programme was considered a success externally, it was based on technology that NAV did not choose and proved difficult and expensive to maintain and operate.

3.1.2 Standardizing technology platforms

Following the pension programme, NAV decided to standardize a single application platform and coordinate releases across various systems and projects. Technical heterogeneity, combined with manual deployment and provisioning, made the organization's application management error prone and time-consuming. The application platform, which was named "the Cloud," was based on virtual server technology, and JBoss and WebSphere middleware ran on a Red Hat Linux operating system. The managers of software development projects were instructed to use the platform to reduce heterogeneity and simplify deployment and provisioning. The introduction of virtual server technology reduced related hardware costs, and semi-automated provisioning reduced software development setup time from weeks to minutes. Thus, development teams were offered a ready-to-use platform and no longer needed to spend time choosing and setting up technical infrastructure. As described by one of the IT architects in the IT Architecture division:

> *"Let me give you an example. In the first project I worked in - it must have been in 2006 - we used it a lot of the time, not only to decide which application platform to use but also when choosing the operating system it would run on. A project destined to build a user functionality would spend a lot of time deciding whether the application should run on Windows or Linux or if it should run on a mainframe solution. We spent*

> *a lot of time doing this. And projects don't spend time doing this anymore. It's been standardized. Everyone just has to relate to it"*.

Most systems were migrated to the application platform, and by the end of 2015, the plethora of hardware and middleware technologies had in general been reduced to three technical platforms: 1) the application platform (known as the "Cloud"; 2), an Oracle forms-based system for managing follow-up activities related to employment (named Arena); and 3) an IBM mainframe system from 1978 used to manage individual benefits (InfoTrygd).

However, although the platform abstracted the underlying complexity of software development and simplified provisioning and deployment, the organization's applications were still large and interconnected. To manage interdependencies between various systems, the organization centralized its operations and banned projects from releasing their own applications. The operations department assumed the responsibility for testing, approving, and releasing deliverables. To maximize resource utilization, software changes were grouped into four yearly main releases and two additional releases, where all changes had to be approved and deployed by the operations department. A single release could include 80,000 development hours.

> *"In 2014, we had a major delivery of approximately 80,000 development hours. [...] it's the biggest we've had. Most of them are around 40–50,000"* (Project manager, NAV).

Coordinated releases and centralized control reduced the number of errors made in these processes. However, this delivery model also reduced the organization's flexibility and responsiveness. It could be months from when a feature was developed to when it was available to users and development teams. The model also increased the complexity of software development, as developers were forced to anticipate needs and adopt a "*think about everything upfront*" attitude.

Coordinated releases and fixed-price contracts gave suppliers few incentives for prioritizing maintainability and long-term efficiency, and the organization's systems increased in both size and complexity.

3.1.3 Increasing the divide between developers and users

The control-centric strategy of NAV was further strengthened by a reorganization of its IT department in 2012. The flat structure of this department was replaced by a "plan-build-run" hierarchy where employees were dedicated to specific parts of the development process. The "plan" department was responsible for high level designing and specifying systems, while the "build" department supervised detailed level design and development activities. Once completed, a system was handed to the "run" department, which was responsible for releasing and operating it. The structure corresponded to the organization's delivery pipeline, where software was developed in non-overlapping stages with formal handovers between departments. To ensure predictability and cost efficiency, the organization instilled a clear separation between customers and suppliers, and projects could not deviate from their specifications without approval from NAV. Change requests had to be carefully documented and approved and be in alignment with supplier contracts.

To ensure consistency across projects, NAV established an IT architecture decision process that was based on the TOGAF[1] framework. A centralized architecture function was created to develop guidelines and supervise projects. The technology decision board developed a "technology catalogue" that listed approved technologies. Projects were designed to employ novel technologies or use established technologies in new ways needed approval from an architecture decision board. These decision boards formed the frontline in an ongoing dispute between projects and the line organization, as projects were focused on short-term targets such as time and cost factors, while the IT architecture unit was dedicated to long-term goals related to maintainability and efficiency.

This centralized approach resulted in lengthy decision processes and reduced the flexibility and efficiency of the development process. Going before the architecture decision board was described as "*going before the king in Parliament and sending change requests for even the smallest thing*" (Dørum 2017, p. 57).

In 2015, an expert committee was asked to evaluate NAV and its ability to generate public value. The resulting report (Vågeng 2015) criticized NAV for failing to meet users' needs and forced the NAV Director to resign. The chair of the expert committee was later nominated as the new NAV director and was given the task of implementing the measures that she had proposed in the report. In addition, a new IT-director was appointed. The IT director playing a crucial role in the transformation of NAV.

3.2 Towards a service-dominant logic

In 2017, NAV began to transition towards a service-dominant logic. Central to this transition was an insourcing of software development and maintenance, a decoupling of monolithic legacy systems into relatively loosely coupled applications, and the implementation of continuous development practices where development teams were responsible for the entire service delivery strategy. These elements are described in more detail below.

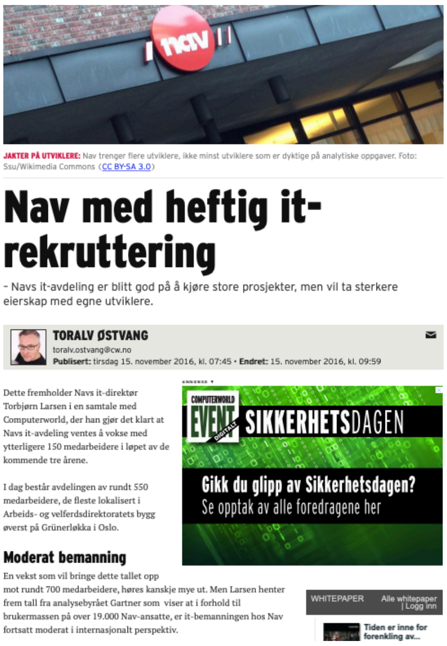
3.2.1 Insourcing software development

One of the first changes suggested by the new IT Director was a change to the sourcing strategy. The development and maintenance of core systems had been outsourced to 15–20 different consultant companies, each pursuing its own goals and targets.

[1] TOGAF is a framework for enterprise architecture that provides an approach for designing, planning, implementing, and governing an enterprise information technology architecture. https://www.opengroup.org/togaf

> *"We [NAV] had decided not to code ourselves. So, there we were, with 200–250 person-years, supervising 300–400 consultants. We were busy specifying needs and running tendering competitions and stuff like that, and sometimes, the needs proved too expensive. Then, we had to go back and clarify some more needs. So, there we were, managing and coordinating, being an intermediary [between the business side of NAV and the consultant company]. With roles that made sure that the system was technically sound and things like that. But, of course, it is quite difficult to control 15–20 suppliers who all have their own interests and needs. How do you control them across 300 systems? How do you ensure the quality of the code base? Well, you can't. That's impossible*" (Manager in the IT department).

The outsourcing of software development and maintenance was replaced by an insourcing strategy. Two key objectives behind this insourcing strategy were continuity and learning, meaning that team members gained insight into all the aspects of the software development process. To accommodate this insourcing strategy, the organization began to employ developers. The goal was to employ 150 developers over the course of only a few years. This recruitment strategy received a considerable amount of attention in the IT community (see Figure 2).



JAKTER PÅ UTVIKLERE: Nav trenger flere utviklere, ikke minst utviklere som er dyktige på analytiske oppgaver. Foto: Ssu/Wikimedia Commons (CC BY-SA 3.0)

Nav med heftig it-rekruttering

– Navs it-avdeling er blitt god på å kjøre store prosjekter, men vil ta sterkere eierskap med egne utviklere.

TORALV ØSTVANG
toralv.ostvang@cw.no
Publisert: tirsdag 15. november 2016, kl. 07:45 • **Endret:** 15. november 2016, kl. 09:59

Dette fremholder Navs it-direktør Torbjørn Larsen i en samtale med Computerworld, der han gjør det klart at Navs it-avdeling ventes å vokse med ytterligere 150 medarbeidere i løpet av de kommende tre årene.

I dag består avdelingen av rundt 550 medarbeidere, de fleste lokalisert i Arbeids- og velferdsdirektoratets bygg øverst på Grünerløkka i Oslo.

Moderat bemanning

En vekst som vil bringe dette tallet opp mot rundt 700 medarbeidere, høres kanskje mye ut. Men Larsen henter frem tall fra analysebyrået Gartner som viser at i forhold til brukermassen på over 19.000 Nav-ansatte, er it-bemanningen hos Nav fortsatt moderat i internasjonalt perspektiv.

ANNONSE
COMPUTERWORLD EVENT DIGITALT
SIKKERHETSDAGEN
Gikk du glipp av Sikkerhetsdagen?
Se opptak av alle foredragene her
WHITEPAPER
Alle whitepaper | Logg inn
Tiden er inne for forenkling av...

Figure 2. NAV received a considerable amount of attention related to their recruitment campaign. For instance, the Norwegian magazine Computer World addressed NAV's ambition to assume internal ownership under the heading "NAV with intense IT recruitment".

With this strategic shift, NAV gradually assumed responsibility for developing and maintaining their own core systems. The focus changed from controlling subcontractors to developing and maintaining systems.

Over time, the organization's ambition was to finance software development and maintenance through its operating budget. This would allow for stable and predictable service delivery where internal employees were dedicated to a domain or product over a prolonged period of time. As existing contracts expired, they were gradually replaced by capacity contracts where consultants were hired by the hour.

> *"Development has been outsourced to consultant companies through traditional management contracts where the supplier has had independent responsibility for the system and maintenance. Now, NAV is taking over this. We have a different sourcing model where we hire people depending on our capacity instead of giving the supplier total responsibility"* (Agile coach in the IT department).

The long-term objective of this shift was that development teams would be mainly staffed by NAV employees. Consultants would only be used during peak periods and in cases where NAV lacked the required expertise.

3.2.2 Decoupling applications

To reduce the complexity of its technical infrastructure, NAV began to decouple monolithic systems into relatively loosely coupled applications. To facilitate this process, the organization introduced a second application platform. This application platform, which was called NAIS (available at www.nais.io), was based on Kubernetes, an open-source framework for managing software containers.

> *"Kubernetes is the open-source framework that comes from Google. It is all of Google's experience over the last 15 years with how to manage infrastructure rewritten by the same people. It's like taking the world's best operations person and fully automating him. That's what Kubernetes is. It provides a lot of tools for running in production, which makes it more robust and more scalable and everything."*

A software container has its own filesystem, CPU, memory, and processing space, and if containers are decoupled from underlying infrastructure, they are portable across clouds and operating systems.

The NAIS platform offered developers a subset of the functionality made available through Kubernetes, and the platform was tailored to NAV-specific needs. The platform was believed to increase the speed of software development by limiting unnecessary creativity on the part of development teams.

As part of the migration to the NAIS platform, large and monolithic systems were dismantled into smaller applications, which reduced their complexity. With modular architecture, it is possible to deploy and manage applications independently, which enables the use of a distributed governance strategy where a development team can manage and deploy their own applications. With the implementation of this distributed governance strategy, NAV's release rates increased from once every few months to several times a day. As these release rates increased, the number of serious system errors was reduced. One of the managers in the IT department described the transition from the old to the new governance strategy as *"taking a super-tanker and splitting it up into 100 speedboats"*.

An important prerequisite for implementing this distributed governance strategy was the establishment of automated operations management: The platform provided fully automated services for tasks such as provisioning, deployment, and load balancing. This meant that teams could develop and release applications without involving the operations department.

To ensure that the transition from centralized to distributed governance did not result in chaos, NAV introduced the concept of "white-listing." Applications were "white-listed" if they were low in complexity and had few external dependencies. Only white-listed applications could be managed by a development team. Complex applications had to be managed by the operations department.

3.2.3 Recoupling the organization

The decoupling of legacy systems into modular applications enabled a recoupling of the organization. This recoupling progressed in two stages. In the first stage, NAV introduced independent development teams that assumed responsibility for developing and operating their own applications. In the second step, these teams and their applications were recombined into product domains.

The introduction of independent development teams occurred gradually. The first independent teams were established in the context of small projects with few dependencies on other projects and systems. One example was the DigiSyFO project. This project, which developed functionality for digitally following-up on sick leave, had one multidisciplinary team. Although most of the developers were hired consultants, the project was planned and managed by NAV employees.

> *"The project was initially set up with a single team with less than ten members consisting of two UX designers, two subject matter experts, one software architect, one developer, operations support, and agile coaches. This grew to approximately 22 project members nine months into the project"* (Agile coach in the IT department).

Whereas other projects employed a staged delivery model with handovers between stages, the members of the DigiSyFO project took responsibility for all parts of its development cycle. In addition, the project was developed iteratively, and activities such as design, development, and operations were interleaved. Software was developed and released continuously, and feedback was constantly collected from end-users and other stakeholders and reintegrated. The project owner, Kristian Munthe, described this process in an interview with MEMU (Haugen 2018), NAV's internal magazine.

> *- It has been absolutely crucial that the project has used "agile methods", which means that they have continuously made improvements instead of waiting until the system was complete. We have received over 80,000 written suggestions. Some of these are on things that could have been easier and better explained. For example, it may be about small formulations that should be changed.*
>
> *And then you change it straight away?*
>
> *- Yes, if we agree with the input. We have made a lot of changes since the beginning."*

The project was deemed a success, and in 2017, it won the government's digitalization reward based on outstanding achievements. The chairman of the jury summarized the project as follows:

> *"This year's winners are a good example of how things can be done in new ways to deliver services that streamline public management and make life easier for users. Both the government and Difi [short for the Agency for Public Management and eGovernment] have high expectations for the digitalization of the public sector"* (Director at the Norwegian Digitalization Agency)

As agile development practices proved successful for smaller projects, NAV decided to apply these methods to larger projects. In 2019, the Parental Benefit project transitioned from staged to continuous deliveries midway through its timeline.

The Parental Benefit project, which was the largest ongoing software development project in the Norwegian public sector at the time, was intended to develop a new system for processing applications for parental benefits. A parental benefit is a welfare benefit intended to compensate parents for losses of income in relation to the birth or adoption of a child. The project, which had an estimated cost of 1.3 billion NOK, was initiated before the elected strategic change and thus followed the "old" delivery model with staged deliveries and handovers between departments.

A leading principle behind the formation of independent teams was that each team would have the competence and authority to develop services independently of other teams and be responsible for an entire service delivery cycle - from the inception of an idea until the service was eventually discontinued. By duplicating competencies across teams and introducing a distributed decision process, the organization ensured that development teams no longer had to await approvals or assistance.

To facilitate the formation of independent teams, the IT department was reorganized, and the hierarchical structure was replaced with a matrix organization. While in the old structure, employees were organized according to their roles in the deployment cycle (planning, development, or operations), in the new structure, employees were grouped according to their competence fields.

This decoupling of applications allowed teams to work more independently. However, not all dependencies between applications could be broken. Many applications were part of larger value paths and had to be developed in relation to other systems and teams. As one of developers of the Parental Benefit project stated:

> *"In my experience, when people don't see the value chain they're contributing to, they begin to sub-optimize. It is artificial to say that you have a truly autonomous team in the area of Parental Benefit. All components and all applications support the same decision process."*

To address these dependencies, applications were grouped into functional domains, and a domain would encompass several applications and teams. The idea of "domain-driven design" (Evans 2004) permeated both the formation of teams and the decoupling of legacy systems.

> *"The idea behind domain-driven design is that it is more important to organize for flow. Optimal flow in the organization and in the code. And this is done by minimizing coordination and chattiness. You must bundle the things that belong together and keep the things that don't belong together apart. You bundle people and code in domains. People within a domain will be more closely linked"* (Manager in the IT department).

By focusing on domains, NAV was able to establish value paths that transcended the organization. However, the domains had to be established gradually and one at a time. In this way, the organization could learn from the experiences gained during the establishment of one domain before establishing the next.

> *"The goal is to remove boundaries between departments. But you can't do that by changing everything at once. All 19,000 employees. Because that won't work. Instead, you can do what we are doing. Establish one domain at a time. Gain experience and prove to the organization that it works. Then you establish the next domain"* (Senior executive of the Parental Benefit project).

As a principle, no legacy systems were brought into the domains. The domain teams only managed new applications. In practice, however, exceptions were made. In some cases, it was more practical to give domain teams responsibility for certain legacy systems while they were being replaced.

By creating cross-cutting domains, NAV was able to improve the flow and collaboration within the organization without physically reorganizing employees. By establishing one domain at a time, NAV could learn and adapt, gradually restructuring the organization and redesigning its service delivery. At the time of this writing (September 2019), the organization had established three domains: Health, famlily, and work.

4 Research methods

The goal of this research was to analyse the role of digital platforms in improving value co-creation in public sector organizations. This research required in-depth knowledge of how digital platforms influence the value co-creation process, the views and opinions of stakeholders concerning digital platforms, and the changing contexts in which these platforms are used. Studies that collect this type of data can be classified as “interpretive case studies” (Walsham 1995). Interpretive studies are based on the assumption that people subjectively interpret the world. A researcher must thus reconstruct this phenomenon by accessing these interpretations.

The research approach adopted in this study was inspired by Pan and Tan (2011)’s structured-pragmatic-situational approach to conducting a case study. This approach suggests that case study research can be conducted through an eight-step process that begins with the researcher gaining access to the case. Once such access has been gained, the “framing cycle” of the method is initiated. The framing cycle begins with the researcher gathering information on the phenomenon of interest and reviewing relevant literature. Based on the resulting mental concept of the examined phenomenon, the researcher collects and organizes an initial set of data and begins engaging in preliminary theorizing. This preliminary stage of theorizing results in the construction of an initial theoretical lens that is used as a “sensitizing device” (Klein and Myers 1999, p. 75) in subsequent data collection and analysis steps. Additional data are then used to further develop this initial lens by adding and refining categories and constructs. The framing cycle continues until the researcher feels sufficiently confident that the theoretical lens captures the phenomenon of interest and makes an adequate contribution to theory and practice.

This framing cycle is followed by an “augmenting cycle”. In the augmenting cycle, the researcher gathers additional data with the aim of developing the theoretical lens into a full-fledged theory. Data are systematically organized and coded using a series of sensemaking strategies for qualitative data analysis (Langley 1999). Once the emergent model has been validated against the empirical data and existing literature (Klein and Myers 1999), the model is presented to and verified by informants. This iterative process involving the emergent model, existing literature, and data continues until "theoretical saturation" (Glaser and Strauss 1967) is reached. Theoretical saturation can be described as the point where additional data neither reveal new properties nor provide further insight into an emergent theory.

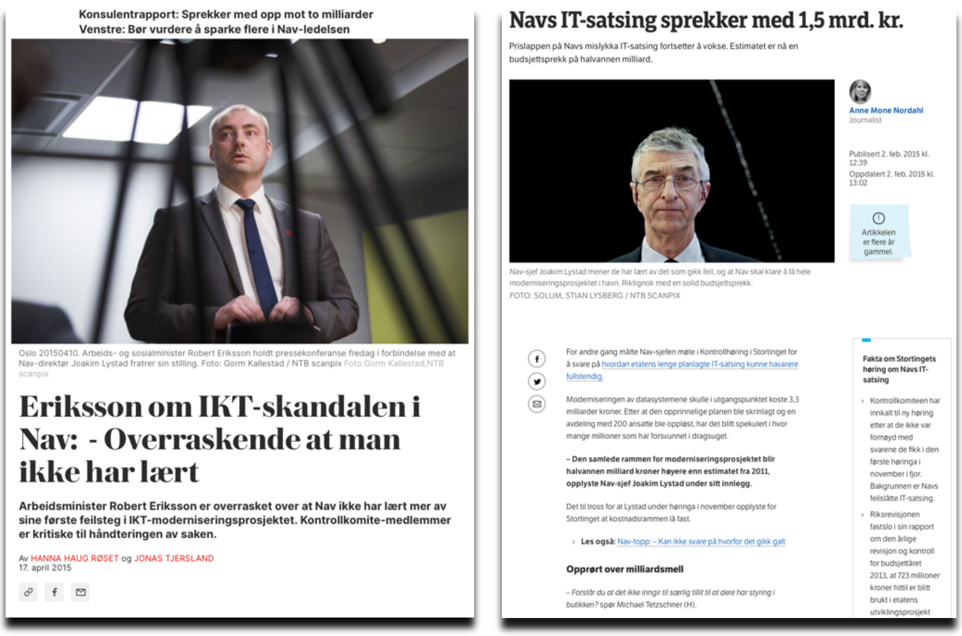
In the remainder of this section, I provide a more detailed description of the research process.

4.1 Case selection

It can be said that NAV provides a unique case for studying how digital platforms might enable improved value co-creation in public sector organizations for several reasons. First, as NAV provides services that range from child benefits and unemployment benefits to social services and retirement pensions, the organization must meet the needs of large and heterogeneous user groups. The organization has almost 2.8 million active users, and most Norwegian citizens will come into contact with NAV at some point during their lives.

Second, NAV was previously characterized by technical and organizational rigidity. When NAV was established, the national insurance agency and the labour agency each had their own case processing systems based on different technology platforms. The national insurance agency had an IBM mainframe solution called InfoTrygd, while the labour agency had an Oracle form-based system called Arena. The new agency chose to implement a service-oriented architecture based on Java as its strategic platform, and the solutions that the organization developed after 2006 were mainly based on this. The most comprehensive system established on this platform was the pension system (PESYS), which handled retirement pensions and disability pensions. In addition to case processing systems, the organization had a large number of registers and systems that supported the processing of other user benefits and document solutions that supported NAV services (Dørum 2017, p. 46). A central objective of the NAV reform was improved collaboration across various professions. However, despite these ambitions of collaboration and co-creation, the structure of the old agencies remained largely intact, resulting in a silo organization with limited interaction across various professions and functions.

The rigidity and inefficiency of the organization’s service delivery was further aggravated by a history of failed IT projects. The latest failure occurred in 2015, when NAV initiated the first of three modernization projects. This project, which had an estimated cost of 1.3 billion NOK, was terminated after only 6 months, attracting massive media coverage (see Figure 3) and resulting in an open Parliamentary hearing. Therefore, when the organization received funding for its next modernization project in 2016 (the Parental Benefit project), it was placed under the close supervision of the Ministry of Finance and was given a limited ability to deviate from the elected plans and specifications. Changing the service delivery model of the organization under such circumstances seemed both difficult and unlikely, making NAV an extreme case in terms of rigidity and path dependence.



Konsulentrapport: Sprekker med opp mot to milliarder
Venstre: Bør vurdere å sparke flere i Nav-ledelsen

Oslo 20150410. Arbeids- og sosialminister Robert Eriksson holdt pressekonferanse fredag i forbindelse med at Nav-direktør Joakim Lystad fratrer sin stilling. Foto: Gorm Kallestad / NTB scanpix Foto:Gorm Kallestad,NTB scanpix

Eriksson om IKT-skandalen i Nav: - Overraskende at man ikke har lært

Arbeidsminister Robert Eriksson er overrasket over at Nav ikke har lært mer av sine første feilsteg i IKT-moderniseringsprosjektet. Kontrollkomite-medlemmer er kritiske til håndteringen av saken.

Av HANNA HAUG RØSET og JONAS TJERSLAND
17. april 2015

Navs IT-satsing sprekker med 1,5 mrd. kr.

Prislappen på Navs mislykka IT-satsing fortsetter å vokse. Estimatet er nå en budsjettsprekk på halvannen milliard.

Anne Mone Nordahl
Journalist

Publisert 2. feb. 2015 kl. 12:39
Oppdatert 2. feb. 2015 kl. 13:02

Artikkelen er flere år gammel.

Nav-sjef Joakim Lystad mener de har lært av det som gikk feil, og at Nav skal klare å få hele moderniseringsprosjektet i havn. Riktignok med en solid budsjettsprekk.
FOTO: SOLUM, STIAN LYSBERG / NTB SCANPIX

For andre gang måtte Nav-sjefen møte i Kontrollhøring i Stortinget for å svare på hvordan etatens lenge planlagte IT-satsing kunne havarere fullstendig.

Moderniseringen av datasystemene skulle i utgangspunktet koste 3,3 milliarder kroner. Etter at den opprinnelige planen ble skrinlagt og en avdeling med 200 ansatte ble oppløst, har det blitt spekulert i hvor mange millioner som har forsvunnet i dragsuget.

– Den samlede rammen for moderniseringsprosjektet blir halvannen milliard kroner høyere enn estimatet fra 2011, opplyste Nav-sjef Joakim Lystad under sitt innlegg.

Det til tross for at Lystad under høringa i november opplyste for Stortinget at kostnadsrammen lå fast.

› **Les også:** Nav-topp: – Kan ikke svare på hvorfor det gikk galt

Opprørt over milliardsmell

– Forstår du at det ikke inngir til særlig tillit til at dere har styring i butikken? spør Michael Tetzschner (H).

Fakta om Stortingets høring om Navs IT-satsing

- Kontrollkomiteen har innkalt til ny høring etter at de ikke var fornøyd med svarene de fikk i den første høringa i november i fjor. Bakgrunnen er Navs feilslåtte IT-satsing.
- Riksrevisjonen fastslo i sin rapport om den årlige revisjon og kontroll for budsjettåret 2013, at 723 millioner kroner hittil er blitt brukt i etatens utviklingsprosjekt

Figure 3. NAV attracted a massive amount of media attention following the termination of its first modernization project in 2015. Among the headlines regarding this event were "Surprised to see that they have not learned", and "NAV's IT failure exceeds 1.5 billion NOK".

4.2 Research design

Research on value co-creation and user involvement has traditionally assumed direct user interaction, where user involvement is limited to a particular time and place. However, in large public organizations, services are directed towards large and heterogeneous user groups that span multiple locations and long periods of time. To explore such value co-creation as it unfolds in a large-scale context, the research presented in this thesis focuses on *scalar devices*: the techniques and technologies that enable actors to manage large-scale enterprises (Ribes 2014). In other words, instead of studying value co-creation directly in the context of interactions with citizens, I approach this issue by studying it indirectly. This study thus focuses on the development practices and technology platforms *inside* NAV, paying less attention the reactions and opinions of individual citizens. This choice of perspective does not imply that the opinions of users are irrelevant. Rather, this research design reflects the assertion that value co-creation in large-scale settings requires indirect and mediated forms of interaction.

It is widely recognized that prior knowledge of the success or failure of a strategic change influences informants' recollections (Van de Ven 1992). This study was therefore designed as a two-year longitudinal case study where I followed the examined change process as it unfolded. To capture the temporal and contextual frame of reference explored, the real-time study was complemented by a historical reconstruction. Together, the real-time study and the historical reconstruction covered a period of 12 years.

In addition to studying NAV, I performed two pilot studies of smaller organizations. These pilot studies provided training in qualitative research methods as well as an opportunity for conceptualizing the examined phenomenon and formulating an initial theoretical lens (Pan and Tan 2011). The pilot studies and the design of the main study within NAV are described below.

4.2.1 Pilot studies

In the first pilot study, which was initiated in the spring of 2016, fieldwork was performed within a medium-sized consultant company. The consultancy was awarded a contract for the development of a case tool to be used for the management of marine resources in the public sector. The project had two development teams, and the developers within the two teams were distributed across multiple locations. My aim was to investigate how agile development practices were applied in a large-scale and distributed context, and I used the software development process as my unit of analysis. As I had limited experience with qualitative research methods, this fieldwork enabled me to practice participant observation and to explore relevant literature on agile development methods. However, although the project gave some insight into the challenges of working in a distributed setting, it was relatively small and had few dependencies on other projects and systems. It was therefore poorly suited to addressing issues relating to complexity and size.

In the second pilot, which was conducted during the autumn of 2016, I studied software development practices within a large Norwegian bank. The unit of analysis that I used was the software development process, and I especially emphasised a team that was responsible for developing web applications for the private sector. The organization and the system portfolio examined in this case were considerably more complex than those explored in the marine case, and while the marine case represented "green field" development, the banking case provided insight into the challenges of developing information systems in relation to existing systems and practices.

Two central concerns in the banking case were the introduction of agile development practices and how innovation could be achieved alongside existing systems and practices. At the heart of the development strategy employed lay the belief that internal ownership and continuous deliveries would increase the efficacy and effectiveness of the services provided. The introduction of agile development practices was accompanied by changes to the organization's digital infrastructure, where monolithic systems were dismantled into a platform-oriented structure. The study, although short in duration, allowed me to further extend my theoretical lens and refine my research questions, thus it prepared me for the study of NAV.

4.2.2 The main study

In the main study, fieldwork was conducted within the IT department of NAV, and the value co-creation process was used as the study's unit of analysis. To capture the breadth of the examined development activities while ensuring a manageable scope in terms of data collection, the study was centred on three elements of the IT organization: The DigiSyFO project, the Parental Benefit project, and the IT Architecture division.

The DigiSyFO project was the first project done by NAV to implement continuous deliveries. This project had a significant impact on the organization's transformation of its service delivery process and thus provided an important understanding of the events leading up to the transformation. Although the DigiSyFO project had few dependencies on other systems and projects, it demonstrated to the organization that agile development practices were both efficient and viable within NAV.

The Parental Benefit project was the largest ongoing software development project within the Norwegian public sector at the time of the study. While the DigiSyFO project was small with few dependencies, the Parental benefit project had more than 60 software developers and was dependent on 42 other systems. The two projects also differed in terms of their development methods and sourcing strategies. While the DigiSyFO project employed agile development methods from its beginning, the Parental Benefit project began with a staged delivery model and adopted an agile delivery strategy half-way through. Together with the DigiSyFO project, the Parental Benefit project exemplified both the breadth of NAV's development activity and the contrast between its new and old service delivery strategies.

The IT Architecture division was responsible for supervising projects and ensuring consistency across systems and projects. This division was chosen for analysis in an attempt to capture the tension between the line organization and individual projects. With the transformation of the organization's service delivery strategy, the IT Architecture division went from supervising to assisting software development teams. The IT Architecture division also provided insight into the tasks and responsibilities of the line organization and how the long-term needs of line organizations often conflicted with the short-term objectives of IT projects. While projects were evaluated based on their ability to deliver functionality in terms of time and cost, the line organization was evaluated based on its ability to operate and maintain these systems over time.

Focusing on these distinctly different parts of the organization and choosing informants with different experiences and interests regarding the transformation ensured the study's sensitivity to differences in interpretations (Klein and Myers 1999). While projects gained influence and autonomy from the ongoing change, the members of the IT Architecture division became less influential. By collecting accounts from both the "winners" and "losers", I was less prone to systematic biases and systematic distortions.

The NAV IT department is situated in Oslo. Since I lived in Trondheim, almost 540 kilometres north of Oslo, I also requested access to the local NAV administrative system. By borrowing a workstation, I was able to access the organization's internal documents and task management systems, thus reducing the need for travelling. Employees in the local NAV administration also provided context and background regarding the ongoing transformation of the IT department. I visited the local NAV administration one to two times a week over the period of a year. During these visits, I reviewed documents and talked to local employees. My contacts in the local administration also provided useful information. Discussions during lunch and coffee breaks also provided important insight into the context of the transformation and the way it was perceived by other parts of the organization.

This real-time study of the transformation was complemented by a historical reconstruction of the events that occurred between 2006 and 2016. This reconstruction enabled me to critically reflect on the context in which the transformation had taken place – a central principle when conducting and evaluating interpretive research (Klein and Myers 1999). Figure 4 shows the fieldwork of the different cases over time.

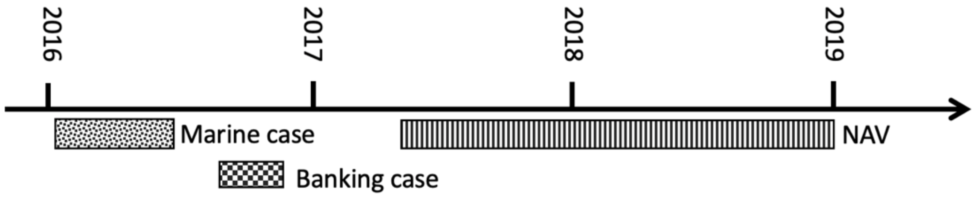


Figure 4. Fieldwork of the main study and pilot studies over time

4.3 Gaining access

The first step when conducting case research is gaining access (Pan and Tan 2011). Since interpretive longitudinal case studies require commitment on the part of both the organization and the researcher, gaining such access can be both difficult and time consuming. This also proved to be the case with NAV.

The members of the Agile 2.0 research project asked for access to the Parental Benefit project in the autumn of 2016. Agile 2.0, which funded my PhD work, was supported by the Research Council of Norway and by the companies Sopra Steria, DNV GL, Equinor, Kantega, Kongsberg Defence & Aerospace, and Sticos. The contract for the development of the Parental Benefit system had been awarded to Sopra Steria, one of the sponsoring partners of the research project, and we therefore assumed that gaining access would be straightforward. However, for unknown reasons, it took almost a year from when the initial request was made until we were granted formal access.

While awaiting formal access through the Agile 2.0 project, I approached contacts in my own professional network. Throughout my PhD studies, I held a parttime position at a small IT consultant company. One of my colleagues in the consultant company introduced me to a manager in the IT department of NAV. The manager was able to grant me temporary access to the IT Architecture division while we were awaiting feedback from the Parental Benefit project. I was given an access card and an e-mail account and was able to move freely within the IT department.

Although this stay within the IT Architecture division was not part of the initial research design, it proved to be a major asset. As described in section 4.2.2, the IT Architecture division provided valuable insight into the conflicting needs of the line organization and individual projects. The relationships that I established in the IT Architecture division were also vital for the recruitment of informants in other parts of the organization.

To make sense of the organization and find relevant information in the myriad of divisions and projects, I worked incrementally and gradually enlarged my circle of informants. Building relationships with informants happens in the same way as relationships are built in other aspects of life: People help you if they get something in return. Feldman et al. (2003) argue that people may choose to help a researcher because the connection with the research gives them status in their jobs, because they feel that helping is part of their job, or because they genuinely like the researcher. I gained access to informants through some of these methods.

The process of recruiting informants typically progressed in 3 steps. First, I found the name of a prospective informant. This was done through a snow-balling approach (Biernacki and Waldorf 1981), where one informant would suggest the next. Without insider information regarding the organization, snowballing was the only viable approach for recruiting informants (Pan and Tan 2011). Second, I found the name of a manager who could grant me access to the informant. Even if an informant wanted to help, he or she often needed approval from his or her manager to talk to me during work hours. Third, I found a person in my existing network who could introduce me to the manager. This was done to increase the likelihood of a positive answer, as people are less likely to say no to someone they know.

4.4 Data collection

This thesis is based on data collected within NAV between March 2017 and August 2019. As summarized in Table 3, three methods of data collection were used.

Real-time studies, which provided the main source of data, were complemented by a historical reconstruction, providing a contextual and temporal frame of reference (Van de Ven 1992). Together, the real-time studies and the historical reconstruction covered a period of almost 13 years, from 2006-2019. As the study was initiated in 2017, the outcome of the change process was still unknown at the beginning of the study, thereby biases involving a priori knowledge of success or failure were avoided. For instance, some of the informants who were sceptical of the transformation in the initial interviews downplayed and sometimes even denied their own scepticism in later interviews.

Document reviews were an important source of information throughout the study of NAV. Before I gained access to the company information, publicly available documents provided me with background on the case and enabled me to prepare for the initial interviews (Pan and Tan 2011, p. 165). This literature included governmental white papers (Government 2004-2005), tender documents from the Parental Benefit project (retrieved from www.doffin.no), newspaper articles, public reports (Johansen and Skålnes 2014; Vågeng 2015), and research papers describing the NAV reform (Askim et al. 2010; Askim et al. 2011; Klausen 2016).

After I gained formal access to NAV, I was given an internal email account and could access most internal document systems, including calendars, project wikis (Confluence), issue-tracking systems (JIRA), and the NAV intranet. To gain insight into NAV's mission, I also became a member of two Facebook groups: one where NAV employees shared internal events and discussed operational issues and another where citizens could ask NAV employees questions. These Facebook groups provided insight into the complexity and diversity of the issues facing NAV and its users.

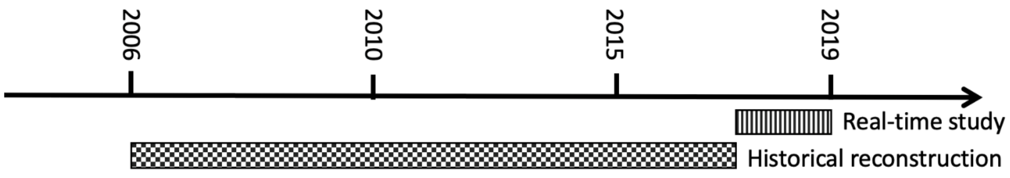


Figure 5. Relation between the real-time study and the historical reconstruction

The examined written documents were complemented by a large number of recorded conference presentations given by NAV employees at a series of conferences. Four of these presentations were transcribed and used in the data analysis. During recent years, NAV has played an active role in sharing the challenges and opportunities associated with digital transformation in the public sector. Conference presentations have also been a key ingredient in the recruitment of IT personnel, and they gave me valuable insights into the specifics of the technologies used to build the platform. Consistent with the second step of the structured-pragmatic-situational approach to conducting case studies (Pan and Tan 2011), these documents enabled me to conceptualize the phenomenon of interest and prepare for the initial interviews.

Participant observation was another data source used during the initial stage of data collection. By observing meetings, I was able to pick up on relevant discussions and concerns within the IT department. During my first three visits to NAV, I spent a total of six days shadowing two members of the IT Architecture division. In addition, I borrowed different workstations in the project area each time I visited. This allowed me to observe different teams and pick up on discussions and activities. My observations were extensively documented in a field diary and became a part of the later analysis. My observations were also sensitized to relevant problems and concerns in the IT department and provided important background during the interviews.

Interviews formed the primary source of data for this study. These interviews provided access to the feelings and opinions of the participating informants (Walsham 1995). Beginning in August 2017, I paid regular visits to the IT department, conducting 5-6 interviews during each visit. Although some rounds of interviewing were performed with my supervisors, I conducted most of the interviews alone. During the two-year study period, I conducted 42 formal interviews. Of these interviews, 23 were recorded and transcribed. Most of the interviews lasted between 45 and 60 minutes. Because of their sensitive nature, not all the interviews could be recorded. In cases where an interview could not be recorded, I took notes during the interview and elaborated on these notes immediately after the interview ended.

The formal interviews were complemented by informal conversations. As mentioned above, I was given an access card and was able to move freely within the IT department. I was able to gain a considerable amount of insight through conversations by the coffee machine and encounters in the hallway. As previously noted, obtaining access is essentially relational and requires relation building with prospective informants. Lunches and coffee breaks provided an arena where I could introduce myself and my research while acquainting myself with the organization. My observations and conversations were extensively documented in a field diary after each visit to NAV. These different data sources are summarized in Table 3.

Table 3. Data sources and descriptions

Data source	Description
Interviews (formal and informal)	● 42 semi-structured interviews conducted with 17 unique informants; 23 were transcribed ● Conversations during coffee and lunch breaks
Participant observations	● Architecture decision meetings ● Project meetings (diverse meetings attended by architects) ● Section meetings within the IT Architecture division
Documents	● Documents retrieved from the Internet (Government white papers, procurement documents, newspaper articles, research papers, and evaluation reports) ● NAV's internal intranet ● JIRA (internal issue-tracking system) ● Confluence (wiki-type web tool used to document project activities) ● Facebook groups for NAV employees and NAV users ● Video presentations given by NAV representatives at various conferences (recorded and available online)

To ensure the validity of the data collected, all the pieces of evidence used in the construction of the case study were triangulated by at least two sources of data (Yin 2003). For instance, since one part of the organization saw the ongoing transition as a threat to the stability and predictability of the organization's services while the other argued that flexibility would in fact increase, I systematically collected accounts from both sides. In addition, I used internal and external reports and project documents to verify the facts and figures presented by the informants.

Multiple perspectives were also gained by choosing informants from different levels of the organization, and they ranged from senior management to project managers, developers, and case workers. The distribution of the informants across different parts and levels of the organization is shown in Table 4.

Table 4. Distribution of the informants across the organization

Organizational division	Number of informants	Stakeholder groups
Parental benefit project	10	Project managers Developers IT architects
DigiSyFO project	2	IT architect Developer
Line organization	3	CTO IT architect Department manager
Other	2	Case worker Business expert

4.5 Data analysis

The data analysis of this study was iterative and overlapped with the data collection that I performed throughout my PhD research, which granted me the flexibility to respond to emergent themes (Eisenhardt 1989). This meant that both my sensemaking strategies and my perception of what constituted appropriate theory evolved throughout my PhD studies.

The data analysis of this research can be described as an iterative three-step process. In the first step, I deductively started with an examination of the existing literature, which enabled me to conceptualize the phenomenon of interest (Pan and Tan 2011). What constituted appropriate literature evolved throughout my PhD work. In the initial stages, my focus was on agile software development. This perspective was consistent with the research goals formulated by the Agile 2.0 project and seemed to be an appropriate conceptualization of the small-scale development project explored in the first pilot study. As described in section 0, the first pilot study examined how agile development practices were applied in a small-scale setting with limited complexity and few external dependencies.

My theoretical perspective was extended as I progressed to the second pilot study. While completing the fieldwork of this study, which took place in the IT department of a large Norwegian bank, I found that the development teams had to relate a large legacy of systems and practices. To conceptualize the challenges associated with agile development practices in large-scale settings, concepts of “path dependence” (Mahoney 2000; Sydow et al. 2009) were explored. More specifically, how so-called “path-breaking" mechanisms can reduce path dependence and pave the way for agile transformations of organizations (Rolland and Vestues 2020).

The third and final theoretical perspective was introduced as I entered NAV. Similar to the banking case, NAV struggled to increase its flexibility and improve co-creation. To address these problems, they reorganized their IT department and were in the progress of restructuring their technical infrastructure. Their monolithic and interconnected systems were gradually being replaced by a platform structure that enabled development teams to work independently. To conceptualize this shift from staged to continuous software deliveries, I turned to the literature on digital platforms and the framework of service-dominant logic. Figure 6 illustrates how these different theoretical perspectives emerged through the fieldwork done in the pilot and main studies.

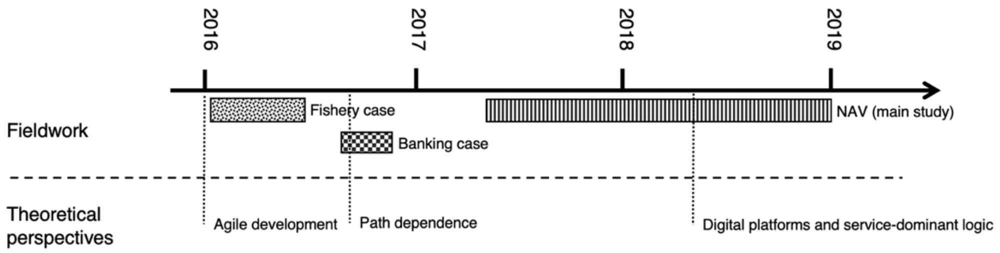


Figure 6. Theoretical perspectives emerging over time

In the second step of the analysis, data were analysed inductively. Since the data indicated an evolutionary process, I used a temporal *bracketing strategy* (Langley 1999) to distinguish between the different stages of evolution. The stages did not have any theoretical implications but this process allowed me to decompose events into comparative units that were used to explore theoretical ideas. This strategy is particularly useful when there is a likelihood that an evolutionary process is shaped by feedback mechanisms or mutual shaping (Langley 1999). There was a certain continuity within each stage and discontinuity at its frontiers. For instance, the temporal bracketing strategy was used to identify the different value logics in NAV between 2012 and 2019 in (Vestues et al. 2021).

The identification of possible stages of evolution was facilitated by a *visual mapping strategy* (Langley 1999). With the visual mapping strategy, the events, decisions, and activities related to the transformation of the IT department were mapped out on a timeline. Figure 7 shows an early version of the map used to analyse the case.

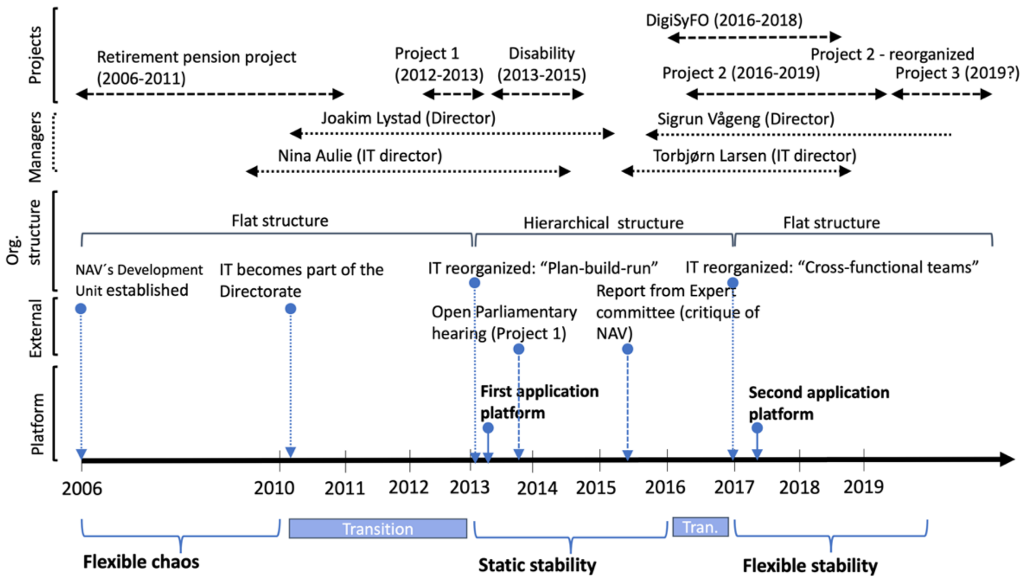


Figure 7. Example of an early version of the visual mapping used in the analysis of the NAV case. The examined events and categories were refined throughout the augmenting cycle.

The map was gradually refined until the timeline contained only the most relevant events and decisions. For instance, in the analysis that formed the basis for the work of (Vestues and Rolland 2021), the final version of the map used contained three categories of events: sourcing strategy, governance strategy, and technical platform events. Together, these three categories covered the most important aspects of the change occurring in NAV.

I also used a *narrative strategy* in my analysis of case data (Langley 1999). When translating the events, decisions, and activities relevant to the transition of the IT department into a chronological story, the overwhelmingly numerous data were condensed into a manageable form, thus avoiding "death by data asphyxiation" (Pettigrew 1990, p. 181). The purpose of adopting a narrative strategy was to capture the order of the examined events and to clarify the relations between decisions, activities, and their consequences (Pan and Tan 2011). New events, decisions, and activities were incorporated into the narrative during each cycle of data collection and analysis. For instance, while the first version of the narrative focused on real-time events in the IT department, the final narrative included both present and past events and covered the years from 2006 to 2019. The narrative was used both to visualize case data and as a basis for communicating with supervisors and co-authors. Selected parts of the narrative were also used in the results sections of various articles (see, for instance, (Rolland and Vestues 2020; Vestues and Rolland 2021)), as well as in the case description presented in this thesis. The relationship between the real-time study and the historical reconstruction is illustrated in Figure 5.

The narrative approach was complemented by an inductive approach (Langley 1999). In the inductive approach, which was inspired by grounded theory, the case narrative, which was mainly written in the words of organizational members, was used to extract process categories. Thus, the narrative represented a first-order analysis, while the derivation of categories represented a second-order analysis (Van Maanen 1979). The categories and corresponding subcategories were filled with quotes used to validate the emerging model. For instance, in exploring platformization within NAV (Vestues and Rolland 2021), I identified two broad categories of events, termed "decoupling" and "recoupling", which were each comprised of two subcategories (see Table 5 for example of how the categories were derived)

Table 5. Example of the inductively derived categories presented in (Vestues and Rolland 2021)

Category	Subcategory	Excerpts from interviews
Decoupling	Decoupling between technical infrastructure and presentation layer	*"We abstracted away some of the complexity by saying that if you are going to build an application, then you should not have to think about which operating system and network protocols to use and low-level technical things."* (Architect IT, section for IT architecture)
	Decoupling of legacy systems into smaller components	"*Systems should be broken up into products, then that can be developed, tested and operated separately. Rather than one big lump, you need several separate components. In this way, you avoid all the coordination which takes a lot of time. If my component is dependent on your component, we have to make sure that your component is released before my component and so on. This is very demanding."* (Software developer, Parental Benefit project)

In the third stage of data analysis, the inductively derived categories were merged with deductively derived theoretical concepts. The objective of this data-model alignment was to find an elegant theoretical explanation for the case data, which would prevent observers from being “dazzled by the full-blown complexity of natural or concrete events” (Calvin S. Hall 1997, p. 13). Data-model alignment was achieved by identifying empirical examples of theoretical concepts. For instance, when describing how platformization unfolded in NAV (Vestues and Rolland 2021), the concepts of “liquefaction” and “resource integration” from the field of service-dominant logic (Lusch and Nambisan 2015) seemed to correspond to our inductively derived concepts of “decoupling" and “recoupling". Since generality can be claimed on the basis of the logic of replication (Yin 2013), I inferred that the decoupling observed in the case provided a means for increasing resource density within NAV and that the structural changes produced by the recoupling of the organization increased the organization’s ability to innovate and co-create value.

A central aim of the data-model alignment as it unfolded in the augmenting cycle of the research approach was to find theoretical explanations for all the empirical findings of the research, thereby extending existing theories (Pan and Tan 2011). For instance, in my exploration of the process of platformization as it unfolded in NAV (Vestues and Rolland 2021), I found that while service-dominant logic identifies “liquefaction” as the sole means for increasing resource density (Lusch and Nambisan 2015), my empirical findings suggested that resource density could also be increased through the decoupling of legacy systems into more loosely coupled applications.

4.6 Reflections on the role of the researcher

A final aspect of the research method concerns my role as a researcher. A researcher’s position resides somewhere along a continuum from an “independent observer” with a descriptive position to an “action researcher” who is actively engaged in a change process (Wareham and Sahay 1999). In my PhD research, I mostly assumed the role of an independent observer. Being a novice researcher and new to the domain of public welfare services, it took almost two years before I gained sufficient insight and confidence to engage in a meaningful analysis of the case. My contribution to the organization thus laid in my role as an active listener and my provision of a safe space for venting frustrations and concerns with the transformation.

My background resembled that of the informants, both in terms of education and work experience. I had worked both as a project manager and a software developer and could effortlessly understand and engage in their problems and concerns. This reduced, and to a large extent eliminated, the distance between us. Social networking tools such as LinkedIn revealed that I shared numerous connections with most of the informants, which induced trust and facilitated the process of gaining access to the organization. A sense of mutual trust and understanding was therefore imperative for gathering insights that went beyond the official narrative. Being able to move freely among the different sites in the IT department, send and receive e-mails from a NAV account, and book meeting rooms in the company calendar also strengthened my sense of being an “insider” and my ability to build trust with the informants.

To verify my observations and interpretations, I discussed them with members of the organization throughout my fieldwork. More formal feedback was also given through workshops and academic papers. Figure 8 gives an example of the illustrations drawn on the white board during one of these workshops. The process of presenting and discussing my findings helped to uncover misunderstandings and provide additional details regarding the narrative. For instance, some informants suggested additional events and decisions that could be added to the timeline. This process also resulted in a greater awareness of different opposing opinions and views, thus illustrating the principles of multiple interpretations and suspicion by Klein and Myers (1999).

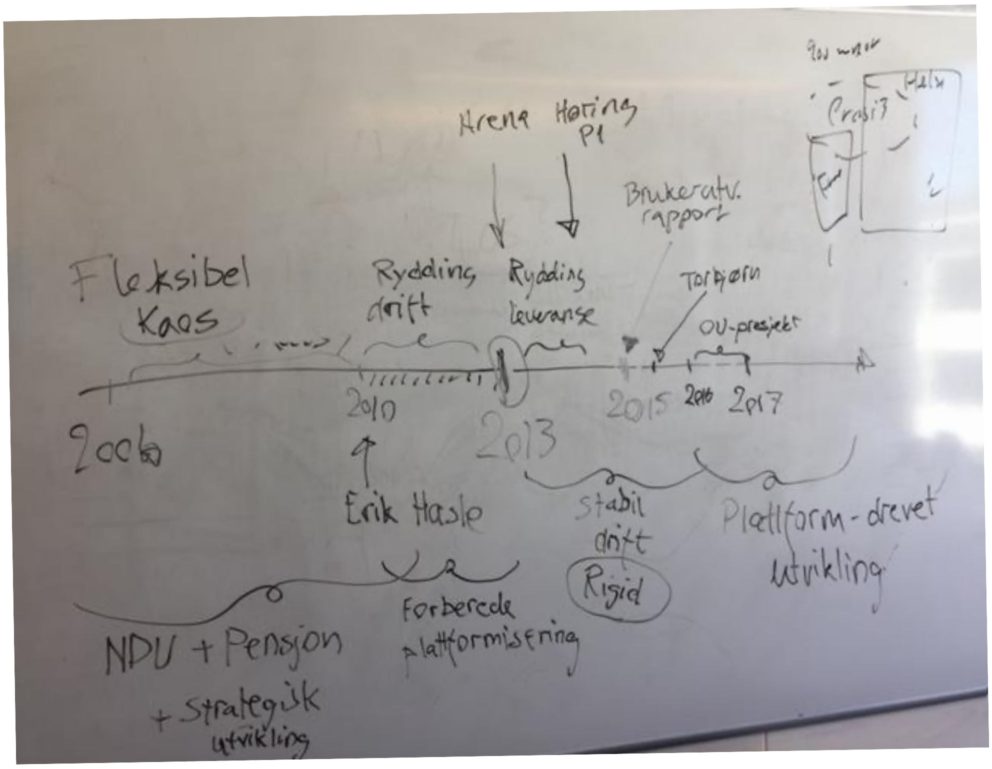


Figure 8 - White board drawings from a presentation of the key findings given to the informants

5 Results

Enclosed in this thesis are the following five papers:

Paper 1. Vestues, Kathrine; Bjørnson, Finn Olav. (2016). *Agile requirements work in a digital transformation project: Managing diverse and dispersed user needs*. Paper presented at the International Research Workshop on IT Project Management (IRWITPM)

Paper 2. Dingsøyr, Torgeir; Mikalsen, Marius; Solem, Anniken; Vestues, Katherine. (2018) *Learning in the Large: An Exploratory Study of Retrospectives in Large-Scale Agile Development*. Agile Processes in Software Engineering and Extreme Programming, 19th International Conference, XP 2018, Proceedings.

Paper 3. Rolland, Knut; Vestues, Kathrine. *Inertia and change in transformation of the IT-function in large organizations: A path theory lens*. Accepted to NOKOBIT 2020

Paper 4. Vestues, Kathrine; Rolland, Knut, *Platformizing the Organization through Decoupling and Recoupling: A longitudinal Case Study of a Government Agency*, (2021) Initially submitted to the Scandinavian Conference of Information Systems (2019) and later fast-tracked and accepted to the Scandinavian Journal of Information Systems (2021).

Paper 5. Vestues, Kathrine; Mikalsen, Marius, Eric Monteiro (2021), *Using digital platforms to promote a service-oriented logic in public sector organizations: A case study*, Accepted to the 54th Hawaii International Conference on System Sciences. Nominated for the "Best paper award" in the "Digital Government" track.

In addition, three papers were published during my PhD work that were not included in this thesis.

A Vestues, Kathrine. (2016). *Planned Research: Scaling Agile Practices in Software Development.* Paper presented at the XP2016 Doctoral consortium.

B Bjørnson, Finn Olav; Vestues, Kathrine; Rolland, Knut-Helge. (2017) *Coordination in the large: a research design*. XP2017 Proceedings of the XP2017 Scientific Workshops, Cologne, Germany — May 22 - 26, 2017

C Vestues, Kathrine. Rolland, Knut Helge. (2019). *Making digital infrastructures more generative*. Paper presented at the Tenth Scandinavian Conference on Information Systems (SCIS2019), Nokia, Finland.

Paper A was written during the first months of my PhD work and thus presents an early version of the research design. Paper B, which was written in collaboration with one of the researchers in Agile 2.0, addresses the challenges of co-creation in large-scale projects but is based on fieldwork conducted by other members of the SINTEF research group. Paper C is based on the preliminary findings of the NAV case. The insights gained through the writing of this paper thus fed into the later papers. Paper D was presented at the Scandinavian Conference on Information Systems in 2019. This paper was fast-tracked to the Scandinavian Journal of Information Systems and later published as a journal article (Paper 4).

These papers were written throughout the duration of my PhD project and reflect the theoretical development of the work, as presented in section 0. The first two papers (Papers 1 and 2) discuss the challenges associated with agile development in large-scale settings. Both papers were published in software engineering outlets and were intended to contribute to the software engineering field. The third paper (Paper 3) discusses the role of inertia in digital transformations and uses NAV as an empirical setting. The last two papers (Papers 4 and 5) employ service-dominant logic as a theoretical lens, discussing how digital platforms and platformization contribute to the co-creation of value. Papers 3, 4, and 5 are directed towards the information systems community. In addition, Paper 5 contributes to the digital government literature by explicitly discussing its findings in the public sector context. Table 6 presents a summary of the papers included in this thesis, the fields where they were published or submitted, their empirical and theoretical grounding, and their contributions.

Table 6. Overview of the empirical and theoretical grounding and contribution of each paper included in the thesis.

Paper	Empirical case	Theoretical lens	Contribution
Paper 1 [SE]	Marine case	Scaling of agile development with an emphasis on requirements elicitation.	• Empirically grounded study on requirements engineering in a large-scale context. • Investigates the challenges involved in collecting and integrating the needs of large and heterogeneous user groups.
Paper 2 [SE]	NAV	Scaling of agile development with an emphasis on practices for learning.	• Empirically grounded study on agile practices in a large-scale context. • Explores the challenges involved in learning across teams.

Paper 3 [IS]	NAV	Digital transformations and path constitution theory.	• Explores how software organizations can develop organizational inertia through path dependence related to software development methods, souring models, software delivery routines, and legacy systems. • Identifies three interdependent mechanisms for breaking out of path dependency: 1) Using digital platforms for software development, 2) attracting and securing competence, and 3) establishing cross-disciplinary software development teams.
Paper 4 [IS]	NAV	Service-dominant logic and platformization as strategies towards improved flexibility and innovation	• Theorizes how existing legacy systems are discontinued in processes of decoupling. • Portrays how the decoupling of legacy systems enables novel recombinations of knowledge and skills in recoupling processes, which, in turn, facilitate new ways of working and organizing. • Theorizes how decoupling and recoupling processes interact and thus facilitate increased levels of platformization.
Paper 5 [IS]	NAV	Service dominant logic and digital platforms as strategies for scaling value co-creation over time and space in public sector organizations	• Explains how adopting a process-oriented approach for value co-creation requires structural changes that encompass sourcing strategies and governance structures. • Shows the key role of digital platforms in scaling value co-creation in public sector organizations

In the following section, I describe the results of each of the articles listed above in light of the general case framing and theoretical perspectives described in chapter 2. The papers are presented in chronological order.

5.1 Paper 1 – Agile requirements work in a digital transformation project: Managing diverse and dispersed user needs

In this paper, we report on an ongoing study regarding a software development project where the aim is to develop a new system for the management of marine resources. The fieldwork was conducted within a consultant company that developed a case tool for the management of marine resources (see section 4.2.1 of the Research methods section for a more detailed description). The members of the development project were responsible for creating software for a complex, diverse, and dispersed customer organization within the public sector. The project employed agile development methods, where specifications, designs, and development were interleaved in one-month iterations. In this paper, we describe the progression of events from when a need was identified until a feature was implemented. Our empirical observations are described through three theoretical lenses: user participation and involvement, power relations in complex organizations, and the balancing of local and global needs in system development. This study explores the challenges of collecting and integrating the requirements of dispersed and heterogeneous user groups.

Theoretically, the study draws on insights from the agile development literature, emphasizing the need for iterative and collaborative practices in the elicitation of user requirements. The study thus addresses two of the central concerns in this thesis, namely, the emergent and context-dependent nature of user needs and the way agile development practices ensure value co-creation over time.

Contribution: As the main author, I had the initial idea and assumed the main responsibility for writing, data collection, and data analysis.

5.2 Paper 2 – Learning in the Large: An Exploratory Study of Retrospectives in Large-Scale Agile Development

This paper focuses on learning and improvement in the context of large-scale agile development. This is a particularly challenging area of work, as such projects consist of several development teams with dependencies between them and typically involve complex integration with existing information systems in projects that are critical for companies or societies (Rolland et al. 2016).

Empirically, the paper reports from the perspective of the NAV IT department, where we investigate the use of retrospective meetings within a large-scale development project. A retrospective meeting is a meeting where members of a development gather to identify opportunities for improvement (Kniberg 2015; Kua 2013). The aim of such a meeting is for a team to engage in collective and iterative learning intended to continuously improve software development processes. An analysis of the issues recorded in the examined project's issue tracking system suggests that most of the issues relate to team-level learning and that retrospective meetings are inadequate for addressing issues that span multiple teams and projects.

Theoretically, this study draws on the agile development literature and is relevant to this thesis for at least two reasons. First, it suggests that agile development enables continuous learning and improvement, which are factors that are crucial for the endurance of value co-creation over time. Second, the study explores agile development practices in a large-scale setting, providing empirical insight into the challenges entailed in achieving value co-creation in settings characterized by complex and interconnected information systems.

Contribution: I contributed to the data collection and data analysis of this paper. In addition, I wrote the case description.

5.3 Paper 3 – Inertia and change in transformation of the IT function in large organizations: A path theory lens

This paper explores path dependence in large-scale development organizations and how such path dependence can be overcome through path-breaking mechanisms. Empirically, the paper is based on a longitudinal case study of the NAV IT department. For many years, NAV struggled with limited progress in terms of digitally transforming itself and digitalizing its services for citizens. The analysis of the paper revealed that changing the IT organization was challenging due to the inertia of its large projects, the outsourcing of its software development, its complex digital infrastructure that made the deployment of new software risky, and a lack of IT and development competence within the organization. Against all odds, however, the organization managed to transform its IT organization, establishing cross-disciplinary teams that delivered software continuously.

Theoretically, the paper draws on path constitution theory. While path constitution theory has been used in various forms and ways in previous studies on information systems (e.g., (Mehrizi et al. 2019; Singh et al. 2015), in this paper, it is primarily used to explain how path trajectories can be broken and the specific mechanisms and practices that are likely to trigger the transformation of paths. In contrast to the work of Sydow et al. (2009), who favour the use of organizational and managerial practices as sources of path-breaking mechanisms, in this paper, path dependency is seen as being sociotechnical in nature in the context of IT organizations. Hence, the mechanisms used for breaking a path, as well as the interactions between them, can be both technical and organizational in nature.

The paper contributes to this thesis in at least two important ways. First, it uses the concept of path dependence to explain how the inertia of existing systems reduces an organization's flexibility, thus reducing its capacity to co-create value. Second, the paper suggests that changes to governance strategies, sourcing strategies, and technology platforms can enable an organization to break away from an existing path.

Contribution: I collected the data, participated in the data analysis, and wrote the case description of this paper. In addition, I proofread the paper and provided comments to the main author.

5.4 Paper 4 – Platformizing the Organization through Decoupling and Recoupling: A Longitudinal Case Study of a Government Agency

This paper investigates how digital platforms can increase digital innovation through the process of platformization. Empirically, the paper reports on a longitudinal case study of NAV. In 2017, NAV began a major platformization of its digital infrastructure. The platformization process involved both technical and organizational changes. By dismantling its legacy systems into platform-oriented infrastructure, the organization was able to establish cross-functional teams that developed and deployed software independently. The implementation of digital platforms thus enabled a shift from staged to continuous development practices, increasing NAV's organizational flexibility and innovation (Fitzgerald and Stol 2017). This ability was further enhanced by the formation of "service domains" (Evans 2004), which consisted of teams and applications that were relatively closely linked. By combining and recombining teams and applications into service domains, the organization assumed a fluid structure where value paths could be dynamically reconfigured in response to the emergent needs of citizens and other stakeholders.

Theoretically, we conceptualize platformization as the unfolding of two interrelated processes, namely, those of "decoupling" and "recoupling"; decoupling refers to the dismantling of legacy systems into platform-oriented infrastructure, and recoupling is defined as the process of recombining knowledge, skills, and software modules into new and improved value paths. These concepts reflect insights from service-dominant logic, which holds that innovation unfolds through the related activities of liquefaction and resource integration (Lusch and Nambisan 2015). From a service-dominant perspective, the decoupling of legacy systems can thus be seen as a process that increases an organization's resource density by "liquifying" its digital resources, while recoupling contributes to the recombination of resources by introducing roles, practices, and processes that are essential to resource integration.

The paper contributes to this thesis in at least three ways. First, it contributes by theorizing how flexibility and resource density are increased through the process of decoupling. Second, it describes how the recoupling of an organization facilitates the recombination of these resources. Third, the paper contributes by theorizing how the processes of decoupling and recoupling interact, thus enabling improved value co-creation over time.

Contribution: I performed the data collection and data analysis of this study and wrote most of the paper. Knut Rolland wrote the abstract and introduction and provided comments and guidance. In addition, Eric Monteiro provided extensive comments throughout the writing of the paper.

5.5 Paper 5 – Using digital platforms to promote a service-oriented logic in public sector organizations: A case study

This paper explores how digital platforms contribute to improved value co-creation in public sector organizations. Empirically, the paper reports on the transformation of the NAV IT department. The organization made radical changes to its sourcing strategy, technology platform, and governance strategy. Its outsourcing strategy was replaced by an insourcing strategy, its monolithic systems were gradually dismantled into more loosely coupled applications, and its staged development practices were replaced by an iterative approach where digital services were developed and maintained by independent teams. The paper suggests that these changes enhanced the organization's ability to co-create value for three reasons. First, they enhanced the organization's ability to collect feedback from large and heterogeneous user groups. Second, they enabled the organization to reintegrate this feedback into subsequent service delivery processes. Third, they ensured that the collection and reintegration of feedback was continuous and ongoing.

In the paper, these changes are conceptualized as a shift from a goods-dominant logic to a service-dominant logic (Vargo and Lusch 2004) that emphasizes value co-creation as being context dependent and continuous. Furthermore, the paper is informed by the public management literature and explores the challenges associated with value co-creation in public sector organizations. Although the public management literature has explored service-dominant logic as an approach for improving public service delivery (Osborne 2018; Osborne et al. 2015; Osborne et al. 2013; Osborne et al. 2016), these studies have not explored the role of technology in achieving a shift from a goods-dominant logic to a service-dominant logic.

The paper is relevant to this thesis for several reasons. First, it argues that digital platforms facilitate shifts from goods-dominant logic to service-dominant logic in public sector organizations. Second, it suggests that digital platforms enable the scaling of value co-creation across time and space.

Contribution: I performed the data collection and data analysis of this study and wrote most of the paper. Marius Mikalsen and Eric Monteiro contributed to the writing of the introduction and the creation of the theoretical framework, and they provided comments. In addition, Torgeir Dingsøyr provided valuable input towards the end of the writing process.

6 Implications

6.1 Implications to theory

In this section, I discuss how this thesis answers the research questions posed in Chapter 1. The findings from the abovementioned studies have several implications for both the information system literature and the software engineering literature.

I begin by discussing the role of agile development in value co-creation and then discuss how the inertia of legacy information systems affects organizations' ability to co-create value. Finally, I discuss how digital platforms contribute to improved value co-creation across time and space.

6.1.1 The role of agile development in value co-creation

Agile software development as value co-creation: In this thesis, I view agile development (Dybå and Dingsøyr 2008) as an expression of service-dominant logic (Lusch and Nambisan 2015), and I hold that agile development provides the roles, practices and processes that underlie value co-creation. Service-dominant logic thus provides a framework for analysing the relationships between development methods, service platforms, and service ecosystems. This perspective aligns with recent trends in the agile literature, where a focus on agile methods at the team level has been extended to the broader organizational level with an emphasis on value creation in the software development process (Dingsøyr and Lassenius 2016).

Although value creation has received some attention from researchers in the software engineering field, the extant literature explores value creation from a business perspective (Boehm 2003; Ramesh et al. 2010), largely overlooking the role of end-users. For instance, Ramesh et al. (2010) discuss how requirements should be prioritized based on expected customer value, while Boehm (2003) introduces the concept of "value-based software engineering". Although they provide valuable insight into the process of eliciting and prioritizing requirements, these studies adopt a business perspective and hold that value is defined and produced by service providers. This thesis thus adds to this literature by exploring value creation from a user perspective and seeing value as being individual in nature and co-created through interactions between public sector organizations and citizens.

To date, few studies have employed service-dominant logic in analyses of agile development. The exceptions that exist either use it to motivate the need for user involvement (Babb and Keith 2012) or as a means for exploring collaboration practices in project settings (Bjerknes and Kautz 2019; Kautz and Bjerknes 2020). This thesis adds to this literature in several important ways. First, it goes beyond merely classifying agile development as value co-creation. By drawing on service-dominant logic, it is able not only to state that agile development contributes to value co-creation but also to use the framework to describe *how* value is co-created through agile development. This description is related to the terms "resource density" and "resource integration" (Lusch and Nambisan 2015), where resource density refers to organizations' capacity to innovate, and resource integration refers to their actual ability to *integrate* or recombine these resources into new and improved value propositions. Agile development thus provides the roles, activities, and processes that underlie resource integration in a software development setting.

Second, the current discussions on value co-creation in the context of software development (Babb and Keith 2012; Bjerknes and Kautz 2019; Kautz and Bjerknes 2020) do not consider the role of technology platforms in improving organizations' capacity to co-create value. This thesis adds to these works by suggesting that for efficient value co-creation to occur, co-creation practices must be supported by appropriate technology platforms. Furthermore, this thesis provides insight into the mutually enabling relationship between technology and work practices, insight which is consistent with the service-dominant framework (Lusch and Nambisan 2015). This thesis thus highlights important and largely overlooked aspects of value co-creation related to the context in which value co-creation occurs and how this context can be improved.

Third, this thesis emphasizes the fact that value co-creation is continuous and ongoing. Babb and Keith (2012) argue that value creation should begin from the inception of a project. I extend this understanding by arguing that value creation should not be limited to the duration of a project but should endure from the inception of an idea until the resulting service is eventually discontinued. This emphasis on value creation as an enduring and ongoing process is consistent with insights from service-dominant logic (Lusch and Nambisan 2015) and studies on continuous software development (Fitzgerald and Stol 2017), which highlight the need for a reorientation from focusing on products (outputs) to focusing on processes (outcomes). Viewing value creation as a continuous and ongoing process in public sector organizations is discussed further in the following section.

A processual perspective on value co-creation: The extant research on value co-creation in public sector organizations discusses the benefits, drivers, and barriers of co-creation in the public sector (Baptista et al. 2019; Voorberg et al. 2015), with an emphasis on co-creation as part of projects' initiation or early design (Voorberg et al. 2015). In this thesis, I complement these studies by exploring the organizational changes undertaken by NAV in achieving value co-creation across large and heterogeneous user groups *throughout* the service delivery cycle.

I address these changes in terms of governance strategy and sourcing strategy. Changes to the sourcing strategy of NAV entailed a transition from outsourcing to insourcing software development. By recruiting software developers and gradually replacing consultants with internal employees, NAV ensured increased continuity in the staffing of projects. While software development had previously been financed through large-scale projects, it became a continuous activity performed by internal employees. The long-term goal of this shift was to move from external financing, where software development relied on infrequent and often unpredictable funding, to more stable funding models. This move provided predictability and continuity, allowing the organization to build the knowledge and skills required to improve its value propositions.

In terms of governance strategy, top-down control was replaced by a bottom-up approach where independent, self-organized teams were responsible for entire service development cycles. By establishing multidisciplinary teams with the skills, knowledge, and authority to solve problems independently, NAV was able to continuously sense and react to the emergent needs of citizens. These findings correspond with insights from the field of service-dominant logic, which suggests that organizations must engage in continuous and ongoing improvements to ensure value co-creation throughout the service delivery cycle (Osborne et al. 2016; Vargo and Lusch 2004).

This these addresses an empirical blind spot in the information systems literature by questioning the applicability of direct user interaction as a means for achieving continuous and ongoing value co-creation across large and heterogeneous user groups (Dingsøyr et al. 2019; Roland 2018). In this way, this thesis complements the existing studies in this field by emphasizing the context-dependent and emergent nature of value co-creation and arguing that public sector organizations need to radically restructure their service delivery models and employ mediated forms of feedback and learning.

Although other studies have addressed the need for responsive service delivery methods in public sector organizations (Mergel 2016; Mergel et al. 2019; Torfing et al. 2016), these studies either do not address the structural changes needed to adopt such approaches (Mergel 2016; Torfing et al. 2016) or view agility and responsiveness as "add-ons" that are applied in selected cases (Mergel et al. 2019). In contrast, the study presented in this thesis sees value co-creation as a set of processes and activities that are applied across departments and organizations, which would radically change the way public sector organizations organize and deliver services.

Bridging the gap between the information systems and software engineering fields: Traditionally, the software engineering field has been dominated by studies on the tools and methods used in the development of software, whereas the predominant concern among information systems researchers has been the dissemination and use of these software systems (Hoda et al. 2018). The study of agile methods (Dybå and Dingsøyr 2008), with its focus on the human and social aspects of software engineering, has brought the fields of software engineering and information systems closer together. I contribute to further closing this gap by applying theories from the information systems field to explore software development practices within organizations. As this thesis shows, novel technologies and development practices remove the boundaries between those who use software and those who make it. In line with this insight, I argue that the research on value co-creation would benefit from a cross-disciplinary approach where insights from the software engineering literature are merged with theories from the information systems field.

6.1.2 The effect of inertia on value co-creation

Constraining flexibility and value co-creation: Public sector organizations tend to develop path dependency related to the interaction and interdependencies between sourcing strategies, technical platforms, and governance strategies. These important aspects of IT organizations tend to become very complementary entities, which makes each entity harder to change because of interdependencies and self-reinforcing mechanisms. As observed in the case study of NAV, adaptive expectation effects led the organization to adopt coordinated release practices across its portfolio, even though these coordination practices proved inefficient for smaller systems and projects. Furthermore, complementary effects related to a historical decision to outsource software development while simultaneously insourcing information systems operations made multidisciplinary teams hard to implement, since experts from the operations department, developers from consulting companies, and business experts were required to work together in teams. Finally, learning and coordination effects reinforced the outsourcing path (Law 2017): After years of outsourcing, the organization had become specialized in procuring software, while it lacked both the skills and the knowledge required for developing information systems inhouse.

In addition, the thesis theorizes that large and interconnected information systems seem to lower the bar for developing path dependency. In this way, organizations with relatively modular and standardized components and information systems are less prone to developing path dependence than organizations with large and interconnected systems. The research also indicates that as technology platforms and software development practices become increasingly entangled, it becomes increasingly difficult to adopt agile development practices; this is a perspective left largely unaddressed in the agile development literature.

Creating a new path: Consistent with a path lens (e.g., (Garud et al. 2010; Singh et al. 2015; Sydow et al. 2009)), this study shows that software development organizations can break away from existing paths and transition towards a service-dominant logic. In conducting the longitudinal study of NAV, we identified three mechanisms that contributed to the breaking of NAV's organizational path. The first mechanism was related to experimentation with agile development practices. Even though DigiSyFO, which piloted agile development practices within NAV, had a limited scope, the successful outcome of the project proved the viability and efficiency of agile development practices within the organization, thus contributing to the delegitimization of the prevailing logic (Mehrizi et al. 2019).

A second path-breaking mechanism was related to the insourcing of software development and maintenance. As its responsibility contracts were replaced by capacity contracts, NAV was increasingly able to respond to the emergent needs of citizens. This was challenging when system development was outsourced, since public procurement legislation, combined with the need for predictability and control, meant that requirement elicitation and co-creation were limited to early stages of service specification and design. To ensure that a project was able to deliver on time and cost, changes had to be formally approved; thus, the organization's ability to respond to emergent needs was limited. By introducing internal ownership, the organization had an increased ability to adapt its development practices and functionality to the emergent needs of citizens and other stakeholders. Development teams received functional requirements and could decide for themselves how to meet them. While sourcing strategies traditionally have been addressed in the software engineering literature, this thesis explores sourcing strategies as part of the shift towards a service-dominant logic.

A third path-breaking mechanism was the decoupling of legacy systems. To reduce complexity and facilitate the recombination of resources, the organization began replacing its monolithic systems with more modular applications, thus facilitating independent contributions from multidisciplinary teams. This aligns with findings showing that relatively loosely coupled infrastructures are less prone to path-dependent behaviour (Hanseth and Lyytinen 2010).

Consistent with the work of (Mehrizi et al. 2019), the examined path-breaking mechanisms were observed to be interdependent and mutually enabling, and the timing and sequence of the mechanisms influenced their likelihood of success or failure. For instance, the formation of multidisciplinary teams preconditioned the insourcing of software development.

6.1.3 Platforms as enablers for value co-creation across time and space

Using digital platforms to scale and scope value co-creation: The extant literature on value co-creation in public sector organizations explores service-dominant logic as an alternative way of theorizing value creation in public sector organizations (Osborne et al. 2013). In this thesis, I complement these studies by investigating the way in which digital platforms enable a shift from a goods-dominant logic to a service-dominant logic within public sector organizations. This enabling effect is related to the way in which digital platforms provide a venue where resource integrators can meet and exchange resources.

The findings suggest that digital platforms play a pivotal role in enabling efficient value co-creation within public sector organizations. The container-based application platform of NAV enabled co-creation in three important ways. First, the modular structure of the platform enabled the formation of independent development teams that could work in relative isolation. As long as the application interfaces remained intact, the development teams could experiment and innovate inside the boundaries of their applications (Gawer 2014). Second, the platform provided indirect and mediated feedback from citizens. By monitoring their applications' use and performance, the development teams were able to continuously capture the reactions of citizens. Third, the platform simplified provisioning and deployment, thereby enabling continuous and ongoing reintegration of feedback into subsequent service deliveries. These insights are consistent with insights from service-dominant logic, which suggests that digital platforms increase both the efficiency and effectiveness of resource exchange (Lusch and Nambisan 2015).

Based on these findings, this thesis further suggests that by enabling mediated feedback and rapid reintegration in subsequent service delivery processes, platforms have the potential to scale co-creation in terms of both time and space. While other studies explore the ways in which digital platforms enable improved communication between citizens and governments within existing structures (de Jong et al. 2019; Zavattaro et al. 2015), this study thereby takes a step further and examines the ways in which platforms might enable the formation of radically new structures and improved forms of service delivery.

Furthermore, this study addresses the relation between the structure of digital infrastructure and an organization's ability to develop and deliver services, suggesting that the transformation of public sector organizations preconditions a transformation of their digital infrastructure; indeed, only by increasing the flexibility of infrastructure can value co-creation be scaled across an organization and feedback be incorporated from large and heterogeneous user groups over prolonged periods of time. In the following sections, I discuss how the flexibility of an organization’s infrastructure can be increased through the process of platformization, thereby enabling increased efficiency in value co-creation.

Platformizing public sector organizations. Platformization is often described as the process of establishing core services and an ecosystem of complementors (Benlian et al. 2018; Bygstad and Hanseth 2018; Cusumano 2010). However, in this thesis, I adopt a broader perspective and draw on the concepts of decoupling and recoupling to theorize how digital platforms enable new ways of organizing and delivering services in an interorganizational setting characterized by monolithic legacy systems and hierarchical structures. My detailed account of the platformization process in NAV uncovers the way in which digital platforms pave the way for a reorganization of service delivery where outsourcing of development activities and staged delivery models are replaced by continuous software development and cross-organizational collaboration.

My theorization shows that decoupling can transform legacy systems into modular platform architecture. In this way, decoupling does not merely result in a platform alongside existing digital infrastructure, as presented in the extant literature (Bygstad and Hanseth 2018; Islind et al. 2016); instead, it transforms existing digital infrastructure into a working platform. My theorizing also shows that platformization, through processes of recoupling, allows for a recombination of knowledge and skills into new organizational forms and practices. This enables an organization to dynamically transform its value streams and continuously produce improved value propositions. Furthermore, platformization requires cyclic interaction between processes of decoupling and recoupling in the sense that decoupling provides increased stability (i.e., a platform core) for new knowledge and skills to emerge, and in turn, recoupling implies increased flexibility and competence for change. Hence, the processes of decoupling and recoupling, although distinct, feed each other cyclically so that if organizations have one without the other, they will not be able to implement platformization. In the following sections, the processes of decoupling and recoupling, as well as the cyclic interaction between the two, are discussed in more detail.

Decoupling legacy systems into platforms: By emphasizing decoupling as a strategy for renewing technology, this thesis contributes to the existing literature on this topic by suggesting that digital platforms provide a means for *replacing* legacy systems, thus addressing issues related to the periphery and the core of innovation. Viewing platformization as a strategy for replacing legacy systems complements existing studies that see platformization either as the process of establishing platforms from scratch (Benlian et al. 2018; Islind et al. 2016) or as the process of masking legacy systems behind programmable application interfaces in an effort to increase their peripheral innovation (Bygstad and Hanseth 2018). This thesis also emphasizes platformization as being emergent and ongoing in nature as opposed to unexpected or abruptly changing.

Decoupling corresponds to Islind et al. (2016)'s description of "platformization", where platformization is used to denote the process of establishing a digital platform. However, whereas Islind et al. describe platformization as the process of establishing a platform *alongside* existing infrastructure, decoupling denotes a process where a platform is established *across* existing infrastructure. Thus, Islind et al. describe a case of *green-field* development, where a platform is established *de neuvo*, whereas the present study presents a case of *brown-field* development, where a platform is tailored to existing infrastructure.

This observation corresponds to the concept of decoupling as described by Benlian et al. (2018), who defines decoupling as unfolding process on the infrastructure and application levels. Benlian et al. (2018) see decoupling as a general trend and a consequence of cloud computing. However, we take a more specific approach and see it as a strategy of technological renewal (Wimelius et al. 2020) that enables the gradual introduction of platform-oriented logic in the context of existing infrastructures. This difference is important since although platform structures have been proven efficient in private sector settings, for instance, through studies of commercial platforms within the private sector (Eaton et al. 2015; Ghazawneh and Henfridsson 2013), public sector organizations with legacy systems and practices find it hard to transition from tightly coupled infrastructures. For such organizations, decoupling provides a strategy for gradually and continuously renewing technology.

At NAV, decoupling progressed in two steps. In the first step, applications were decoupled from their underlying digital resources through the use of virtual servers. Virtualization technologies provided increased flexibility, as resources could be dynamically scaled and scoped up or down (Benlian et al. 2018; Krancher et al. 2018). In the second step, legacy systems were decoupled into smaller applications. The container platform enabled the reuse of third-party services and facilitated the decoupling process through the use of web protocols (i.e., SOAP and REST).

The use of platformization as a means for increased flexibility and innovation in digital infrastructures has also been addressed by Bygstad and Hanseth (2018). In a multilevel study of a large e-health initiative, they examine the way in which layered architecture, where legacy systems are encapsulated in a platform core, creates a "platform-oriented" infrastructure. However, while Bygstad and Hanseth (2018) describe a process where legacy systems are encapsulated and hidden from application developers, this thesis describes a process where legacy systems are gradually replaced. Rather than seeing these differing strategies as separate and opposing, they should be seen as complementary and potentially mutually enabling; indeed, the encapsulation of legacy systems might be an aid, or a preliminary stage, on the way towards renewing and replacing legacy systems.

Although the approach proposed by Bygstad and Hanseth (2018) increases *external* innovation (offering complementary services that enable innovation), it leaves legacy systems mostly unchanged. In this way, the strategy fails to improve the maintainability and evolvability of legacy systems, which was seen as the main goal of the decoupling process that unfolded at NAV. The decoupling of its legacy systems provided NAV with a set of modular components that could be combined and recombined into new products and services (Henfridsson et al. 2018), thus increasing its resource density and probability of innovation (Lusch and Nambisan 2015). The modular structure enabled the reallocation of applications to the most suitable actor in the organization (Normann 2001). In practice, in the context of a large legacy of ageing systems, it might not be realistic or desirable to replace all systems. Therefore, renewal can be achieved through a combined strategy where some systems are hidden while others are replaced.

Recoupling organizations into new organizational forms: In this study, I argue that decoupling legacy systems paves the way for an alternative way of organizing. In the examined case, centralized control was replaced by distributed decision structures where independent development teams were responsible for developing and managing applications. The teams were composed of technology experts and business domain representatives. This enabled innovation through the recombination of skills and knowledge across various departments and subject domains (Brown and Duguid 1991). Thus, I propose that the process of recoupling allows for recombinations of knowledge and skills that enable flexible change in organizations.

Since most of NAV's applications belonged to larger value chains, teams that were responsible for developing functionality and that belonged to the same value chain were combined into service domains. In this way, decoupling the digital infrastructure enabled the formation of a relatively self-contained, self-adjusting system of loosely coupled actors (Lusch and Nambisan 2015), where actors contributing to common value chains were more closely connected than actors contributing to different value chains (Evans 2004). These teams and domains transcended the matrix of the organization, providing a fluid structure where employees could be dynamically combined and recombined in response to emergent needs (Ciborra 1996; Schreyögg and Sydow 2010).

These findings complement the platform literature by highlighting the way in which digital platforms enable the restructuring of organizations through the dynamic recombination of teams and applications. By simultaneously opening the "black boxes" of technology and organization (Zammuto et al. 2007), this thesis moves beyond an interorganizational perspective (Eaton et al. 2015; Yoo et al. 2010) and explores the way in which digital platforms facilitate innovation *within* organizations. According to this intraorganizational view, digital platforms enable the recombination of digital resources

(Henfridsson et al. 2018), knowledge and skills across divisions and departments within organizations, further increasing their potential for innovation.

The examined teams were composed of business and technology experts; thus, they had the skills and knowledge needed to develop and manage applications independently. Using a modular platform where applications could be deployed in isolation, the teams were able to assume responsibility for all the stages of the development process—from the inception of an idea until the related service was eventually discontinued. In this way, the organization was able to move from staged development, where different departments were responsible for different parts of the development process, to a continuous process in which one team was responsible for an entire software development cycle (Fitzgerald and Stol 2017). By shifting from staged development practices to a continuous and network-oriented approach, NAV facilitated innovation across teams (Lusch and Nambisan 2015). These findings confirm certain insights from the software engineering literature, which hold that practices and continuous development enhance innovation through feedback and learning (Fitzgerald and Stol 2017). This thesis complements these insights by exploring the way in which digital platforms can enable continuous development practices in large-scale software development settings. In the examined context, continuous development practices enabled feedback from users to be integrated into future versions of services—thus, an additional stakeholder was added to the recoupling process.

Although the concept of recombination is not new (Henfridsson et al. 2018; Lusch and Nambisan 2015), this research contributes to this idea by highlighting the interrelation between social organization and technical infrastructure while opening the "black boxes" of technology and organization (Zammuto et al. 2007). As teams are formed across existing structures, organizational recoupling is less prone to the knowledge disruption associated with structural recombination (Karim and Kaul 2015). However, at NAV, the transition from centralized to distributed control inferred a considerable shift in the organization's power structures—where decision authority was transferred from centralized and coordinating roles to development teams. Therefore, the transition faced considerable resistance from parts of the organization and required coordinated efforts and persuasion at all its levels. However, these issues are beyond the scope of this research.

Combining decoupling and recoupling: Grounded in this research, this thesis theorizes that the processes of decoupling and recoupling interact cyclically, as decoupling increases the potential for recoupling and vice versa. Specifically, the analysis revealed that the process of decoupling provides new ways of organizing the development of information systems—thus increasing an organization's capacity to recouple. Similarly, the recoupling of an organization produces new organizational capabilities for renewing information systems, which, in turn, facilitates further decoupling.

Consistent with previous research, this study shows that the decoupling of digital infrastructure enables a recombination of services (Benlian et al. 2018) and the introduction of alternative organizational logic (Yoo et al. 2010). This research complements related previous studies by emphasizing platformization as an emergent phenomenon and exploring the way in which hierarchical organizations are gradually and incrementally replaced by distributed models.

NAV began by establishing one domain, and based on these experiences, it continued to establish others. This incremental approach also faced resistance, as a proven track record (where independent development teams outperformed traditional project deliveries in terms of efficiency and flexibility) was a powerful and convincing argument in discussions with sceptics. In this manner, the incremental approach reduced resistance and increased the likelihood of a successful transition. With this incremental approach, the platformization process became a process of continuous organizational improvement, where feedback from one cycle was fed into subsequent cycles, allowing for a gradual, knowledge-based transformation of the organization.

7 Conclusion

The goal of this research was to explore how platformization contributes to value co-creation in public sector organizations. The research goal was detailed in three research questions. To conclude this thesis, I provide summarized answers to each of the questions below.

RQ1: What is the role of agile development in value co-creation?

The main conclusion drawn from the answer to the first research question is that agile development contributes to value co-creation by providing the roles, practices and processes that underlie resource integration. Agile development thus provides iterative and collaborative work practices that enable value co-creation over time.

This thesis contributes to the literature on agile development by theorizing that agile methods are a manifestation of service-dominant logic. The thesis thus places agile practices in a technical and organizational context, suggesting that for a successful transition from service-dominant to goods-dominant logic to occur, organizations must make simultaneous changes to their development methods, organizational structure, and technology platforms.

RQ2: How does inertia affect an organization's ability to co-create value?

The main conclusion drawn from the answer to the second research question is that the inertia of existing systems and practices might reduce an organization's flexibility, thus limiting its capacity to recombine resources and co-create value.

Since public sector organizations are particularly prone to path dependence, it is important to identify self-reinforcing mechanisms within them that might lead to path dependence and find mechanisms that might enable them to break away from path-dependent behaviours should they occur. In this thesis, three interdependent path-breaking mechanisms are proposed. These include the use of digital platforms for software development, the insourcing of software development, and the establishment of multidisciplinary software development teams.

RQ3: How do digital platforms enable value co-creation across time and space?

The main conclusion drawn from the answer to the third research question is that digital platforms enable value co-creation *across time and space* by facilitating the collection and reintegration of feedback from large and homogeneous user groups. The thesis thus suggests that the introduction of digital platforms in public sector organizations enables value co-creation at scale and scope.

Furthermore, the thesis argues that in public sector organizations, digital platforms can be introduced across existing infrastructures through the process of *platformization.* The platformization process is theorized as unfolding through two separate but interconnected processes of *decoupling* and *recoupling*, where decoupling contributes to increased resource density and recoupling denotes the process of establishing the activities, practices, and processes that underlie resource integration. Platformization thus encompasses the technical and organizational changes necessary for organizations to transition from a goods-dominant logic to a service-dominant logic.

The decoupling of legacy systems into more loosely coupled applications enables the formation of multidisciplinary teams, which can facilitate collaboration and innovation across an organization. Teams and applications can further be combined and recombined into service domains, which provides a fluid structure where services are continuously improved in response to the emergent needs of citizens.

7.1 Limitations

The focus of our study has been the broad strategic and technical changes needed to move public sector organizations towards a service-dominant logic. To pursue this goal, I adopted a supply-side focus in the exploration of organizational and technological changes. I thus gave limited attention to the perceptions of citizens in my exploration of the ongoing transformation. The rationale behind this decision is twofold. First, capturing both the supply side *and* the demand side of the complex case of NAV was not possible within the constraints of this thesis. Second, many of NAV's services are part of a larger value chain that includes a wide array of public and private actors. It will therefore take time before the effects of the ongoing transformation propagate to citizens. Although insight into the reactions of users would have added depth and flavour to the study, I argue that given the aim of this study, my supply-side focus provides valuable insight in its own right.

As my theorizing is based on a single case study, it is inevitably subject to some limitations. For example, results based on single case studies are not by default valid in the contexts of other cases (Yin 2013). In addition, my theorizing regarding platformization is restricted to public sector organizations.

A note must also be made about the working traditions in Norway, which favour a democratic and egalitarian approach. It can therefore be assumed that agile development practices are easier to implement in Norway and in countries with similar work traditions than they would be in countries with more hierarchical traditions. Norwegian legislation has required consultation with employees and union representatives in the context of IT projects since 1977 (Affairs 2006). Insight into the transferability of these approaches to other cultures would therefore require additional research.

7.2 Opportunities for further research

This thesis investigates the use of digital platforms as a means for improving value co-creation in public sector organizations. It is by no means exhaustive and leaves room for several streams of future research.

First, it will be a considerable amount of time before the effects of the examined transformation are known. Although the outcome seemed favourable as I completed my research in 2019, further investigation is needed to uncover the long-term effects of the transformation on the organization.

Second, the research was restricted to a certain domain and a certain culture. Exploring the applicability of similar approaches in other public sector or private sector contexts thus presents another opportunity for future research. It would also be of interest to investigate the applicability of such approaches in less egalitarian cultures than that of Norway.

Third, my research lacks details regarding the specific monitoring and feedback mechanisms used in the delivery process. Exploring the different forms of mediated feedback and the way in which they evolve over time presents another opportunity for future research.

Fourth, as noted above, the research presented in this thesis adopts a supply-side perspective. Exploring such transformations from a citizen perspective provides an additional opportunity for further research.

Fifth, transformations such as the one described in this thesis have considerable impacts on the types of roles and competencies needed within an organization. Such a transformation also impacts power structures, as decision authority becomes decentralized. Investigating how changes to technology impact social structures presents another opportunity for research.

8 References

Aanestad, M., Grisot, M., Hanseth, O., and Vassilakopoulou, P. 2017. "Information Infrastructures and the Challenge of the Installed Base," in *Information Infrastructures within European Health Care: Working with the Installed Base*. Springer.

Affairs, M. o. L. a. S. 2006. "Act Relating to Working Environment, Working Hours and Employment Protection, Etc.," M.o.L.a.S. Affairs (ed.). Norway: Lovdata.

Alford, J., and Hughes, O. 2008. "Public Value Pragmatism as the Next Phase of Public Management," *The American Review of Public Administration* (38:2), pp. 130-148.

Arthur, W. B. 1989. "Competing Technologies, Increasing Returns, and Lock-in by Historical Events," *The economic journal* (99:394), pp. 116-131.

Arthur, W. B. 1990. "Positive Feedbacks in the Economy," *Scientific american* (262:2), pp. 92-99.

Arthur, W. B. 1994. *Increasing Returns and Path Dependence in the Economy*. University of michigan Press.

Askim, J., Christensen, T., Fimreite, A. L., and Laegreid, P. 2010. "How to Assess Administrative Reform? Investigating the Adoption and Preliminary Impacts of the Norwegian Welfare Administration Reform," *Public Administration* (88:1), pp. 232-246.

Askim, J., Fimreite, A. L., Moseley, A., and Pedersen, L. H. 2011. "One-Stop Shops for Social Welfare: The Adaptation of an Organizational Form in Three Countries," *Public Administration* (89:4), pp. 1451-1468.

Babb, J. S., and Keith, M. 2012. "Co-Creating Value in Systems Development: A Shift Towards Service-Dominant Logic," *Journal of Information Systems Applied Research* (5:1), p. 4.

Baldwin, C. Y., and Clark, K. B. 2000. *Design Rules: The Power of Modularity*. MIT press.

Baptista, N., Alves, H., and Matos, N. 2019. "Public Sector Organizations and Cocreation with Citizens: A Literature Review on Benefits, Drivers, and Barriers," *Journal of Nonprofit & Public Sector Marketing*), pp. 1-25.

Barroca, L., Sharp, H., Dingsøyr, T., Gregory, P., Taylor, K., and AlQaisi, R. 2019. "Enterprise Agility: A Balancing Act-a Local Government Case Study," *International Conference on Agile Software Development*: Springer, Cham, pp. 207-223.

Battleson, D. A., West, B. C., Kim, J., Ramesh, B., and Robinson, P. S. 2016. "Achieving Dynamic Capabilities with Cloud Computing: An Empirical Investigation," *European Journal of Information Systems* (25:3), pp. 209-230.

Benlian, A., Kettinger, W. J., Sunyaev, A., and Winkler, T. J. 2018. "Special Section: The Transformative Value of Cloud Computing: A Decoupling, Platformization, and Recombination Theoretical Framework," *Journal of Management Information Systems* (35:3), pp. 719-739.

Biernacki, P., and Waldorf, D. 1981. "Snowball Sampling: Problems and Techniques of Chain Referral Sampling," *Sociological methods & research* (10:2), pp. 141-163.

Bjerknes, G., and Kautz, K. 2019. "Agile Development as Service Ecosystems," *Proceedings of the Australasian Conference on Information Systems*.

Boehm, B. 2003. "Value-Based Software Engineering: Reinventing," *SIGSOFT Softw. Eng. Notes* (28:2), p. 3.

Boland Jr, R. J., Lyytinen, K., and Yoo, Y. 2007. "Wakes of Innovation in Project Networks: The Case of Digital 3-D Representations in Architecture, Engineering, and Construction," *Organization science* (18:4), pp. 631-647.

Bonsón, E., Royo, S., and Ratkai, M. 2015. "Citizens' Engagement on Local Governments' Facebook Sites. An Empirical Analysis: The Impact of Different Media and Content Types in Western Europe," *Government information quarterly* (32:1), pp. 52-62.

Bosch, J. 2015. "Speed, Data, and Ecosystems: The Future of Software Engineering," *IEEE Software* (33:1), pp. 82-88.

Bosch-Sijtsema, P., and Bosch, J. 2015. "User Involvement Throughout the Innovation Process in High-Tech Industries," *Journal of Product Innovation Management* (32:5), pp. 793-807.

Brown, A., Fishenden, J., Thompson, M., and Venters, W. 2017. "Appraising the Impact and Role of Platform Models and Government as a Platform (Gaap) in Uk Government Public Service Reform: Towards a Platform Assessment Framework (Paf)," *Government Information Quarterly* (34:2), pp. 167-182.

Brown, J. S., and Duguid, P. 1991. "Organizational Learning and Communities-of-Practice: Toward a Unified View of Working, Learning, and Innovation," *Organization science* (2:1), pp. 40-57.

Bygstad, B., and Hanseth, O. 2018. "Transforming Digital Infrastructures through Platformization,").

Calvin S. Hall, G. L., John B. Campbell. 1997. *Theories of Personality, 4th Edition*, (4th ed.). Wiley.

Cennamo, C., and Santaló, J. 2019. "Generativity Tension and Value Creation in Platform Ecosystems," *Organization Science* (30:3), pp. 617-641.

Ciborra, C. 2000. *From Control to Drift: The Dynamics of Corporate Information Infastructures*. Oxford University Press on Demand.

Ciborra, C. U. 1996. "The Platform Organization: Recombining Strategies, Structures, and Surprises," *Organization science* (7:2), pp. 103-118.

Comission, E. 2017. "Tallinn Declaration on Egovernment at the Ministerial Meeting During Estonian Presidency of the Council of the Eu on 6 October 2017."

Conboy, K. 2009. "Agility from First Principles: Reconstructing the Concept of Agility in Information Systems Development," *Information Systems Research* (20:3), pp. 329-354.

Conboy, K., and Fitzgerald, B. 2004. "Toward a Conceptual Framework of Agile Methods: A Study of Agility in Different Disciplines," *Proceedings of the 2004 ACM workshop on Interdisciplinary software engineering research*, pp. 37-44.

Conway, M. E. 1968. "How Do Committees Invent," *Datamation* (14:4), pp. 28-31.

Cunningham, W. 1992. "The Wycash Portfolio Management System," *ACM SIGPLAN OOPS Messenger* (4:2), pp. 29-30.

Danneels, L., Viaene, S., and Van den Bergh, J. 2017. "Open Data Platforms: Discussing Alternative Knowledge Epistemologies," *Government Information Quarterly* (34:3), pp. 365-378.

David, P. A. 1985. "Clio and the Economics of Qwerty," *The American economic review* (75:2), pp. 332-337.

David, P. A. 1994. "Why Are Institutions the 'Carriers of History'?: Path Dependence and the Evolution of Conventions, Organizations and Institutions," *Structural change and economic dynamics* (5:2), pp. 205-220.

de Jong, M. D., Neulen, S., and Jansma, S. R. 2019. "Citizens' Intentions to Participate in Governmental Co-Creation Initiatives: Comparing Three Co-Creation Configurations," *Government information quarterly* (36:3), pp. 490-500.

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

Dingsøyr, T., and Lassenius, C. 2016. "Emerging Themes in Agile Software Development: Introduction to the Special Section on Continuous Value Delivery," *Information and Software Technology* (77), pp. 56-60.

Dingsøyr, T., Mikalsen, M., Solem, A., and Vestues, K. 2018. "Learning in the Large-an Exploratory Study of Retrospectives in Large-Scale Agile Development," *International Conference on Agile Software Development*: Springer, Cham, pp. 191-198.

Dørum, I. 2017. "En Smidigere Arkitekturprosess - Fra «Hviskeleken» Til Felles ArkitekturforståElse?," in: *Informatics*. University of Oslo.

Dunleavy, P., Margetts, H., Bastow, S., and Tinkler, J. 2006. "New Public Management Is Dead—Long Live Digital-Era Governance," *Journal of public administration research and theory* (16:3), pp. 467-494.

Dybå, T., and Dingsøyr, T. 2008. "Empirical Studies of Agile Software Development: A Systematic Review," *Information and software technology* (50:9), pp. 833-859.

Eaton, B., Elaluf-Calderwood, S., Sorensen, C., and Yoo, Y. 2015. "Distributed Tuning of Boundary Resources: The Case of Apple's Ios Service System," *MIS Quarterly: Management Information Systems* (39:1), pp. 217-243.

Eisenhardt, K. M. 1989. "Building Theories from Case Study Research," *Academy of management review* (14:4), pp. 532-550.

Eisenmann, T., Parker, G., and Van Alstyne, M. W. 2006. "Strategies for Two-Sided Markets," *Harvard business review* (84:10), p. 92.

Evans, E. 2004. *Domain-Driven Design: Tackling Complexity in the Heart of Software*. Addison-Wesley Professional.

Feathers, M. C. 2005. *Working Effectively with Legacy Code*. Safari Tech Books Online.

Feldman, M. S., Bell, J., and Berger, M. T. 2003. *Gaining Access: A Practical and Theoretical Guide for Qualitative Researchers*. Rowman Altamira.

Fishenden, J., and Thompson, M. 2012. "Digital Government, Open Architecture, and Innovation: Why Public Sector It Will Never Be the Same Again," *Journal of public administration research and theory* (23:4), pp. 977-1004.

Fitzgerald, B., and Stol, K.-J. 2017. "Continuous Software Engineering: A Roadmap and Agenda," *Journal of Systems and Software* (123), pp. 176-189.

Fuchs, C., and Hess, T. 2018. "Becoming Agile in the Digital Transformation: The Process of a Large-Scale Agile Transformation,").

Garud, R., Kumaraswamy, A., and Karnøe, P. 2010. "Path Dependence or Path Creation?," *Journal of Management Studies* (47:4), pp. 760-774.

Gawer, A. 2014. "Bridging Differing Perspectives on Technological Platforms: Toward an Integrative Framework," *Research Policy* (43:7), pp. 1239-1249.

Geertz, C. 1973. *The Interpretation of Cultures*. Basic books.

Ghazawneh, A., and Henfridsson, O. 2013. "Balancing Platform Control and External Contribution in Third-Party Development: The Boundary Resources Model," *Information Systems Journal* (23:2), pp. 173-192.

Glaser, B. G., and Strauss, A. L. 1967. *The Discovery of Grounded Theory : Strategies for Qualitative Research*. New York: Aldine de Gruyter.

Government, T. 2004-2005. "St.Prp. Nr. 46 (2004-2005) - Ny Arbeids- Og Velferdsforvaltning," A.-o. sosialdepartementet (ed.). www.stortinget.no.

Hand, L. C., and Ching, B. D. 2020. "Maintaining Neutrality: A Sentiment Analysis of Police Agency Facebook Pages before and after a Fatal Officer-Involved Shooting of a Citizen," *Government Information Quarterly* (37:1), p. 101420.

Hanseth, O., and Lyytinen, K. 2010. "Design Theory for Dynamic Complexity in Information Infrastructures: The Case of Building Internet," *Journal of Information Technology* (25:1), pp. 1-19.

Haugen, G. 2018. "Opptur Med Digital Sykmelding," in: *MEMU*. NAV.

Helmond, A. 2015. "The Platformization of the Web: Making Web Data Platform Ready," *Social Media+ Society* (1:2), p. 2056305115603080.

Henfridsson, O., Nandhakumar, J., Scarbrough, H., and Panourgias, N. 2018. "Recombination in the Open-Ended Value Landscape of Digital Innovation," *Information and Organization* (28:2), pp. 89-100.

Henfridsson, O., Yoo, Y., and Svahn, F. 2009. "Path Creation in Digital Innovation: A Multi-Layered Dialectics Perspective," Association for Information Systems.

Hoda, R., Salleh, N., and Grundy, J. 2018. "The Rise and Evolution of Agile Software Development," *IEEE Software* (35:5), pp. 58-63.

Islind, A. S., Lindroth, T., Snis, U. L., and Sørensen, C. 2016. "Co-Creation and Fine-Tuning of Boundary Resources in Small-Scale Platformization," Cham: Springer International Publishing, pp. 149-162.

Johansen, S., and Skålnes, S. 2014. "Etterevaluering Av Nav Ikt Basis,").

Ju, J., Liu, L., and Feng, Y. 2019. "Public and Private Value in Citizen Participation in E-Governance: Evidence from a Government-Sponsored Green Commuting Platform," *Government Information Quarterly* (36:4), p. 101400.

Kallinikos, J., Aaltonen, A., and Marton, A. 2013. "The Ambivalent Ontology of Digital Artifacts," *Mis Quarterly*), pp. 357-370.

Karim, S., and Kaul, A. 2015. "Structural Recombination and Innovation: Unlocking Intraorganizational Knowledge Synergy through Structural Change," *Organization Science* (26:2), pp. 439-455.

Kassen, M. 2013. "A Promising Phenomenon of Open Data: A Case Study of the Chicago Open Data Project," *Government Information Quarterly* (30:4), pp. 508-513.

Kautz, K. K., and Bjerknes, G. 2020. "Information Systems Development as Value Co-Creation," *Communications of the Association for Information Systems* (46:1), p. 33.

Klausen, J. E. 2016. "Tone Alm Andreassen Og Jacob Aars: Den Store Reformen. Da Nav Ble Til," *Tidsskrift for samfunnsforskning* (56:3), pp. 318-322.

Klein, H. K., and Myers, M. D. 1999. "A Set of Principles for Conducting and Evaluating Interpretive Field Studies in Information Systems," *MIS quarterly*), pp. 67-93.

Kniberg, H. 2015. *Scrum and Xp from the Trenches*. Lulu. com.

Krancher, O., Luther, P., and Jost, M. 2018. "Key Affordances of Platform-as-a-Service: Self-Organization and Continuous Feedback," *Journal of Management Information Systems* (35:3), pp. 776-812.

Kruchten, P., Nord, R. L., and Ozkaya, I. 2012. "Technical Debt: From Metaphor to Theory and Practice," *Ieee software* (29:6), pp. 18-21.

Kua, P. 2013. "The Retrospective Handbook," *E--book available at: https://leanpub. com/the--retrospective--handbook*).

Langley, A. 1999. "Strategies for Theorizing from Process Data," *Academy of Management*).

Law, F. 2017. "Breaking the Outsourcing Path: Backsourcing Process and Outsourcing Lock-In," *European Management Journal* (36:3), pp. 341-352.

Lusch, R. F., and Nambisan, S. 2015. "Service Innovation: A Service-Dominant Logic Perspective," *MIS quarterly* (39:1).

Lusch, R. F., Vargo, S. L., and Tanniru, M. 2010. "Service, Value Networks and Learning," *Journal of the academy of marketing science* (38:1), pp. 19-31.

Mahoney, J. 2000. "Path Dependence in Historical Sociology," *Theory and society* (29:4), pp. 507-548.

Mehrizi, M. H. R., Modol, J. R., and Nezhad, M. Z. 2019. "Intensifying to Cease: Unpacking the Process of Information Systems Discontinuance," *MIS Quarterly* (43:1), pp. 141-165.

Mergel, I. 2016. "Agile Innovation Management in Government: A Research Agenda," *Government Information Quarterly* (33:3), pp. 516-523.

Mergel, I., Edelmann, N., and Haug, N. 2019. "Defining Digital Transformation: Results from Expert Interviews," *Government Information Quarterly* (36:4), p. 101385.

Mergel, I., Ganapati, S., and Whitford, A. B. 2020. "Agile: A New Way of Governing," *Public Administration Review*).

Mergel, I., Kattel, R., Lember, V., and McBride, K. 2018. "Citizen-Oriented Digital Transformation in the Public Sector," *Proceedings of the 19th Annual International Conference on Digital Government Research: Governance in the Data Age*, pp. 1-3.

Meyer, U., and Schubert, C. 2007. "Integrating Path Dependency and Path Creation in a General Understanding of Path Constitution. The Role of Agency and Institutions in the Stabilisation of Technological Innovations," *science, technology & Innovation studies* (3:1), pp. 23-44.

Moore, M. H. 1995. *Creating Public Value: Strategic Management in Government*. Harvard university press.

Nam, C. 2020. "Behind the Interface: Human Moderation for Deliberative Engagement in an Erulemaking Discussion," *Government Information Quarterly* (37:1), p. 101394.

Nam, T. 2012. "Suggesting Frameworks of Citizen-Sourcing Via Government 2.0," *Government Information Quarterly* (29:1), pp. 12-20.

Normann, R. 2001. *Reframing Business: When the Map Changes the Landscape*. John Wiley & Sons.

Normann, R., and Ramirez, R. 1993. "From Value Chain to Value Constellation: Designing Interactive Strategy," *Harvard business review* (71:4), pp. 65-77.

O'Reilly, T. 2011. "Government as a Platform," *Innovations: Technology, Governance, Globalization* (6:1), pp. 13-40.

OECD. 2017. "Digital Government Review of Norway - Boosting the Digital Transformation of the Public Sector," OECD, OECD Publishing, Paris.

Osborne, S. P. 2018. "From Public Service-Dominant Logic to Public Service Logic: Are Public Service Organizations Capable of Co-Production and Value Co-Creation?." Taylor & Francis.

Osborne, S. P., Radnor, Z., Kinder, T., and Vidal, I. 2015. "The Service Framework: A Public-Service-Dominant Approach to Sustainable Public Services," *British Journal of Management* (26:3), pp. 424-438.

Osborne, S. P., Radnor, Z., and Nasi, G. 2013. "A New Theory for Public Service Management? Toward a (Public) Service-Dominant Approach," *The American Review of Public Administration* (43:2), pp. 135-158.

Osborne, S. P., Radnor, Z., and Strokosch, K. 2016. "Co-Production and the Co-Creation of Value in Public Services: A Suitable Case for Treatment?," *Public Management Review* (18:5), pp. 639-653.

Paasivaara, M., Behm, B., Lassenius, C., and Hallikainen, M. 2018. "Large-Scale Agile Transformation at Ericsson: A Case Study," *Empirical Software Engineering*), pp. 1-47.

Pan, S. L., and Tan, B. 2011. "Demystifying Case Research: A Structured–Pragmatic–Situational (Sps) Approach to Conducting Case Studies," *Information and Organization* (21:3), pp. 161-176.

Parker, G. G., Van Alstyne, M. W., and Choudary, S. P. 2016. *Platform Revolution: How Networked Markets Are Transforming the Economyand How to Make Them Work for You*. WW Norton & Company.

Pettigrew, A. M. 1990. "Longitudinal Field Research on Change: Theory and Practice," *Organization science* (1:3), pp. 267-292.

Pierson, P. 2000. "Increasing Returns, Path Dependence, and the Study of Politics," *American political science review* (94), pp. 251-267.

Poell, T., Nieborg, D., and van Dijck, J. 2019. "Platformisation," *Internet Policy Review* (8:4), pp. 1-13.

Ramasubbu, N., and Kemerer, C. F. 2016. "Technical Debt and the Reliability of Enterprise Software Systems: A Competing Risks Analysis," *Management Science* (62:5), pp. 1487-1510.

Ramesh, B., Cao, L., and Baskerville, R. 2010. "Agile Requirements Engineering Practices and Challenges: An Empirical Study," *Information Systems Journal* (20:5), pp. 449-480.

Ribes, D. 2014. "Ethnography of Scaling, or, How to a Fit a National Research Infrastructure in the Room," *Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing*: ACM, pp. 158-170.

Roland, L. K. S., Terje Aksel; Sæbø, Johan Ivar; Eric Monteiro. 2018. "P for Platform," *Scandinavian Journal of Information Systems* (30:2).

Rolland, K.-H., and Vestues, K. 2020. "Inertia and Change in Transformation of the It-Function in Large Organizations: A Path Theory Lens," *Norsk konferanse for organisasjoners bruk at IT*.

Rolland, K. H., Fitzgerald, B., Dingsøyr, T., and Stol, K.-J. 2016. "Problematizing Agile in the Large: Alternative Assumptions for Large-Scale Agile Development,").

Rolland, K. H., Ghinea, G., and Grønli, T.-M. 2015. "Ambidextrous Enterprise Architecting: Betting on the Future and Hacking Path-Dependencies," *ECIS*.

Ruijer, E., Grimmelikhuijsen, S., and Meijer, A. 2017. "Open Data for Democracy: Developing a Theoretical Framework for Open Data Use," *Government Information Quarterly* (34:1), pp. 45-52.

Sanchez, L. M., and Nagi, R. 2001. "A Review of Agile Manufacturing Systems," *International Journal of Production Research* (39:16), pp. 3561-3600.

Schreyögg, G., and Sydow, J. 2010. "Crossroads—Organizing for Fluidity? Dilemmas of New Organizational Forms," *Organization science* (21:6), pp. 1251-1262.

Schreyögg, G., and Sydow, J. 2011. "Organizational Path Dependence: A Process View," *Organization Studies* (32:3), pp. 321-335.

Sharp, H., and Robinson, H. 2010. "Three 'C's of Agile Practice: Collaboration, Co-Ordination and Communication," in *Agile Software Development: Current Research and Future Directions,* T. Dingsøyr, T. Dybå and B.N. Moe (eds.). Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 61-85.

Singh, R., Mathiassen, L., and Mishra, A. 2015. "Organizational Path Constitution in Technological Innovation: Evidence from Rural Telehealth," *Mis Quarterly* (39:3).

Stray, V., Sjøberg, D. I., and Dybå, T. 2016. "The Daily Stand-up Meeting: A Grounded Theory Study," *Journal of Systems and Software* (114), pp. 101-124.

Sydow, J., Schreyögg, G., and Koch, J. 2009. "Organizational Path Dependence: Opening the Black Box," *Academy of management review* (34:4), pp. 689-709.

Tan, B. 1998. "Agile Manufacturing and Management of Variability," *International Transactions in Operational Research* (5:5), pp. 375-388.

Tiwana, A., Konsynski, B., and Bush, A. A. 2010. "Research Commentary—Platform Evolution: Coevolution of Platform Architecture, Governance, and Environmental Dynamics," *Information Systems Research* (21:4), pp. 675-687.

Torfing, J., Sørensen, E., and Røiseland, A. 2016. "Transforming the Public Sector into an Arena for Co-Creation: Barriers, Drivers, Benefits, and Ways Forward," *Administration & Society* (51:5), pp. 795-825.

Törmer, R. L., and Henningsson, S. 2018. "From Drift to Central Guidance: A Path Constitution Perspective on the Platformization of an Information

Infrastructure," *The 26th European Conference on Information Systems (ECIS) 2018European Conference on Information Systems*: Association for Information Systems. AIS Electronic Library (AISeL).

Vågeng, S. 2015. "Et Nav Med Muligheter," Ministry of Labor and Social Affairs, https://www.regjeringen.no/globalassets/departementene/asd/dokumenter/2015/sluttrapport-ekspertgruppen-nav_9.4.15.pdf.

Van de Ven, A. H. J. S. m. j. 1992. "Suggestions for Studying Strategy Process: A Research Note," (13:S1), pp. 169-188.

Van Maanen, J. 1979. "The Fact of Fiction in Organizational Ethnography," *Administrative science quarterly* (24:4), pp. 539-550.

Vargo, S. L., and Lusch, R. F. 2004. "Evolving to a New Dominant Logic for Marketing," *Journal of marketing* (68:1), pp. 1-17.

Vestues, K., Mikalsen, M., and Monteiro, E. 2021. "Using Digital Platforms to Promote a Service-Oriented Logic in Public Sector Organizations: A Case Study," in: *Hawaii International Conference on System Sciences*. Hawaii.

Vestues, K., and Rolland, K.-H. 2021. "Platformizing the Organization through Decoupling and Recoupling: A Longitudinal Case Study of a Government Agency," *Scandinavian Journal of Information Systems*).

Voorberg, W. H., Bekkers, V. J., and Tummers, L. G. 2015. "A Systematic Review of Co-Creation and Co-Production: Embarking on the Social Innovation Journey," *Public Management Review* (17:9), pp. 1333-1357.

Walsham, G. 1995. "Interpretive Case Studies in Is Research: Nature and Method," *European Journal of information systems* (4:2), pp. 74-81.

Williams, L., and Cockburn, A. 2003. "Guest Editors' Introduction: Agile Software Development: It? S About Feedback and Change," *Computer*:6), pp. 39-43.

Wimelius, H., Mathiassen, L., Holmström, J., and Keil, M. 2020. "A Paradoxical Perspective on Technology Renewal in Digital Transformation," *Information Systems Journal*).

Womack, J. P., and Jones, D. T. 1997. "Lean Thinking—Banish Waste and Create Wealth in Your Corporation," *Journal of the Operational Research Society* (48:11), pp. 1148-1148.

Womack, J. P., Jones, D. T., and Roos, D. 2007. *The Machine That Changed the World: The Story of Lean Production--Toyota's Secret Weapon in the Global Car Wars That Is Now Revolutionizing World Industry*. Simon and Schuster.

Yin, R. K. 2003. "Case Study Research Design and Methods Third Edition," *Applied social research methods series* (5).

Yin, R. K. 2013. *Case Study Research: Design and Methods*. Sage publications.

Yoo, Y., Henfridsson, O., and Lyytinen, K. 2010. "Research Commentary—the New Organizing Logic of Digital Innovation: An Agenda for Information Systems Research," *Information systems research* (21:4), pp. 724-735.

Zammuto, R. F., Griffith, T. L., Majchrzak, A., Dougherty, D. J., and Faraj, S. 2007. "Information Technology and the Changing Fabric of Organization," *Organization science* (18:5), pp. 749-762.

Zavattaro, S. M., French, P. E., and Mohanty, S. D. 2015. "A Sentiment Analysis of Us Local Government Tweets: The Connection between Tone and Citizen Involvement," *Government information quarterly* (32:3), pp. 333-341.

Appendix: The papers and co-authorship statements

PAPER I

12-10-2016

Agile requirements work in a digital transformation project: Managing diverse and dispersed user needs

Kathrine Vestues
kathrine.vestues@idi.ntnu.no

Finn Olav Bjornson
bjornson@idi.ntnu.no

Agile requirements work in a digital transformation project: Managing diverse and dispersed user needs

Kathrine Vestues
Norwegian Univ. of Science and Technology
Kathrine.vestues@idi.ntnu.no

Finn Olav Bjørnson
Norwegian Univ. of Science and Technology
bjornson@idi.ntnu.no

ABSTRACT

Successful requirements engineering is vital to the success of software projects. Agile software development seeks to limit the risk of misunderstanding requirements by emphasizing evolutionary delivery and more end-user involvement. But what happens when features are not accepted because the customers cannot agree among themselves? In this paper we report on an ongoing study where a software development company is creating a software system from scratch for a complex, diverse, and dispersed customer organization. We describe our ongoing study in which we follow a feature of the software system from idea to implementation. We attempt to explain our observations through three theoretical lenses: User participation and involvement, power relations in complex organizations, and balancing of local and global needs in system development.

Keywords

Agile development, Agile project management, Agile requirements engineering, User participation

INTRODUCTION

Since the formulation of the agile manifesto in 2001, agile methods have transformed software development practice (Dingsøyr et al. 2012) by emphasizing tolerance for changing requirements, evolutionary delivery and more end-user involvement. According to (Dybå et al. 2014), agile software development represents a new approach for managing software projects, that puts less emphasis on up-front plans and strict control and relies more on informal collaboration, coordination, and learning. In their chapter on agile project management, they provide software project managers with a set of four principles for handling complexity and uncertainty inherent in agile software projects. In this paper we focus on their first principle, minimum critical specification, which deals with how requirements should be elicited and specified. Understanding "the problem" the system is intended to address, they claim, is one of the keys to project success.

In the study described in this paper, we follow the development of a new software tool that is to be used by inspectors in a public, Norwegian directorate. The directorate is responsible for ensuring sustainable use of Norwegian natural assets, and as part of their operation, they perform a large number of onsite inspections to ensure that laws and regualtions are being met. A surveys performed within the directorate revealed that to fulfill this task in a consistent and efficient manner, they needed a new software system. Prior to the development of this system, inspections had been performed using pen, paper and simple text editing tools.

Despite intense focus on user involvement in all stages of development of the new system, the project manager still struggles to get users to approve the system. Key features are deployed, but remain unused. This failure to meet user expectations and deliver usable functionality is both frustrating and costly for the project management.

In this paper we describe our ongoing investigation of the challenges of requirements engineering in this complex customer organizations. To better understand the process of feature development in the project, we follow the development of a single feature from idea to the deployable functionality. This gives us a fruitful perspective when looking at requirements engineering in complex organizations, and seeing how this process affects the organization.

The paper is structured as follows: First, we provide a detailed description of the customer organization, where the main complexity of this case seems to reside. We move on to briefly describe our research design for investigating the case, before providing a theoretical background for our current understanding of the challenges faced by the project. We then move on to discuss the case in light of the different theories in order to find possible explanations for our observations. We conclude by outlining our planned research for following the case further.

CASE

In the case study, we look at the development of a new control and supervision tool that is under development for a public, Norwegian directorate. The directorate is responsible for the management of natural assets, and among its tasks is the supervision and control of how private corporations use these assets. A survey revealed that they needed a new and improved software system to support these supervisory and control operations. The new system is ment to ensure efficient work processes, quality of data and equal treatment of supervised bodies. It will support the work of two distinct user groups with very different requirements and knowledge: (1) Inspectors working in the field and (2) office personnel analyzing inspection data and compiling reports. There existed no system on which the new system could be based.

The directorate has approximately 500 employees, of which 220 work at the head office that is located in the west of Norway. The rest of the employees are distributed across 20 regional offices throughout the country – from Kjøllefjord in the north, to Kristiansund in the south. Each regional office has responsibility for the natural assets in whitin their allotted, geographic region. For instance case processing and operational control takes place at the regional offices, while administration and support is handled by the head office.

The development of the new software system is organized as a project on the customer side, with a dedicated project manager, steering committee, and a group of user representatives known as “product owners”. The development of the system was awarded to an external supplier, with a development team distributed across three Norwegian cities: Trondheim, Oslo and Bergen. The project uses agile software development methodology, working in one month iterations. Each iteration is initiated with a planning session, and ended with a customer demo and retrospective. The product backlog is continuously groomed and reprioritized.

One especially important group of features in the system are checklists for planning and execution of controls and audits. The checklists are seen as vital for ensuring predictability and quality during inspections. As one of the caseworkers at the head office expressed it: “You cannot save a bad inspection with good case management". The functionality was part of the original tender, and was designed and developed in close collaboration with end users. At present, the project has been running for a year, and is estimated to continue for ten more months. The checklist functionality has been completed, but not used. It is currently uncertain if it ever will be taken into use.

RESEARCH DESIGN

The objective of this study is to see how user participation, organizational politics and tension between local and national needs affect the requirements process. To do so we have chosen to use a longitudinal case study. The rich descriptions given by a qualitative study can help us understand, explain and provide guidelines for requirements engineering in projects where user groups are dispersed and diverse.

The case was chosen because of the complexity of its organization: Both the development team and the customer is distributed across multiple locations. In addition, user groups are diverse, and there exists no prior solution on which the new system can be based. It therefore requires extensive negotiation and communication both within the customer organization and between developers and customer representatives to agree on requirements. The case is currently into its second year, and is estimated to continue until June of 2017. We will follow the case until the system has been completed and deployed.

We will explore the case from the view of the development team, observing and interviewing team members. In addition, we will have access to interviews with key members of the customer organization. These interviews will be performed by fellow researchers engaged in a related research project. To narrow down the complexity of the case, we will follow a single unit of functionality from early needs analysis, through discussions, prototyping, development, deployment and use. By looking at the evolution of one requirement, we can deepen our understanding of the requirements process, and be able to better understand the pitfalls and challenges of requirements engineering in complex socio-technical environments.

In our future data analysis we plan to use triangulation to broaden our understanding, and test the validity of the data. More data is needed before this triangulation can be perfomed. We will analyse data from observations, interviews and written artefacts, such as backlogs and memos. As formal interviews are performed, they will be transcribed and analyzed. We have not yet concluded on method of analysis.

Table 9 - Collected data

Artefacts	Description
Project wiki	Contains descriptions of initial requirements. Edited and used by client representatives, as they were elected for development, these requirements were transferred turned into Epics and User stories and handed over to development team
Product backlog	Collection of Epics, User stories and taks that have been elected for development. Registered in project and issue tracking tool
Observations	Observations done of the team at their work stations and in meetings. Only team mebers in Trondheim will be observed in person. Other team members (Oslo and Bergen) will only observed online (video meetings)
Informal interviews with development team	Conversations taking place after meeting, during luch and at coffe machine. The content of these conversations are logged in note book after conversations have taken place. Informal interviews will only be done with team members in Trondheim
Formal interviews with development team	The following interviews will be performed: - Scrum master / project manager (Trondheim) - 2 developers (Trondheim) - Test manager (Trondheim) - Interface designer (Oslo)
Formal interviews with customer	So far 6 interviews have taken place by other researchers. 13 more interviews have been planned. We will have access to use these in our analysis
Official documents	Original tender found on the web. Organizational charts and descriptions of goals and missions of the directorate

At present we have done observations of development team, had informal interviews with the development team, and looked at publicly available documents describing early project requirements. Observations include project meetings internal to the development team, and meetings where the customer has been present over video-link, as well as team members working at their work stations.

REQUIREMENTS ENGINEERING THEORY

Understanding the intended purpose of system, and the needs of its users, is imperative for successful software development. For the development of complex systems with dynamic, non-deterministic and non-linear characteristics, where accurate estimates, stable plans, and predictions are difficult to establish in early stages, many software project will employ agile methodologies. Agile software development methods are a collection of methods that promote teamwork, collaboration, and process adaptability. Some researcher see requirements engineering in agile software development as different from traditional requirements engineering (Bjarnason et al. 2011), and that requirements must be seen as evolving elements in an ecology (Jarke et al. 2011). In agile projects, requirements engineering is an integral part of the development process throughout the project lifecycle. An agile project will often have an initial design phase, but these early requirements are coarsely defined, and will be gradually elaborated throughout the project. Ideally, this will prevent the project from specifying features that will never be developed, and ensure that at any given time, the functionality that gives the most customer value will be prioritized and developed. Research has shown that many challenges faced by traditional requirements engineering will be solved with an agile approach (Inayat et al. 2015), especially relating to project communication and stakeholder involvement. Customer collaboration will continue throughout the project lifecycle, and requirements are added, altered and reprioritized at regular intervals. The use of agile requirements engineering will however introduce some new challenges. Among these are expectations for customer involvement, and customer inability and disagreement, which could clearly be seen in our case study. When choosing vendors, the directorate had agreed that the project would require intense involvement on their part.

Based on a preliminary analysis, it seems useful to look at data through the following theoretical lenses in an attempt to explain observations: (1) User participation and involvement, (2) power relations in complex customer organizations, and (3) the balancing of local and global needs in system development.

User participation

Most software development projects have some degree of user participation. Participation is used to avoid user resistance, gain user commitment and ensure that user requirements are met. These ideas of empowerment and democratization are central to agile software development methodologies such as Scrum and XP, where customer involvement is central to requirements elicitation and prioritization. Users are involved through a range of practices, such as constant planning, extreme prioritization, prototyping and test-driven development (Cao and Ramesh 2008). Most studies show a positive correlation between user participation and successful software design (Abelein and Paech 2015), but there still exist many example of projects that have failed in spite of user participation (Cavaye 1995). One has to pay careful attention to how and when users are involved. Due to time constraints, users often find it difficult to engage themselves until they are forced to use the new system. Management often fails to dedicate time for users to test and give feedback on prototypes and test installations. In our case study, developers complain that users are unable to answer questions and attend meetings due to operational tasks. This in spite of promises to devote adequate time to the project. Apart from the project manager, who is dedicated full time to the project, customer resources are obliged to maintain their usual responsibilities. The Directorate has to fulfill its responsibility to the public regardless of the new system development. Techniques such as prototyping and sketching might be useful tools for involving users, but require high users engaged. If users are not sufficiently engaged, prototyping will only give an illusion of involvement. User´s reaction will await final release of the software, leaving developers baffled and frustrated when they find users unwilling to accept the software. In our case study, failure to meet customer requirements was a recurring headache: Even after rounds of prototypes and demonstrations, customers were displeased.

Requirements engineering work will be especially challenging when the new system does not replace or build on an existing system: "*In a new development project, developers have a poor understanding of product requirements, and are often unable to make wise product decisions because of requirements ambiguity. Ideally, they would benefit significantly from user inputs in these projects. However, given the novelty of the technology involved and their uncertainty about "what the end product should look like", users too are likely to find it difficult to express their needs and suggest design guidelines. In these projects, project managers and leaders can play a strong role in clarifying the expectations of the customers/users as well as the software development firm*" (Subramanyam et al. 2010). This too could be seen in the case study. There was no existing software on which to base the new solution. In addition, the complex organization with its diverse and dispersed needs, made it difficult for project managers and leaders to give clear advice and make conclusive decisions.

Power relations in complex customer organizations

In their paper on understanding the political ecology of requirements engineering, Bergman et al. (Bergman et al. 2002b) addresses the political nature of requirements, and argues that requirements engineering practice and theory must become more engaged with this issue. They describe requirements as emerging from a set of solution spaces depending on the different problem spaces of the involved principals in the organization. Navigating and synthesizing the heterogeneous technical, social, economic and institutional factors are a key to successful requirements engineering. Further, they argue that the traditional functional ecology of requirements engineering is dependent on goal congruence among the stakeholders and if this is not the case, a political ecology of requirements engineering is more suited to establish valid requirements.

Following the track on politics in requirements engineering, a new research agenda for power and politics in requirements engineering was proposed (Milne and Maiden 2011). It is argue that given the increased complexity, uncertainty and organizational embeddedness faced by requirements engineering in practice, power and politics have become increasingly relevant factors that have not been given adequate consideration.

Balancing local and global needs

Much has been written about the need to adapt software systems to the local context, as the local context always will be unique with its local practices and existing infrastructure resources. A failure to recognize this need will prevent employees from performing their work tasks in a suitable and efficient manner. It has, however, been demonstrated that in global information infrastructure systems, a balance must be found between the local and global needs (Rolland and Monteiro 2002). The larger organization wants standardization across sites, while local inspectors require autonomy and flexibility in the face of local variations. Although the organization only operates within Norwegian boarders, the large geographic and cultural difference between different regional offices and between regional offices and the head office makes analogy with global versus local needs relevant and suitable.

RESULTS AND DISCUSSION

Initial observations and informal interviews with members of the development team show that important functionality fails to meet customer expectations. Checklists, which are seen as central features in the new control and supervisory tool, have still not been taken into use, months after deployment. Rework and delays result in increasing costs and widespread frustration. Although recognizing the need for close collaboration with developers, the directorate fails to follow up on their good intentions. As the project proceeds, the customer organization is absorbed by daily chores, failing to give the project sufficient attention. Developers are often left to wait for feedback on important clarifications, slowing down progress and causing widespread frustration. The organization is prone to internal disagreements, and are unable to conclude on controversial issues. According to developers, decisions they think final are often refuted and reemerged during customer demonstrations and retrospectives, leaving the development team ill at ease. To complicate the situation further is the fact that there existed large local variations between regions and types of inspections. Satisfying the needs of one user group seems to impede the work of others. Additional requirements come from caseworkers located at the head office, who have different work practices and information needs compared to inspectors.

In our search for explanations to these project shortcomings, we look in different theoretical directions. One of these is the participation of users in system design. Following an agile methodology, users are involved in every stage of the project, and they make extensive use of collaborative design techniques such as prototyping and sketching. Still, users are dissatisfied with system functionality once deployed. This failure to meet expectations causes frustration, delays and cost overruns. A key reason for this behavior seems to be time pressure on the part of the customer representatives. They are not alleviated of their regular responsibilities, and operational tasks have to be prioritized, leaving the development team to wait. Developers spend a substantial amount of time waiting for customer feedback and clarifications, showing that if user involvement is to be efficient, users must be at least partly dedicated to the project. Otherwise, feedback will come as functionality is deployed, undermining the agile mindset.

Another part of the explanation for the project´s inability to develop adequate features may be the complexity of the customer organization. It seems that both the customer side project manager and the supplier has underestimated the role of intra-organizational politics. Power structures within a complex organization can hinder efficient system development (Bergman et al. 2002b; Milne and Maiden 2011). Drawing on the theory of Bergman et al. the constant rounds of renegotiation of requirements can be seen as a sign of political ecology problems, since they have a tendency to manifest as repeated rounds of systems change orders that are sometimes described as requirements creep (Bergman et al. 2002a). Apparent agreement is reached by defining the requirements too ambiguous, and problems then arise by placing undue influence on system designers who are working at a concrete level and have to make decisions on the ambiguous features. Bergman et al. describes the consequence of this eventuality as important stakeholders losing control, and that this may lead to them emerging later a opponents to the current system because their needs are not being met. Although the theory on political ecology was developed with large-scale requirements analysis in mind, we would like to argue that the theory is also valid in a smaller setting. It would also be interesting to see how the model can be applied in an agile setting. Data from the case study suggest that internal politics and power struggles have affected the project. Further analysis is required to find the degree to which this affected the design of checklist functionality.

It is also possible the project´s apparent inability to meet customer expectations is due to the imbalance between local and global needs (Rolland and Monteiro 2002). There seems to be a conflict between the desire to standardize work processes, and a need for local adaptations in the field. Inspectors are dispersed across large geographic areas, and are responsible for a multitude of facilities with differing information requirements. Case workers at the main office, on the other hand, require elaborate and consistent data, enabling them to make sound and predictable decisions. Facilities that are supervised and fail to meet laws and regulations can be subject to prosecution or injunction. The directorate therefore strives to be unified and fair in the way supervisions are performed.

LIMITATIONS OF THE WORK

The case study describes software development in a public, Norwegian organization. This means that Norwegian laws on public procurement policies strongly effect the acquisition and development process. In addition, the research is still at an early stage, and a more data collection and analysis is needed for final conclusions to be made.

CONCLUSIONS AND FUTURE WORK

Returning to the principle of "minimum critical specification" in agile project management, our case demonstrates how requirements can be misaligned, interpreted differently, and ultimately not be accepted, despite the best intentions of both customer and developers, and following agile practices to the letter. Our preliminary observations and analysis points to the complex structure of the customer as the source of the challenge, and several theories explains different aspects of this. What is clear, however, is that there is little support in the agile literature and methods to deal with these kinds of challenges.

By following the case to its conclusion we aim to contribute to the theoretical framework surrounding requirements engineering with a complex customer structure in an agile context. We also hope this research will produce practical advice for agile project managers dealing with complex customer structures.

REFERENCES

Abelein, U., and Paech, B. 2015. "Understanding the Influence of User Participation and Involvement on System Success–a Systematic Mapping Study," *Empirical Software Engineering* (20:1), pp. 28-81.

Bergman, M., King, J. L., and Lyytinen, K. 2002a. "Large-Scale Requirements Analysis as Heterogeneous Engineering," *Social Thinking-Software Practice*, pp. 357-386.

Bergman, M., King, J. L., and Lyytinen, K. 2002b. "Large-Scale Requirements Analysis Revisited: The Need for Understanding the Political Ecology of Requirements Engineering," *Requirements Engineering* (7:3), pp. 152-171.

Bjarnason, E., Wnuk, K., and Regnell, B. 2011. "A Case Study on Benefits and Side-Effects of Agile Practices in Large-Scale Requirements Engineering," *Proceedings of the 1st Workshop on Agile Requirements Engineering*: ACM, p. 3.

Cao, L., and Ramesh, B. 2008. "Agile Requirements Engineering Practices: An Empirical Study," *IEEE software* (25:1), pp. 60-67.

Cavaye, A. L. M. 1995. "User Participation in System Development Revisited," *Information & Management* (28:5), pp. 311-323.

Dingsøyr, T., Nerur, S., Balijepally, V., and Moe, N. B. 2012. "A Decade of Agile Methodologies: Towards Explaining Agile Software Development," *Journal of Systems and Software* (85:6), pp. 1213-1221.

Dybå, T., Dingsøyr, T., and Moe, N. B. 2014. "Agile Project Management," in *Software Project Management in a Changing World*. Springer, pp. 277-300.

Inayat, I., Salim, S. S., Marczak, S., Daneva, M., and Shamshirband, S. 2015. "A Systematic Literature Review on Agile Requirements Engineering Practices and Challenges," *Computers in human behavior* (51), pp. 915-929.

Jarke, M., Loucopoulos, P., Lyytinen, K., Mylopoulos, J., and Robinson, W. 2011. "The Brave New World of Design Requirements," *Information Systems* (36:7), pp. 992-1008.

Milne, A., and Maiden, N. 2011. "Power and Politics in Requirements Engineering: A Proposed Research Agenda," *2011 IEEE 19th International Requirements Engineering Conference*: IEEE, pp. 187-196.

Rolland, K. H., and Monteiro, E. 2002. "Balancing the Local and the Global in Infrastructural Information Systems," *The information society* (18:2), pp. 87-100.

Subramanyam, R., Weisstein, F. L., and Krishnan, M. S. 2010. "User Participation in Software Development Projects," *Communications of the ACM* (53:3), pp. 137-141.

PAPER II

Learning in the Large - An Exploratory Study of Retrospectives in Large-Scale Agile Development

Torgeir Dingsøyr[1,3](✉), Marius Mikalsen[1], Anniken Solem[2], and Kathrine Vestues[3]

[1] Department of Software Engineering, Safety and Security, SINTEF, 7465 Trondheim, Norway
{torgeir.dingsoyr,marius.mikaelsen}@sintef.no
[2] SINTEF Technology and Society, SINTEF, 7465 Trondheim, Norway
anniken.solem@sintef.no
[3] Department of Computer and Information Science, Norwegian University of Science and Technology, Trondheim, Norway
kathrine.vestues@ntnu.no

Abstract. Many see retrospectives as the most important practice of agile software development. Previous studies of retrospectives have focused on process and outcome at team level. In this article, we study how a large-scale agile development project uses retrospectives through an analysis of retrospective reports identifying a total of 109 issues and 36 action items as a part of a longitudinal case study. We find that most of the issues identified relate to team-level learning and improvement, and discuss these findings in relation to current advice to improve learning outcome in large-scale agile development.

Keywords: Agile software development · Software engineering Teamwork · Team performance · Post mortem review · Reflection Learning · Process improvement

1 Introduction

Retrospective meetings are extremely important in agile methods. The Agile Practice Guide describes them as "*the single most important practice in agile development*" [1] and in his much read book on Scrum and XP, Kniberg states that retrospectives are the "*number-one-most-important thing in Scrum*" [2]. According to the 11th State of Agile report [3], the retrospective is the third most used agile practice. We find many suggestions on how to conduct retrospectives in the agile practitioner literature such as [4, 5] and online[1].

The purpose of retrospectives is to explore the work results of a team in an iteration or a phase in order to "*learn about, improve, and adapt its process*" [1]. The advice offered in the agile community has mainly focused on learning and improvement for

[1] See for example https://plans-for-retrospectives.com/en/?id=32-64-113-13-67 and http://www.funretrospectives.com/ and https://labs.spotify.com/2017/12/15/spotify-retro-kit/.

J. Garbajosa et al. (Eds.): XP 2018, LNBIP 314, pp. 191–198, 2018.
https://doi.org/10.1007/978-3-319-91602-6_13

the team, while such practices also have a potential to provide learning both on the individual level and for a larger project or organization.

In this article, we focus on the practice of learning and improving through retrospectives in large-scale agile development. The research agenda for large-scale agile development has identified knowledge-sharing as an important topic [6]. This is a particularly challenging area of work, as such projects consists of several development teams with dependencies between teams and typically involve complex integration with existing ICT systems in projects that are critical for companies or societies [7].

We structure this article as follows: First, we provide background on studies of retrospective practices and prior studies on analysing content and effect of retrospectives and formulate research questions. Second, we present an exploratory case study and method for analysis of retrospectives. Third, we discuss what the retrospectives have addressed and what could be done to improve the learning outcome of retrospectives in the large and suggest directions for future research.

2 Background

Given the importance of retrospectives in agile development, the topic has received relatively little attention in scientific studies. A review of previous studies on IT retrospectives finds a multitude of definitions of retrospectives, descriptions of a number of outcomes, different practices described, and "*no project retrospective measurements given to confirm […] whether outcomes have been successfully achieved*" [8].

Kniberg [2] describes a retrospective practice as a team exercise lasting 1–3 h where a team identifies what has been «good», what «could have been better» and «improvements», and suggest voting on the improvements to focus on in the next iteration. The practices described in the research literature [8] typically involve additional steps, for example including root cause analysis in order to analyse topics identified before deciding on action items to include in the next iteration.

In a study of retrospective practices at team level, Lethinen et al. [9] found that most discussions were related to topics close to and controllable by the team, but that topics that could not be resolved at the team level due to their complexity nevertheless recurred over time.

Many previous studies have seen retrospectives as an arena for reflection to enable learning and process improvement [10]. Andryiani et al. [11] studied retrospectives with a framework describing stages of reflection as reporting, responding, relating, reasoning and reconstructing. A finding is that agile teams may not achieve all levels of reflection simply by performing retrospective meetings. The study found that "*important aspects discussed in retrospective meetings include identifying and discussing obstacles, discussing feelings, analysing previous action points, identifying background reasons, identifying future action points and generating a plan*" [11].

We have not been able to identify studies of retrospectives in large-scale development, but a blog post describes how Spotify conducted large-scale retrospectives[2]

[2] https://labs.spotify.com/2015/11/05/large-scale-retros/.

in «one big room» in a format similar to world café [12]. Advice in one of the large-scale development frameworks, Large-Scale Scrum (LeSS),[3] is to hold an «overall retrospective» after team retrospectives, to discuss cross-team and system-wide issues, and to create improvement experiments.

We do not know of studies investigating the learning effect of retrospectives, but a summary of relevant theories of learning such as Argyris and Schön's theory of learning and Wenger's communities of practice can be found in one overview article [13], which discusses learning on individual-, team-, and organizational level. Argyris and Schön distinguish between smaller improvement («single loop learning») and more thorough learning («double loop learning»).

In this explorative study, we ask the following research questions: *1. How are retrospectives used in a large-scale agile development project? 2. What could be done to improve the learning outcome of retrospectives in large-scale agile projects?*

3 Method

We are currently conducting a longitudinal case study [14] of a large-scale development project. The case was selected as it is one of the largest development projects in Scandinavia, and is operating in a complex environment with heavy integration with other ICT systems.

The customer organization has 19 000 employees, and close to 300 ICT systems. A new solution will require changes to 26 other systems. The project uses a stage-gate delivery model with 4 stages (analysis of needs, solution description, construction, and approval, similar to a previous project described in [15]). We followed the first release, with 37 developers in four development teams. Teams had a Scrum-master, one or two application architects, one or two testers, and up to ten developers. The project uses the Scrum-method, with three-week iterations, starting with a planning meeting and ending with a demo and retrospective.

The project has three main releases, and this article is based on an analysis of minutes of meetings from 10 retrospectives in the first release. The minutes include iterations 3 to 9, with an exception of iteration 6, when no retrospective was held due to summer holidays. The minutes cover a 5-month period.

We have limited the study to an analysis of retrospective minutes from two of the four teams. The minutes describe who were present in the face-to-face meeting, a list of issues that went well, a list of issues that could be improved and most often a list of action items. The length of the minutes varied from half a page to two pages. The minutes were posted in the project wiki.

We all read three minutes individually, and then jointly established a set of categories, taken from the Scrum guide[4], which describes the purpose of the sprint retrospective as an arena for inspecting how the last sprint went with regards to the categories «people», «relationships» (merged with people), «process», and «tools».

[3] https://less.works/less/framework/index.html.

[4] http://www.scrumguides.org/.

We added the categories «project» and «other teams» to specifically address the large-scale level. These categories were used to code issues and action items.

4 Results

The analysis of minutes from retrospectives in Table 1 shows the issues recorded by the teams during the seven iterations. Most issues were related to «process» (41) and «people and relationships» (30). In the following, we describe issues that emerged in selected categories, and then present the resulting action items as recorded in the minutes.

Table 1. Issues that went well and issues that could be better. In total 109 issues were recorded during seven iterations for two teams. Roughly 40% of issues were statements on issues that went well and 60% about issues that could be improved.

	Iteration 3	Iteration 4	Iteration 5	Iteration 6	Iteration 8	Iteration 9	Iteration 10	Sum
Process	7	3	7	8	5	7	4	41
People & relationships	1	1	13	5	4	6	0	30
Other topics	1	2	3	0	2	2	2	12
Project	0	0	4	0	1	3	2	10
Tools	1	0	3	1	1	1	2	9
Other teams	2	0	1	2	2	0	0	7

Due to space limitations, the following results describe the issues we found relating to the categories that shed most light on how large-scale agile development influences the teams. These are «process» (41 reported issues), «project» (10 reported issues) and «other teams» (7 reported issues).

In terms of *process*, there were issues such as that the build breaks too often, design takes too much capacity from the team, that they would like more consistent use of branching in Git (tool for version control and code sharing), and that frequent check-ins makes it difficult to terminate feature-branches. The following excerpt illustrates how process issues manifest: "*A lot of red in Jenkins [a continuous integration tool], which makes it difficult to branch from «develop»*". Other issues were concerned with quality control and routines in the team, such as the need for better control and routines for branching of the code, need for more code reviews, too many and messy Jira (issue tracker) tasks, and architects have limited time to follow up on development. Issues concerning lack of structure for bug reporting were reported as such: "*Structure concerning tests, bugs are reported in all possible ways – Mails – Skype – face to face, very difficult to follow up and have continuity in test/bugfix etc.*"

Project issues are related to the overall organisation of the project as a whole. Such issues were far less frequently reported, and those we found included having updated requirements for user stories when entering sprints, that solutions designs should be more detailed, product backlog elements should be ready before sprint start, and addressing how developers move between teams. The following illustrates how one team reports the need for more technical meetings between teams on a project level: "*Review of code/project, all meetings are about organisation, but it should be one meeting about how our code/setup/project looks from a technical perspective*".

Finally, for the category *other teams*, i.e. how teams interact in a multi-team setting, we found how there were issues with regard to how teams "takes instructions" from several different parties, and how there was challenges in detecting dependencies in the code before you develop and test. The following excerpt from the retrospective minutes illustrates how one team is not involved sufficiently in the planning of refactoring: "*We want to be notified in advance when there are big refactorings or other significant changes in the codebase, before it happens*".

The retrospective minutes also contains actions decided on by the teams. In total, the two teams identified 36 action items, where most were related to «process» and to «other topics». We show the distribution and provide examples of action items in Table 2.

Table 2. Action items from retrospective minutes according to topic.

Topic	Number	Example action items
Process	13	"Review and assign quality assurance tasks during daily stand-up."
Other topics	7	"We need a course on the «react» technology."
Tools	5	"More memory on the application development image."
People and relationships	5	"Organise an introduction round for new team members."
Project	4	"Have backlog items ready before an iteration starts."
Other teams	2	"Be more aware of dependencies when assigning tasks, make sure that other teams we depend on really give priority to these tasks."

5 Discussion

We return to discuss our two research questions, starting with *how are retrospectives used in a large-scale agile development project?*

We found that retrospectives were used at team level, where short meetings were facilitated by the scrum master and reported in minutes on the project wiki. Minutes were available to everyone in the project, including customer representatives.

Our analysis of topics addressed in the retrospectives shows that most of the issues identified as either «working well» or «could be improved» related to *process*, followed by *people and relationships*. In the «large-scale» categories *project* and *other teams* we found in total 17 issues of the total 109. However, as shown in the results, many of the issues described as *process* were related to the scale of the project, such as identifying challenges with the merging of code or detailing of specifications before development would start. We find, however, that teams mainly deal with team-internal issues in retrospectives.

The analysis of the action items shows that 6 of the 36 action items identified during the work on the release were in the «large-scale» categories. However, we see that some of the action items in the other categories are related to scale. One example is the item "organizing an introduction round for new team members" in the category *people and relations*, which describes an action item which would not be necessary on a single-team project. However, our impression is also here that most action items concern issues at the team level.

We have not been able to conduct an analysis of the effect of retrospectives at team level. We consider that such meetings give room to develop a common understanding of development process, tasks and what knowledge people in the team possess, what in organizational psychology is referred to as *shared mental models* [13] and have been shown to relate to team performance. A common critique of retrospectives is that teams meet and talk, but little of what is talked about is acted upon. We have not been able to assess how many of the 36 identified action items were acted upon, but found in one minute that "all action items suggested to the project management has been implemented". The 36 action items identified can be considered small improvement actions. Given the short time spent on retrospectives, they do not seem to facilitate «deep» learning («double loop» learning in Argyris and Schön's framework). Having minutes public could also lead to critique being toned down or removed completely.

This leads us to discussing our second research question - w*hat could be done to improve the learning outcome of retrospectives in large-scale agile projects?*

In the background we pinpointed particular challenges of large-scale agile development such as dealing with a high number of people and many dependencies [7]. A retrospective can be used for a number of purposes. Prior studies in organizational psychology suggest that in projects with many teams, the coordination between teams are more important than coordination within teams [16]. It is reason to believe it would be beneficial to focus attention on inter-team issues in large projects. The LeSS framework suggests organizing inter-team retrospectives directly after the team retrospectives. Alternatively, teams can be encouraged to particularly focus on inter-team issues as part of the team retrospectives. A challenge in the project studied is that the contract model used may hinder changes, for example the contract model specifies handover phases between companies involved in the *analysis of needs* phase and the *solution description* and *development phase*. However, given the limitations, it is important that the project adjusts work practice also on inter-team level to optimize use of limited resources.

This exploratory study has several limitations, where one is that we have only analysed minutes available on the project wiki from two of four teams.

6 Conclusion

Many in the agile community regard retrospectives as the single most important practice in agile development. It is therefore interesting to know more about how retrospectives are practiced in large-scale development where there is a dire need to learn and improve as many participants are new to the project, the customer organization, and to the development domain. We found that short retrospectives were conducted at team level and mostly addressed issues at the team-level. The action items mainly addressed team level issues. Most actions also seem to relate to smaller improvements, what Argyris and Schön call «single-loop learning».

A large-scale project will benefit from learning and improvement on the project level, and this would be strengthened by following the advice from LeSS by facilitating retrospectives at the project level. Further, to shift learning effects towards «double-loop learning», we suggest that more time is devoted to the retrospectives.

In the future, we would like to initiate retrospectives at the inter-team level, explore the types of issues that are raised, and also gain more knowledge about perceptions of retrospectives by interviewing project participants.

Acknowledgement. This work was conducted in the project Agile 2.0 supported by the Research Council of Norway through grant 236759 and by the companies Kantega, Kongsberg Defence & Aerospace, Sopra Steria, Statoil and Sticos.

References

1. Project Management Institute and Agile Allience: Agile Practice Guide: Project Management Institute (2017)
2. Kniberg, H.: Scrum and XP from the Trenches, 2nd edn. InfoQ (2015)
3. Version One: 11th State of Agile Report (2016)
4. Derby, E., Larsen, D.: Agile Retrospectives: Making Good Teams Great. The Pragmatic Bookshelf (2006)
5. Kua, P.: The Retrospective Handbook (2013). E-book available at: https://leanpub.com/the-retrospective-handbook
6. Moe, N.B., Dingsøyr, T.: Emerging research themes and updated research agenda for large-scale agile development: a summary of the 5th international workshop at XP 2017. Presented at the Proceedings of the XP 2017 Scientific Workshops, Cologne, Germany (2017)
7. Rolland, K.H., Fitzgerald, B., Dingsøyr, T., Stol, K.-J.: Problematizing agile in the large: alternative assumptions for large-scale agile development. In: International Conference on Information Systems, Dublin, Ireland (2016)
8. Skinner, R., Land, L., Chin, W., Nelson, R.R.: Reviewing the Past for a Better Future: Reevaluating the IT Project Retrospective (2015)
9. Lehtinen, T.O., Itkonen, J., Lassenius, C.: Recurring opinions or productive improvements—what agile teams actually discuss in retrospectives. Empirical Softw. Eng. **22**, 2409–2452 (2017)
10. Babb, J., Hoda, R., Norbjerg, J.: Embedding reflection and learning into agile software development. IEEE Softw. **31**, 51–57 (2014)
11. Andriyani, Y.: knowledge management and reflective practice in daily stand-up and retrospective meetings. In: Baumeister, H., Lichter, H., Riebisch, M. (eds.) XP 2017. LNBIP, vol. 283, pp. 285–291. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-57633-6_21
12. Brown, J., Isaacs, D.: The world cafe: Shaping our futures through conversations that matter. Berrett-Koehler Publishers Inc., San Francisco (2005)
13. Dingsøyr, T.: Postmortem reviews: purpose and approaches in software engineering. Inf. Softw. Technol. **47**, 293–303 (2005)

14. Runeson, P., Höst, M.: Guidelines for conducting and reporting case study research in software engineering. Empirical Softw. Eng. **14**, 131–164 (2009)
15. Dingsøyr, T., Moe, N.B., Fægri, T.E., Seim, E.A.: Exploring software development at the very large-scale: a revelatory case study and research agenda for agile method adaptation. Empirical Softw. Eng. **22**, 1–31 (2017)
16. Marks, M.A., Dechurch, L.A., Mathieu, J.E., Panzer, F.J., Alonso, A.: Teamwork in multiteam systems. J. Appl. Psychol. **90**, 964 (2005)

PAPER III

INERTIA AND CHANGE IN TRANSFORMATION OF THE IT-FUNCTION IN LARGE ORGANIZATIONS: A PATH THEORY LENS

Knut H. Rolland, University of Oslo and Kristiania University College

Kathrine Vestues, NTNU

Abstract:

Digitalization and digital transformation of large incumbent organizations, more often than not, require substantial changes to how the IT-function conducts and sources software development. This, however, often involves battling strong organizational and technical inertia making radical changes hard to implement. Yet, radical changes are often a prerequisite for digitally transforming the firm. In this regard, we present a longitudinal case study of a large governmental organization that over a relatively short period managed to radically change their IT-function in spite of strong inertia related to its software development methods and practices, sourcing arrangements, and legacy systems. In explaining this, the paper presents two contributions. First, we build on path theory to explain how IT organizations can develop organizational inertia in terms of path dependence related to software development methods, souring models, software delivery routines, and legacy systems. Second, we identify three interdependent mechanisms of breaking out of such path dependency: use of digital platforms for software development, attracting and securing competence, and establishment of cross-disciplinary software development teams. The paper concludes by giving some directions to further Information Systems (IS) research on this topic.

1. INTRODUCTION

In both practitioner-oriented and research it is often emphasized that new digital technologies and solutions provide organizations with a substantial potential for transformation. Especially technologies like Internet of Things, Big data analytics, digital platforms, machine learning, and combinations of these are typically expected to have transformational effects. Recent IS literature on digital transformation confirms that under certain conditions when organizations maintain a capability to change and utilize such technologies in their particular contexts, they may produce novel solutions and innovation (Jonsson et al., 2018; Svahn et al., 2017). In fact, much literature reflects a fundamental assumption that digital technologies are different in the sense that they are more malleable and evolvable than their more physical counterparts (Arvidsson and Mønsted, 2018). At the same time, however, many incumbent firms and organizations are also struggling to transform their business models and ways of working. Incumbent firms are often limited by legacy systems that are hard to discontinue (Mehrizi et al., 2019), current ways of sourcing IT (Law, 2018), and transformation towards more agile ways of working can to be long-drawn and paved by uncertainty (Dikert et al., 2016). Given these typical circumstances in incumbent organizations, it is likely that the way that the IT-function is organized and the way it operates is important for the organization's capability to undertake digital transformation.

This article posits that fundamental changes in the way that the IT-function is organized and practiced are necessary prerequisites for successful digital transformation and digitalization to take place. To put it bluntly, the IT-function requires transformation, in order to be able to facilitate digital transformation of the firm. However, besides studies on the normative models of the IT-function (Guillemette and Paré, 2012) and numerous studies of various IT governance models (Gregory et al., 2018), there are few empirical studies that investigates the actual transformation process of the IT-function in incumbent organizations in contexts of digital transformation. Hence, the main objective in this paper is to investigate to what extent current organizing and practices of the IT-function produces different forms of technical and organization inertia and how such inertia can be broken in order to promote transformation and change. In so doing, we build on insights from path dependence theory (Sydow et al., 2009) in order to theorize a specific form of inertia – namely path dependency and explain why intended changes typically are so hard to implement. More accurately, we conceptualize path dependency of the IT-function as an unintended cause of historical and contingent decisions regarding IS development methods, delivery models, sourcing strategies, and organizing, that over time generate self-reinforcing dynamics that reinforce rather than transform existing ways of working. While path dependency theory has been used in various forms and ways in previous IS research (e.g. Mehrizi et al., 2019; Sing et al., 2015), we are primarily interested in explaining how path trajectories can be broken, and the specific mechanisms and practices that are likely to trigger transformation of paths. In contrast to Sydow et al. (2009) who favors organizational and managerial practices as sources for path breaking, we see path dependency in context of IT-organizations as socio-technical, and hence that mechanisms for breaking a path can be both technical and organizational, as well as their interaction.

Empirically, we present a longitudinal case study of an IT organization in a governmental organization in Norway. Our case organization had for many years struggled with limited progress in digitally transforming itself and to digitalize its service for citizens. Our analysis reveals that changing the IT organization at the government organization were challenging due to the inertia of large projects and outsourcing of software development, a complex digital infrastructure making deployment of new software risky, and a lack of IT and development competence. Against all odds, however, after a major failure of a large software development programme in 2012-14, the government organization managed to turn around its IT organization and establish cross-disciplinary teams, deliver new software continuously, and re-organizing and excising outsourced projects. As a consequence, the government organization has been able to digitally transform a much larger part of their citizen services than before.

Grounded in this analysis the paper suggests that although IT organizations are likely to struggle with inertia and in particular path dependency it is possible to break out of exiting trajectories and transform themselves. Henceforth we theorize that in context of IT organizations that are characterized by outsourcing of software development, legacy IS, and few and big deliverables, some practice seems to be more likely than others to break path dependency. In particular the paper elaborates on three mechanisms for breaking path dependency in such context that relates to: 1) use of digital platforms in software development; 2) attract new competence; and 3) establishment of cross-disciplinary teams.

The paper is structured in the following way. First, we review relevant literature concerning inertia and change of IT organizations, and then we in Section 2.2. develop a framework for conceptualizing inertia in IT organizations as path dependency and mechanisms for breaking with paths. Section 3 covers the description of the case study as well as some brief insights on data collection and analysis. Furthermore, in Section 4 we present the case narrative. Next, in the concluding section, we discuss our contributions and some concluding remarks for further research on the topic of inertia and change in IT organizations.

2. BACKGROUND LITERATURE AND THEORY

2.1 Inertia and change in IT organizations in context of digital transformation

The concept of *digital transformation* reflects a substantial change in the role that digital technologies play for individuals, organizations and the society as a whole. In a recent paper by Hinings et al. (2018: p. 52) relates digital transformation explicitly to digital innovation, practices, at various levels of analysis, by defining it as: "*the combined effects of several digital innovations bringing about novel actors (and actor constellations), structures, practices, values, and beliefs that change, threaten, replace or complement existing rules of the game within organizations, ecosystems, industries or fields*". In this paper, we comply to Hinings et al.'s definition of digital transformation, as a context for what organizations are trying to achieve through their IT-function.

As much as digital transformation implies radical changes in terms of both technologies and organization, inertia has been put forward as a significant barrier for digital transformation. Vial (2019) finds that inertia is one of the “most significant barriers of” digital transformation. In the context of digital transformation, lack of organizational capabilities and resources, as well as path dependency across supply chains in specific industries are identified as typical sources of inertia. However, inertia can also refer to the limited changeability of digital technologies in use in organizations, due to path dependency because of high switching costs (Shapiro and Varian, 1999), digital debt (Rolland et al., 2018), entrenched legacy IS (Mehrizi et al., 2019), and complexity of the installed-base of large-scale infrastructures (Aanestad and Jensen, 2011).

In terms of the IT-function and especially regarding software development, the literature on agile software development proposes not only that IT organizations should adopt agile methods but a more comprehensive agile transformation (Dikert et al. 2016). This shift requires a tighter integration of business strategizing, and all phases and types of work involved in software development and deployment are needed. In order to achieve this, a new organizational structure has been proposed (Fitzgerald and Stol, 2017). First, DevOps incorporates the two words of development and operations, bridging the gap between the two. It emphasizes effective and unified collaboration (Ebert et al. 2016; Lwakatare et al. 2015). This shift means a change towards cross-functional teams working with continuous feature deliveries as opposed to organizational silos performing these functions separately (Ebert et al. 2016). Further, DevOps signifies the demand for a closer connection between the development of software and the deployment into production (Fitzgerald and Stol, 2017). Although agile methods and continuous development practices have become appealing alternatives for many organizations, it has proven challenging to introduce such methods in larger organizations and enterprises. Henceforth, recent literature has frequently pointed at failures and inherent challenges of agile transformation processes (Dikert et al., 2016; Korvonen, 2013; Paasivaara et al., 2018), challenges in adopting DevOps (Kuusinen et al., 2018), and specific CD practices (Chen, 2015). In a systematic literature review of large-scale agile transformations Dikert et al. (2016) have identified a number of challenges. These include change resistance, coordination challenges in multi-team environments, hierarchical management and organizational boundaries, and requirements engineering challenges.

Paasivaara et al. (2018) studied large-scale transformation at a global telecom company, Ericsson, and confirm that such processes are not without organizational problems and challenges. In meeting these challenges, the authors recommend: 1) an experimental approach to transformation, 2) evolutionary implementation of new ways of working, 3) increased specialization of teams’ work, and 4) to use a common agile framework for the entire organization. However, since most studies are largely descriptive, the current literature is relatively weak on theorizing why certain transformations succeed and why others fail. Henceforth, also agile transformations of IT organizations can become a barrier due to inertia of existing software development practices.

Highly relevant for transforming IT organizations is also the level and content of outsouring of different IT functions. Law (2018) gives insights how IT sourcing decisions have inherent inertia and path dependencies in the way that it influences competence, ways of working, and organizing of the IT function. Grounded in a longitudinal case study, the author shows how sourcing of IS development has path dependencies (Law, 2018). Moreover, the study also shows how organizations can break with current paths and generate new paths through resource commitment, transferability of competence, and mobilizing for expanding the scope of options for selecting superior alternatives.

With this backdrop, this paper argues that there is a need for understanding the role of the IT-function in relation to digital transformation. As digital transformation involves combinations of different technologies (Hinings et al., 2018), specific organization capabilities (Li et al., 2018), agile ways of working across and integrating business strategy, software development and IT operations (Chan et al., 2019; Fitzgerald and Stol, 2017), and new governance regimes (Gregory et al., 2018), it is necessary that the competencies and practices of the IT-function changes as well.

2.2 Path dependency as a lens for conceptualizing transformation of IT organizations

In order to conceptualize and explain transformations of software organizations, such as the IT organization in the Government Organization, we adopt a lens of path theory (Garud et al., 2010; Meyer and Schubert, 2007; Singh et al., 2015; Sydow et al., 2009; Sydow et al. 2012). Path theory, or path constitution theory, comes in different flavors, and has been developed in multiple streams of literature. In general, path theory attempts at integrating the (somewhat) competing theories of path dependency and path creation. Path dependency and the related concept of lock-in to a situation with no remaining decision options for renewal, were initially coined in evolutionary economics as a theoretical basis for explaining why actors select less than optimal solutions in constituting the path of complex technologies (Arthur 1989; David 1985). In short, path dependency implies that historical choices and events narrow actors' choice options in the present (Meyer and Schubert, 2007). As partly a critique and an extension, Garud and colleagues (Garud et al. 2010; Karnøe and Garud 2012) developed path creation as an alternative view emphasizing that actors can mindfully deviate from existing paths. Applying a path theory perspective on IT organizations, then, opens the analytical possibility for explaining *both* why even quite radical changes in for example development approaches and practices (Laanti et al. 2011) and sourcing strategies (Law, 2018) takes place, and why similar changes in other cases do not seem to materialize – at least not to the same extent (Bick et al. 2017).

Perhaps most relevant for transformations of IT organizations is Sydow et al. (2009)'s operationalization of path theory on an organizational-level. They develop a framework for analyzing path constitution across three distinct phases: 1) preformation phase; 2) formation phase; and 3) lock-in phase. They describe path constitution as a process that at some point reaches a critical juncture that marks the transition from a preformation phase into a formation phase. At this point, a specific pattern or organizational path starts emerging, and a self-reinforcing mechanism kicks in. Because of the self-reinforcing mechanism, the options for organizational actors to deviate from the organizational path are increasingly diminishing.

Central in Sydow et al.'s (2009) conceptualization is that path dependency and the constitution of paths are driven by *self-reinforcing mechanisms*. The authors develop a framework consisting of four more specific occasions in organizations or organizational units, where self-reinforcing effects can take place. First, there are *coordination effects* caused by actors adopting the same organizational routine or rule, so that it becomes increasingly attractive for others to adopt the same routines and rules. Thus, coordination gets increasingly more efficient and less costly. The second element in the framework is *complementary effects* which refers to self-reinforcing effects arising out of "the interaction of two or more separate but interrelated resources, rules, or practices" (Sydow et al., 2009: p. 699). Complementary effects are generated through combinations of multiple routines and practices, so that it becomes increasingly attractive to adopt all of them as an "institutional cluster" (David 1994). Third, there are self-reinforcing *learning effects* at different levels in organizations. Obviously, the more often a specific task is performed, the easier it becomes to perform it, and the more efficient it is executed. Arguably, all these three variants of self-reinforcing mechanisms are relevant for IT organizations. In terms of learning effects and coordination effects related to choices in development approaches (i.e. waterfall, hybrid, or agile), development platforms (i.e. AWS, Google, or Azure), programming languages (i.e. Java or Python), and deployment routines (i.e. seldom or continuous). Moreover, as illustrated in a case study by Law (2018), sourcing models do indeed carry strong path dependencies. Having first outsourced software development activities, makes it progressively harder to break this setup due to self-reinforcing mechanisms of complementary effects (i.e. sourcing model, competence, and contracts). A fourth aspect where self-reinforcing mechanisms can play casual part, is the so-called *adaptive expectation effects.* Since individual preferences or choices are affected by other relevant actors' expectations, it becomes increasingly preferable to choose a certain solution that is perceived as 'right' by a growing number of relevant actors. This is also a question of legitimacy: "…individuals or subsystems not subscribing to the mainstream practices are afraid of losing legitimacy and – if associated with failure – of becoming stigmatized as "outsiders"" (Sydow et al., 2009: p. 700). This mechanism is highly relevant in relation to large IT efforts that have inherent risks and are often prestigious (Fitzgerald and Russo 2005; Flyvbjerg et al. 2003). Whereas these organizational phenomena have the potential – through a self-reinforcing mechanism – to generate path dependency and lock-in effects, there are also contextual conditions of ambiguity and complexity that can increase the possibility for a self-reinforcing mechanism to be triggered (Sydow et al, 2009). For IT organizations this issue is also highly relevant as they often serve various user groups with diverging interests and needs, and thereby increasing the ambiguity in expectations towards new software solutions. IT organizations are often also struggling with already complex digital infrastructures that may lead to conservative deployment schemes there users only seldom get new software and upgrades.

However, the self-reinforcing mechanism does not explain how breaking of a path leads to the possible formation of new paths. The original version of path dependency theory argued that paths could only be broken by external shocks or extraordinary events (Myers and Schubert, 2007). In contrast, path creation as coined by Garud et al. (2010) is perhaps the most elaborate attempt at describing how actors can *deliberately* break paths. In a path creation perspective, agency is seen as a largely distributed underscoring of possibilities in which many different actors in different situations and contexts can contribute to deliberately 'breaking' existing paths. Furthermore, path creation emphasizes that novel paths can stem from improvisation and bricolage, as well as actively cultivating serendipity as for example the case of innovating the famous Post-it Notes (Garud et al., 2010). Law (2018) shows how path dependencies can be broken by strategically mobilizing resources for creating new paths. Rolland et al. (2015) use the terms architectural path dependencies and architectural hacking, in order to theorize evolution of enterprise architectures both is influenced by path dependencies and gives opportunities for path creation. In this paper, we will suggest the term *path breaking* as a collective notion explaining the practices and digital technologies that render existing organizational paths obsolete. We summarize the theoretical concepts and their relevance to IT organizations in Table 1.

Table 1. Conceptualizing Transformation of IT organizations

Concept	Definition	Mechanisms	Relevance
IT organizational path dependency	The continuation of inflexible and possibly inefficient action patterns that are shaped by the unintended consequences of former decisions and positive feedback processes (Sydow et	There are four types of self-reinforcing mechanisms that can lead to organizational path dependency (Sydow et al. 2009): 1. Coordination effects 2. Learning effects 3. Complementary effects	Coordination, learning and adaptive expectation effects potentially causing path dependency in relation to: ISD methods, delivery routines, sourcing models, tools and infrastructure (Law, 2018). Complementary effects related to successful combinations of software development methods, delivery routines, sourcing models, tools and

	al., 2009)	4. Adaptive expectation effects	infrastructure.
Path breaking in IT organizations	The transformation of an existing organizational path that radically breaks with historical action patterns (Djelic and Quack, 2007)	There are three types of mechanisms found in the literature for breaking and possible generating new paths: 1. Experimenting with new practices and technologies (Rolland et al., 2015) 2. Paradoxical intervention (Sydow et al., 2009) 3. Mobilizing resources for collective action (Garud et al. 2010)	Recent evidence that many IT organizations manage go from traditional waterfall-like approaches to agile or hybrid development methods even in large-scale projects (Laanti et al., 2011). Also, path dependencies and near lock-in regarding sourcing models can be broken (Law, 2018).

3. METHOD AND CASE DESCRIPTION

3.1 Case description

The fieldwork was conducted within the IT organization of a large public Norwegian welfare organization. The Government Agency (GA) is one of the pillars in the Norwegian welfare system and is responsible for redistributing one third of the national budget through schemes such unemployment benefits and pensions. The organization was established in 2006, following the reform of the welfare system. With the reform, the formerly independent Employment agency and National insurance agency were merged in to a single unit. In addition, the reform also involved the establishment of a formal collaboration between GA and the municipal social services. In total, the organization employs approximately 19,000 people, of which 14,000 are central government employees and 5,000 are employed in the municipalities. In 2016, the IT organization at GA was relatively small compared to the number of employees with 510 employees, where approximately 200 are working with IT operations. In addition, there was approximately 400 consultants working on small and large IT-projects.

Shortly after GA was established, the Parliament elected a reform of the Pension system. To realize the reforms, GA had to develop a new system for the management of age pensions. The old system was based on dated technology and could not support the required changes. The resulting project was large, with an estimated development cost of several hundred million Euros. GA neither had the resources nor the competence to run a project of this magnitude. The project was therefore outsourced to consultant companies. The solution was delivered on time and within budget, and although the IT organization was skeptical many of the architectural decisions made by the consultant company, the project was considered a success among users and business experts. The outsourcing of software development would last for a decade. A waterfall approach to software development was institutionalized across the IT organization. This methodology was well documented and supported by training schemes, reporting routines, and tools. Deployment was organized as large coordinated releases that could approach 150 000 development hours. Henceforth, there was path dependence related to the competence (e.g. learning effects) in the IT organization as they were experts on controlling and planning outsourced projects and IT operations, but they had less expertise in actual development work.

3.2 Data collection

The paper draws on a longitudinal case study (Pan and Tan 2011; Yin 2009). Data was collected through in-depth semi-structured interviews (Myers 2013), observations, and document analysis. A total of 41 interviews were conducted (Table 2). Interviews lasted approximately 60 minutes. Data was collected both from the line-organization (section for IT Architecture), and from ongoing projects – with particular emphasis on an ongoing mega project (the Parental Benefit project). Data collection started in February 2017 and lasted to February 2019.

Table 2. Overview of Interviews

Roles	The project	Architecture unit	Other	Total
Managers	12	2	1	15
Developers	11		3	14
Architects	6	4	1	11

3.3 Data analysis

The data analysis was iterative and overlapped with data collection, thus granting flexibility to respond to emergent themes (Eisenhardt 1989). This followed Pan and Tan’s (2011) process approach, with a framing cycle, followed by an augmenting cycle. During the framing circle, we reviewed documents, did participant observation, and conducted the first batch of interviews. The data material was then written up in a case narrative covering the past 12 years. The narrative provided an overview of events leading up to the agile transformation. The data material was analysed thematically recognizing both self-reinforcing and path breaking mechanisms.

Based on the case narrative we identified defining episodes in the transformation of GA. We describe these stages more fully in the case results section. For each phase, we identified mechanisms that either reinforced or broke with the existing path.

4. CASE NARRATIVE

In our analysis of the case, we have chosen to zoom in on four phases that represents four important attempts at transforming GA's IT organization over a time period of seven years. This is summarized in Table 3 below.

Table 3. Four Phases in the Transformation of GA's IT organization

Phase	Description	Outcomes
Phase 1 (2011-2015) – Failed IT program	A major IT program to modernize and substitute existing legacy systems fails. Management responded by reverting to the old strategy, thereby reinforcing the existing IT organizational path.	**Reinforcing existing IT organizational path:** • Outsourcing of software development continued. • Staged waterfall-type software delivery method with agile practices within each stage. • Large coordinated releases continued. • Application platform allowed for semi-automated provisioning and deployment
Phase 2 (2016-17) – Successful piloting of new software development practices	In the wake of the failed IT program, a new CEO and CTO are appointed. A project is established to pilot cross-disciplinary development teams and continuous deliveries.	**Breaking with current IT organizational path:** • The Pilot project represented a break with current practice and demonstrated continuous delivery and cross-disciplinary development teams as a doable alternative. • The project develops new functionality and has few dependencies to other systems and projects.
Phase 3 (2017) – Mobilizing a new sourcing strategy	NAV realizes a new sourcing strategy by attracting talents from the IT community. The IT organization is re-organized giving room for different ways of working.	**Generating a new IT organizational path:** • Insourcing of software development. • Large projects still employ staged waterfall process and large coordinated releases. • Introduction of second-generation application platform based on Kubernetes, an extensible open source container platform that allows for fully automated deployment and provisioning and monitoring of software
Phase 4 (2018) – Fully implementing BizDevOps	New delivery model based on multidisciplinary teams that have responsibility for entire process within a "product area". Cross-disciplinary teams and continuous delivery practices	**Reinforcing existing IT organizational path:** • Continued insourcing strategy. • Mixture of staged and incremental delivery methods. • Mix of continuous and coordinated deployment.

	introduced into existing projects with high complexity and many dependencies.	• Increasing number of applications migrated to new application platform.

4.1 Phase 1: Reinforcing the existing IT organizational path after a failure project

As the merger was completed and all front-line offices had been established in 2010, GA was ready to plan its digital future. Hence, in 2011, they begun to plan a large modernization program. The program would replace one of the oldest and largest management systems - a benefit management system which had been in operation since 1978. The system was built on dated technology and had become difficult to change and maintain. The system would be replaced through three consecutive projects, where the commencement of one project would depend on the successful completion of former. Each project would last for approximately 2 years with an estimated cost of close to 340 million Euro. Due to an elected reform of legislation governing payments of disability benefit, the first project would develop functionality for the payment of disability benefit. Disability benefit is a welfare benefit that is paid to persons who have a permanently reduced earning capacity due to illness or injury. The reform was elected by Parliament in 2012 and would take effect 1. January 2015. Hence, this date marked an absolute deadline for the project.

At this point in time GA had a long history of outsourcing software development. Since the project was considered strategically important, it was however run by internal resources. The project was perceived as an opportunity to fix "everything" that was malfunctioning. Unfortunately, an ambitious scope, combined with strict deadlines and lack of experience led to failure. The project was terminated after just 6 months on ground of "unforeseen complexity". The failure led to an open parliamentary hearing, and eventually triggered the resignation of the GA Director and was later followed by the resignation of the IT director. To be able to support the legislative changes initiated by the reform, the organization returned to the delivery strategy that had successfully in the past: The project was outsourced in its entirety to external consultants, and the attempt at transforming the organizational path ended up reinforcing it: Disability benefits were implemented as part of the Pension system, using the same methodology as they had done in the Pension project. Henceforth, the failed project also created strong adaptive expectation effects for other projects to follow the outsourcing strategy to software development.

4.2 Phase 2: Breaking the existing IT organizational path by experimenting with new development practices

Although the IT organizational path remained unchanged after the Disability project, it had sparked several initiatives. Among these were the development of an application platform, and the development of digital follow-up activities related to sick leave. The project, which was called DigiMed, departed from existing development practices, delivering software continuously and incrementally. Even though it was small, the project represented a new way of developing and managing software and became a pilot for a new delivery strategy. In place of handovers and the waterfall type method, software was developed by cross-disciplinary teams that were responsible for the entire software development cycle. Software was released frequently, triggering continuous feedback from sponsors and users. In the aftermath of the failed IT program, GA also appointed a new IT Director. The new Director immediately started establishing a new IT organization and proclaimed that GA should change its strategy and rely less on consultants and more on internal expertise. The promise of working with high-end technologies and the radical strategy of the IT Director – breaking with a focus on outsourcing development and working in multidisciplinary BizDevOps teams also attracted experienced developers to GA's IT organization. This contributed to breaking the current organizational path dependency as the new developers mobilizing support for this way of working. The IT director later described the DigiMed project as an important front-runner for changing the existing organizational path: "*It demonstrated the value [of working like this]*". In this way the new IT Director managed to mobilize resources for breaking with the existing ways of conducting software development in cross-disciplinary teams and practicing Continuous Delivery. As the DigiMed-project was successful, it represented a superior way of working compared to the old-fashioned waterfall-approach.

4.3 Phase 3: Mobilizing a new IT organizational path

In 2017, GA decided to change their outsourcing path, and went from outsourcing to insourcing of software development. The insourcing strategy was accompanied by a reorganization of the IT department. The hierarchical structure was replaced by a flat organization, and the old management team was replaced by managers that believed in the strategic shift. The DigiMed pilot project had proven successful, and the IT Director- wanted to scale cross-disciplinary development teams to the rest of the IT organization. The ambition was to take leadership and ownership of their own solutions and developing expertise in-house – thereby assuming larger responsibility for systems development, maintenance, and operations.

Unlike the prior organization, where there were handovers between departments, the new strategy gave teams responsibility for the entire product development cycle. This was achieved by establishing autonomous, cross-functional teams, having different expertise such as architects, developers and testers in one team as opposed to the former organization, where specification, quality assurance and operations were performed as own units or departments in the organization. Consequently, establishing cross-disciplinary development teams, where each team had to know everything about the solutions that support their product, and what framework conditions they operated under, as well as being responsible for own architecture and the relationships to other applications.

Since GA lacked internal development competence, consultants had to be replaced gradually. This was done by establishing internal product development teams that were staffed with consultants, but with a GA team lead. The consultants in the product teams would be replaced as GA succeeded with their employment strategy. Close to 200 developers would be employed within the next two years and responsibility contracts where consultant firma were responsible for development and maintenance were gradually replaced by capacity contracts, limiting the use of consultants to peak periods and specialist competence.

Further, to facilitate the transformation of IT organization, GA developed an open source framework based on Kubernetes. The platform provided automated provisioning and deployment and allowed development teams to deploy and manage their own applications without the involvement of the Operations department. *"The platform takes the world's best operations person and automates him"* (Member of platform team). Hence, the use of container-based platforms strengthened the new organizational path, as it enabled a distributed governance structure where cross-disciplinary teams could work in relative isolation. The new platform, together with cross-disciplinary development teams, and an insourcing strategy also slowly sparked coordination effects and complementary effects for the new overall strategy. In addition, the use of new platform technology was an effective way of attracting IT competence in a competitive market.

By the end of 2017, the organization was split between two competing organizational paths. On the one side stood proponents of the old organizational path; the managers and coordinators that had specialized in procurement and control. On the other side stood developers and managers who believed in cross-disciplinary development teams and CD.

4.4 Phase 4: Reinforcing a new IT organizational path by fully implementing cross-disciplinary teams

The IT organizational path supporting insourcing was reinforced by applying the new delivery model to an ongoing mega project. Up-until then, the new delivery model had only been used in smaller projects with a limited number of dependencies. The IT department recognized that to succeed with the strategic shift, they needed to implement the new strategy across projects, including the large and ongoing parental benefit project. Attempting to implement CD in an ongoing mega project was however challenging. To initiate the change, the mega project begun by establishing a single cross-disciplinary team that would work alongside the traditional development teams. Even through the cross-disciplinary team only developed a small and isolated part of the system, they demonstrated the benefits of working in multidisciplinary team. The establishment of the cross-disciplinary team helped convince project sponsors that the entire project could deliver better value be working in multidisciplinary teams.

One of the essential initiatives to provide better conditions towards CD was to terminate the deployment process characterized as "main deliverables". GA and the parental money project were used to coordinating all applications, projects and environments and bundle them into a main deployment schedule. In order to terminate this deployment process, it was necessary to separate and isolate applications and systems for quicker deployment and decreasing the number of dependencies. This required isolation between applications, reducing dependencies so deployment did not need to be coordinated and delivered at the same time. Also, if applications and systems were broken up, and could rotate independent of each other, it could reduce the need for plan-build-run organizing. This would replace handovers between departments with autonomous teams, operating independently, taking responsibility for the entire system development cycle by being cross- functional. As applications and teams had varying environments and context to work by, the control system of white, grey and black-listing helped to distinguish which teams and applications were mature enough for CD. Teams and applications that managed to isolate themselves were allowed to be white-listed and could adopt CD. If the application had too many dependencies, they needed to be coordinated into main deliverable, bundled with other applications as the ability to modify these systems were low.

5. DISCUSSION AND CONCLUDING REMARKS

This paper aims at empirically describe and theoretically explain both inertia and change in IT organizations. Insight into such transformations are important as organizations are moving towards becoming increasingly "digital organizations" involving a fusion between IT strategy and business strategy (Bharadwaj et al. 2013). This shift often requires IT organizations to transform their sourcing models, IT-organization, and move towards cross-disciplinary teams continuously deploying new software and systems (Fitzgerald and Stol, 2017). In order to theorize and explain outcomes of such processes in IT organizations, we draw from path theory (Garud et al. 2010; Mahoney, 2000; Meyer and Schubert, 2007; Sydow et al., 2009) and show how IT organizations can develop path dependency as well as breaking long enduring path trajectories. Grounded in our path analysis of a longitudinal case study, we provide two distinct contributions.

First, we theorize how IT organizations tend to develop path dependency related to interaction and interdependencies between development activities and methods, IT sourcing practices, and software delivery routines. These important aspects of the IT organization tend to become largely complementary entities making each individual entity harder to change because of interdependencies and self-reinforcing mechanisms. As seen in the case study, although many of the involved consultants and internal employees in GO's IT organization were strong proponents of a transition towards cross-disciplinary teams and continuous development, learning and coordination effects prevented a transformation during Phase 1. Furthermore, the complementary effects of a historical decision to follow a strategy for outsourcing software development while simultaneously insourcing IT operations, made cross-disciplinary teams very hard to implement as it required experts from IT operations, developers from consulting companies, and business experts to work together in teams. Finally, also adaptive expectation effects were present in Phase 1 of the case. Top management of the GO organization and other Government departments expected the IT organization to return to the old regime of outsourcing software development that had worked out previously. Paradoxically, the attempt at transforming the way the IT organization worked towards more agile practices, GO ended up reinforcing the existing organizational path characterized by outsourcing, waterfall development, and conservative deployment rates. Furthermore, we theorize that a setting with complex digital infrastructure with legacy systems seems to lower the bar for development of path dependency. In this way, organizations with more modularized and standardized components and IT systems are likely to be less vulnerable to path depended as for instance outsourcing of some of the systems and development activities will not be so interdependent with the rest of the IT organization.

A second contribution is related to the practices and mechanisms that helped break the path trajectory characterized by outsourced software development, waterfall methods, and long-drawn routines and infrequent delivery of new software. Consistent with a path lens (e.g. Garud et al., 2010; Singh et al., 2015; Sydow et al., 2009), our case shows that IT and software organizations can indeed break with existing organizational paths and transform themselves. Especially, based on the longitudinal case presented in this research, we theorize that three practices were essential in breaking the IT organizational path dependency. First, we theorize that experimenting with and implementing different kinds of cross-disciplinary development teams is a path breaking mechanism. In the case of GO, introducing cross-disciplinary development teams was to conduct a paradoxical intervention (Sydow et al., 2009) as there was a stringent division between the line organization and the software development project. This led to a more effective software development process avoiding numerous hand-overs that involved misunderstandings and immature design decisions. Cross-disciplinary teams involved a fusion of different competencies improving technical decision making, and integrated insights from the line organizations with the development expertise. A second mechanism was related to how the IT organization attracted and secured competence. In the case of GO, the new IT director did this by publicly announcing in newspapers and at practitioner's conferences that GO was turning 180 degrees to implement insourcing, autonomous teams, and continuous delivery. This opened the scope of action as it made it possible for the IT organization to attract individuals that had experience and competence with cross-disciplinary development teams and continuous delivery from high-end platform companies. This, however, came as a surprise for consultant companies delivering on the existing outsourced projects. But, for other actors at

GO it provided an opportunity to re-adjust the way that current projects were run. So, during Phase 4, the outsourced projects and the consultants working on the customer-side of the projects did not have many choices as they were offered new favourable contracts complying to the new way of working. Consequently, we theorize that in order to transform an IT organization attracting and securing competence becomes crucial. Thirdly, the introduction of digital platform for new software development t virtualizing infrastructure resources and automating many IT operations tasks is crucial for the transformation of IT organizations. Thus, we theorize that an essential path breaking mechanism not indicated in previous literature, was in our case related to an applications development platform configured by a team of newly hired experts. This platform made it easier to introduce cross-disciplinary development teams that could develop and deploy new software in a relatively autonomous manner as it provided more standardized interfaces to some of the complex integration and infrastructure issues. The new platform also automated previously complex tasks related to monitoring and testing. Moreover, the platform was very important for attracting new developers and technical architects to GO as it was seen as the current state-of-the art for developing complex software systems. In this way, the platform can be argued to be a mechanism that in interaction with the other two path breaking mechanisms helped establish cross-disciplinary teams and continuous delivery practices on a more permanent basis. The use of new technologies and platform technologies are not much recognized in existing literature on agile transformation. For example, Dikert et al., (2016) that offers a comprehensive literature review does not mention this as part of their list of success factors. A different turn is to look at how new digital technologies and layered modular architectures (Yoo et al. 2010) such as digital platforms has a potential to become a mechanism for establishing cross-disciplinary teams and continuous delivery practices – given certain contextual conditions. For example, Rolland (2018) explain how a large media company managed to renew their digital infrastructure and process in media production by continuously leveraging a digital platform and its ecosystem.

Furthermore, it is interesting how these three mechanisms seem to have worked well together in a complementary fashion that in the end lead the IT organization down a very different path trajectory characterized by insourcing of software development, highly agile and cross-disciplinary teams, short and frequent delivery routines, and over time the discontinuation of some of the most crucial parts of a legacy system.

Taken together this provides us with new implications for digital transformation of incumbent organizations. Our research shows that it is essential to transform the IT organization in order to be able to digitally transform the organization as a whole. This essential part of digital transformation processes seems to be nonexciting in current research on the topic. Regrettably, current IS research on digital transformation also fails to underscore how digital technologies both can facilitate change and as well be a source of inertia. The literature on digital innovation and transformation provides numerous examples of how different kinds of digital technologies, like Internet of Things, big data analytics, and robots facilitate changes in organizations (e.g. Barrett et al. 2012; Jonsson et al., 2018; Lehrer et al., 2018; Nicolescu et al., 2018). However, there are only a small number of publications investigating at a finer level of granularity what components and aspects of digital technologies are changed over time and how they can reinforce inertia and path dependency as well as a flexibility to change.

The research also shows that IT organizations in incumbent organizations are likely to suffer under path dependence which make it hard to change. Henceforth, although IT organizations can be organized and structured in multiple ways as portrayed by Guillemette and Paré (2012), moving from one ideal model to a different one seems problematic because of path dependency.

To conclude, a path lens analysis, enables us to explain *both* why radical changes in for example development approaches and practices can be deliberately implemented and why similar changes in other cases do not seem to materialize – at least not to the same extent (Bick et al., 2017). It also underscores that it is not enough to look for single success factors or independent factors that challenge transformations of IT organizations (Dikert et al., 2016), but to look at how different interdependent issues and mechanisms of the IT organization are producing or breaking organizational paths. Furthermore, a path analysis explains how the transformation process unfolds over time, and how actors' actions may reinforce or transform the current IT organization (Singh et al., 2015). On a more practical side, our case describes how it is possible for also hierarchical and complex organizations to introduce cross-disciplinary development teams and CD successfully. Along with Paasivaara et al. (2018) we would recommend organizations to introduce cross-disciplinary development teams and CD carefully in an evolutionary manner. However, grounded in our study we would recommend IT organizations to unlock the interdependencies across sourcing models, software development methods and delivery schemes. Also, utilizing development platforms need to be considered in association with cross-disciplinary development teams and CD in order to succeed.

References

Aanestad, M., & Jensen, T. B. (2011). «Building nation-wide information infrastructures in healthcare through modular implementation strategies." *The Journal of Strategic Information Systems*, **20(2):** 161-176.

Arvidsson, V., & Mønsted, T. (2018). Generating innovation potential: How digital entrepreneurs conceal, sequence, anchor, and propagate new technology. *the Journal of strategic information systems*, *27*(4), 369-383.

Barrett, M., Oborn, E., Orlikowski, W. J., & Yates, J. (2012). Reconfiguring boundary relations: Robotic innovations in pharmacy work. *Organization Science*, *23*(5), 1448-1466.

Bharadwaj, A., El Sawy, O. A., Pavlou, P. A., & Venkatraman, N. (2013). Digital business strategy: toward a next generation of insights. *MIS quarterly*, 471-482.

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*), pp. 1-1.

Chan, C. M., Teoh, S. Y., Yeow, A., & Pan, G. (2019). Agility in responding to disruptive digital innovation: Case study of an SME. *Information Systems Journal*, *29*(2), 436-455.

Chen, Lianping. 2015. "Continuous delivery: Huge benefits, but challenges too." *IEEE Software* 32, no. 2 (2015): 50-54.

David, P. A. 1994. "Why Are Institutions the 'Carriers of History'?: Path Dependence and the Evolution of Conventions, Organizations and Institutions," *Structural change and economic dynamics* (5:2), pp. 205-220.

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), pp. 87-108.

Djelic, M. L., & Quack, S. (2007). Overcoming path dependency: path generation in open systems. *Theory and society*, *36*(2), 161-186.

Ebert, C., Gallardo, G., Hernantes, J., and Serrano, N. 2016. "Devops," *IEEE Software* (33:3), pp. 94-100.

Eisenhardt, K. M. 1989. "Building Theories from Case Study Research," *Academy of management review* (14:4), pp. 532-550.

Fitzgerald, B., and Stol, K.-J. 2017. "Continuous Software Engineering: A Roadmap and Agenda," *Journal of Systems and Software* (123), pp. 176-189.

Fitzgerald, G., and Russo, N. L. 2005. "The Turnaround of the London Ambulance Service Computer-Aided Despatch System (Lascad)," *European Journal of Information Systems* (14:3), pp. 244-257.

Flyvbjerg, B. 2006. "Five misunderstandings about case-study research". *Qualitative inquiry*, Vol. 12, issue 2, pp. 219-245.

Garud, R., Kumaraswamy, A., and Karnøe, P. 2010. "Path Dependence or Path Creation?," *Journal of Management Studies* (47:4), pp. 760-774.

Gregory, R. W., Kaganer, E., Henfridsson, O., & Ruch, T. J. (2018). "IT Consumerization and the Transformation of IT Governance." *Mis Quarterly*, **42(4)**: 1225-1253.

Guillemette, M. G., & Paré, G. (2012). Toward a new theory of the contribution of the IT function in organizations. *Mis Quarterly*, 529-551.

Hinings, B., Gegenhuber, T., & Greenwood, R. (2018). "Digital innovation and transformation: An institutional perspective." *Information and Organization*, **28(1):** 52-61.

Jonsson, K., Mathiassen, L., & Holmström, J. (2018). Representation and mediation in digitalized work: evidence from maintenance of mining machinery. *Journal of Information Technology*, *33*(3), 216-232.

Karnøe, P., and Garud, R. 2012. "Path Creation: Co-Creation of Heterogeneous Resources in the Emergence of the Danish Wind Turbine Cluster," *European Planning Studies* (20:5), pp. 733-752.

Korhonen, K. 2013. "Evaluating the Impact of an Agile Transformation: A Longitudinal Case Study in a Distributed Context," *Software Quality Journal* (21:4), pp. 599-624.

Kuusinen, K., Balakumar, V., Jepsen, S. C., Larsen, S. H., Lemqvist, T. A., Muric, A., Nielsen, A. O., and Vestergaard, O. 2018, "A Large Agile Organization on its Journey towards DevOps," in *44th Euromicro Conference on Software Engineering and Advanced Applications*, T. Bures and L. Angelis, Eds., ed, pp. 60-63.

Laanti, M., Salo, O., and Abrahamsson, P. 2011. "Agile Methods Rapidly Replacing Traditional Methods at Nokia: A Survey of Opinions on Agile Transformation," *Information and Software Technology* (53:3), pp. 276-290.

Law, F. (2018). Breaking the outsourcing path: Backsourcing process and outsourcing lock-in. *European Management Journal*, *36*(3), 341-352.

Lehrer, C., Wieneke, A., Vom Brocke, J. A. N., Jung, R., & Seidel, S. (2018). How big data analytics enables service innovation: materiality, affordance, and the individualization of service. *Journal of Management Information Systems*, *35*(2), 424-460.

Li, L., Su, F., Zhang, W., & Mao, J. Y. (2018). Digital transformation by SME entrepreneurs: A capability perspective. *Information Systems Journal*, *28*(6), 1129-1157.

Lwakatare, L. E., Kuvaja, P., and Oivo, M. 2015. "Dimensions of Devops," *International conference on agile software development*: Springer, pp. 212-217.

Mahoney, J. (2000). Path dependence in historical sociology. *Theory and society*, *29*(4), 507-548.

Mehrizi, M. R., Modol, J. R., & Nezhad, M. Z. (2019). "Intensifying to cease: unpacking the process of information systems discontinuance." *MIS Q*, **43:** 141-165.

Meyer, U., and Schubert, C. 2007. "Integrating Path Dependency and Path Creation in a General Understanding of Path Constitution. The Role of Agency and Institutions in the Stabilisation of Technological Innovations," *science, technology & Innovation studies* (3:1), pp. 23-44.

Myers, M. D. 2013. *Qualitative Research in Business and Management*. Sage.

Nicolescu, R., Huth, M., Radanliev, P., & De Roure, D. (2018). Mapping the values of IoT. *Journal of Information Technology*, *33*(4), 345-360.

Paasivaara, M., Behm, B., Lassenius, C., and Hallikainen, M. 2018. "Large-Scale Agile Transformation at Ericsson: A Case Study," *Empirical Software Engineering*), pp. 1-47.

Pan, S. L., and Tan, B. 2011. "Demystifying Case Research: A Structured–Pragmatic–Situational (Sps) Approach to Conducting Case Studies," *Information and Organization* (21:3), pp. 161-176.

Rolland, K. H., Ghinea, G., and Grønli, T.-M. 2015. "Ambidextrous Enterprise Architecting: Betting on the Future and Hacking Path-Dependencies," *ECIS*.

Rolland, K. H., Mathiassen, L., & Rai, A. (2018). “Managing digital platforms in user organizations: the interactions between digital options and digital debt.” *Information Systems Research*, **29(2):** 419-443.

Shapiro, C., Carl, S., & Varian, H. R. (1999). *Information rules: a strategic guide to the network economy*. Harvard Business Press.

Singh, R., Mathiassen, L., and Mishra, A. 2015. "Organizational Path Constitution in Technological Innovation: Evidence from Rural Telehealth," *Mis Quarterly* (39:3).

Svahn, F., Mathiassen, L., & Lindgren, R. (2017). Embracing Digital Innovation in Incumbent Firms: How Volvo Cars Managed Competing Concerns. *Mis Quarterly*, *41*(1).

Sydow, J., Schreyögg, G., and Koch, J. 2009. "Organizational Path Dependence: Opening the Black Box," *Academy of management review* (34:4), pp. 689-709.

Sydow, J., Windeler, A., Müller-Seitz, G., and Lange, K. 2012. "Path Constitution Analysis: A Methodology for Understanding Path Dependence and Path Creation," *Business Research* (5:2), pp. 155-176.

Vial, G. (2019). “Understanding digital transformation: A review and a research agenda.” *Journal of Strategic Information Systems*, **28**: 118-144.

Yin, R. K. 2009. "Case Study Research: Design and Methods. Essential Guide to Qualitative Methods in Organizational Research (Vol. 5)," *The Information Systems Research Challenge (Harvard Business School Research Colloquium). London: Sage*.

Yoo, Y., Henfridsson, O., and Lyytinen, K. 2010. "Research Commentary—the New Organizing Logic of Digital Innovation: An Agenda for Information Systems Research," *Information systems research* (21:4), pp. 724-735.

PAPER IV

Platformizing the Organization through Decoupling and Recoupling

A longitudinal case study of a government agency

Kathrine Vestues
Department of Computer and Information Sciences, Norwegian University of Science and Technology
kathrine.vestues@ntnu.no

Knut H. Rolland
Department of Informatics, University of Oslo; University College Kristiania
knutro@ifi.uio.no

Abstract. Digital platforms are known for their ability to produce and reproduce flexible solutions for multiple user groups and for attracting third-party developers. Consequently, the unit of analysis for digital platforms is often the consumer market. In contrast, in this paper, we investigate how platformization has the potential to increase digital innovation within organizations through a transformation of legacy systems and the organizational structure. Empirically, we draw on a longitudinal case study of a Norwegian government agency following their platformization process over two years. In advancing a sociotechnical perspective on platformization, we offer three distinct contributions. First, we contribute by theorizing how existing legacy systems are discontinued in processes of decoupling. Second, we capture how this enables novel recombination of knowledge and skills in the recoupling processes, which, in turn, facilitates new ways of working and organizing. Third, we theorize on how decoupling and recoupling processes interact, thus facilitating increased levels of platformization.

Key words: Platformization, Digital platforms, Digital infrastructures, Digital transformation, IS development.

Accepting editor: Katrin Jonsson

1 Introduction

Digital platforms have attracted considerable attention from practitioners and information systems (IS) scholars. For practitioners, digital platforms have become the newest buzzword for establishing new business and rapidly scaling services to a large number of consumers. In the IS literature, digital platforms are often portrayed as multi-sided markets enabled by flexible platform architectures. As a combination of technical architectures and business models, digital platforms are capable of producing very varied, as well as high-quality, products and services (Cennamo and Santaló 2019; Gawer and Cusumano 2014; Parker et al. 2016; Tiwana 2010; Wareham et al. 2014; Yoo et al. 2010). Thus, digital platforms are often put forward as technologies with an almost inherent capacity for continuously producing new features and innovations. For example, IS researchers recently investigated how innovation platforms, such as Google Android and Apple iOS, evolve through the provisioning of 'boundary resources' that balance platform control while retaining flexibility and openness to attract third-party developers (Eaton et al. 2015; Ghazawneh and Henfridsson 2013).

Additionally, although not frequently investigated, many organizations utilize platforms as part of the organizations' digital infrastructure—especially as a more interoperable and flexible replacement for existing legacy systems (Rolland et al. 2018). Thus, the current body of research on digital infrastructures is relevant for understanding how organizations adopt and adapt digital platforms, and in particular, how organizations struggle with the inertia of an installed base (Aanestad and Jensen 2011). Therefore, solving problems related to the inertia of the installed base has been a long-standing concern among Scandinavian IS scholars. This stream of IS research has emphasized various solutions and conceptualizations, for example, *cultivation* (Aanestad and Jensen 2011; Ciborra 2000), *gateways* (Hanseth 2001), and *bootstrapping* (Hanseth and Lyytinen 2010). To this end, utilizing platform architectures and design principles seems highly relevant for redesigning infrastructures into modularized architectures that facilitate change and modifications.

While building and expanding on these insights, we investigate a different strategy based on platformization of digital infrastructures in organizations. Platformization has been suggested as a viable way of increasing flexibility and innovation in digital infrastructures (Bygstad and Hanseth 2018; Islind et al. 2016). By breaking monolithic systems into modular applications, functionality can be added, changed, or removed without affecting other applications and modules. For most organizations, however, this strategy requires a gradual transition, in which legacy systems are gradually dismantled into modular applications (Bygstad and Hanseth 2018). Although digital platforms come in many technical forms and organizational arrangements (Gawer and

Cusumano 2014), their layered modular architecture (Rolland et al. 2018; Yoo et al. 2010), and the way in which boundary resources within the platforms can be combined and recombined to produce novel solutions and products (Constantinides et al. 2018; de Reuver et al. 2017; Henfridsson and Bygstad 2013), make them especially suited for transforming existing infrastructures and improving the capabilities for innovation and change.

Against this backdrop, our objective is to explore the potential for digital platforms to allow organizations to change the way they develop and maintain software and thus, increase flexibility and innovation. The mediating capacity of platforms has enabled new ways of developing and offering services (Rolland et al. 2018). However, this capacity is not restricted to open platforms and external developers: By using platforms internally to coordinate the efforts of autonomous teams, organizations can increase flexibility and innovation within the boundaries of a firm. The modular layered architecture of the platform (Yoo et al. 2010) allows cross-functional teams to develop and deploy applications without coordinating their efforts with other teams and applications. This potentially enables innovation which is difficult in fragmented digital infrastructures consisting of a web of interconnected legacy systems. Based on the transformative potential of internal digital platforms, we ask the following research question

What characterizes how processes of platformization unfold in large-scale organizations?

We explored this question through a longitudinal case study within NAV, the Norwegian Labor and Welfare Administration. NAV is the backbone of the Norwegian welfare system and is responsible for redistributing one third of the national budget through programs such as age pensions, sick leave benefits, and disability benefits. The agency is also responsible for stimulating the population's work ability and increasing the number of citizens in active employment.

In 2017, NAV embarked on a major platformization of its digital infrastructure, in which legacy systems were dismantled into modular applications. The present study results show that the dismantling of monolithic systems enables radical changes in the way organizations develop and deliver digital services. Theoretically, we conceptualize platformization as the unfolding of two interrelated processes of decoupling and recoupling. These concepts reflect insights from service-dominant logic, which sees innovation as unfolding through the related activities of liquefaction and resource integration (Lusch and Nambisan 2015). Decoupling denotes the dismantling of legacy systems into modular applications by, for example, using so-called microservices, Platform as a Service (PaaS), and container platforms. Recoupling refers to the social changes enabled by the decoupling. Thus, platformization during decoupling implies utilizing and introducing a platform-oriented governance model combined with the loosely coupled architecture of a platform core and a wide variety of different modules (apps) running on top. By decoupling legacy systems, the organization increases the availability of the digital components. These components can, in turn, be combined and recombined to produce novel services. Recoupling establishes roles and practices that facilitate recombination—providing flexibility and innovation.

Our contribution is threefold. First, we contribute by theorizing how existing legacy systems are discontinued in decoupling processes. Second, we capture how this enables novel recombination of knowledge and skills in recoupling processes, which, in turn, facilitates new ways of working and organizing. Third, we theorize on how decoupling and recoupling processes interact, thus facilitating increased levels of platformization. Generally, decoupling and recoupling processes, although distinctly different, are mutually interdependent. Decoupling paves the way for recoupling, and in turn, recoupling increases the organizational capabilities for undertaking further decoupling. Thus, this theorizing makes the fundamental assumption that platformization is a sociotechnical phenomenon that to succeed requires digital platforms and the appropriate organizational processes and capabilities.

The remainder of the paper is organized as follows. First, we introduce related literature on digital platforms, before we present a theoretical framework for platformization. We then formulate the research methods and results. Finally, we discuss the findings and make concluding remarks.

2 Digital platforms in organizations

In the past decade, the body of scholarly research on digital platforms has grown considerably. IS scholars have published numerous works, including theories of how platforms evolve (Eaton et al. 2015; Ghazawneh and Henfridsson 2013), how platforms increase flexibility in organizations (Rolland et al. 2018), and how platforms enable innovation based on modular layered architectures (Yoo et al. 2010). Platforms and ecosystems have been studied from multiple perspectives and across different fields of research. For example, scholars from disciplines such as economics (Parker et al. 2016; Tan and Wright 2018), accounting (Kornberger et al. 2017), strategy (Jacobides et al. 2018), and systems engineering (Boudreau 2010; Spagnoletti et al. 2015; Tiwana et al. 2010) have conducted empirical studies of platforms.

In a review of the platform literature, Gawer (2014) argues that the discourse on digital platforms has been dominated by two distinct theoretical perspectives: one inspired by industrial economics, the other by engineering design. From an economic perspective, platforms create value by acting as mediators between two or more categories of consumers who would otherwise not connect. This value is linked to platforms as multi-sided markets that produce network effects (Parker et. al 2016). Network effects relate to the ability to build networks where any additional user will enhance the experience of existing users (Vassilakopoulou et al. 2017). The more users participate in the network, the more valuable the network becomes for each participant.

However, the economic perspective has several limitations. First, it sees the platform as fixed and exogenous, and does not consider how platforms evolve and interact with their surroundings. Similarly, as noted by de Reuver et al. (2017), the lack of emphasis on the technical architecture means that many studies within this research stream fail to conceptualize the digital features and affordances of platforms. Certainly, various platforms can be defined as multi-sided markets but will still differ in terms of digital architectures. This is important for how a platform may evolve—or fail to evolve (Rolland et al. 2018). Second, the interaction between the owner and the complementor is reduced to a simple producer-consumer relationship, which fails to capture the complex relationship between different actors in the ecosystem. In platform ecosystems, an actor consumes and produces services, sometimes as part of the same transaction (Lusch and Nambisan 2015). Different technical configurations also affect the way in which actors co-create value. Based on these shortcomings, Gawer (2014) concludes that although the economic perspective provides valuable perspectives on the platform's ability to mediate and coordinate, this perspective fails to address concerns related to innovation and evolution.

The second perspective is the engineering perspective, which sees platforms as technical artifacts with a modular architecture that have a stable core and numerous peripheral components. For instance, Baldwin and Woodward (2009) argue that all platforms share a common trait: They have a modular architecture that is centered on the core and the periphery. In this view, "a platform architecture partitions a system into stable core components and variable peripheral components" (ibid., 24). The layered modular architecture makes platforms particularly well suited for innovation. The reason is three-fold. First, modularity enables the recombination of modules (Henfridsson et al. 2018). Recombination is the most basic form of innovation and happens as actors combine existing modules into novel products or services. By breaking a system into smaller parts with standardized interfaces, modules can more easily be recombined.

Second, defining clear interfaces between modules allows for specialization and autonomous innovation. As long as interfaces remain intact, a development team can innovate within the confines of a module, without affecting other teams or modules. Third, the separation between the core and the periphery abstracts the complexity of the underlying infrastructure, providing developers with "valuable ignorance" of the core's native functionalities (Tiwana 2013, p. 82). This increases the speed of innovation as developers can focus on their own work yet be able to integrate their applications with the platform (ibid.).

However, the engineering perspective also has limitations. First, platforms are viewed as relatively stable structures in terms of software architecture and apps with limited attention in the way in which they evolve. Within organizations, however, there are usually intersecting projects, diverging needs, and numerous legacy systems that will influence and shape the evolution of the platform. Second, although the engineering perspective provides insight into the platform's ability to facilitate distributed and autonomous innovation, the literature provides few insights into how autonomy is exercised and how complementors innovate. Third, from the engineering perspective, innovation is seen as unbounded (Yoo et al. 2012)—limited only by the availability of components. Platforms introduced as a means for increased innovation within organizations, however, are limited by economic, structural, cognitive, and institutional constraints (Yoo et al. 2010, 730).

Based on the identified shortcomings, Gawer (2014) proposes a third perspective on digital platforms that bridges the gap between the economic and engineering perspectives. This third perspective is described as "organizational" and sees platforms as evolving organizations or meta-organizations that "federate and coordinate constitutive agents who can innovate and compete," and technical structures with "a modular technological architecture composed of a core and a periphery." In contrast to the economic and engineering perspectives, the organizational perspective is explicitly sociotechnical, conceptualizing platforms as a "sociotechnical assemblage encompassing the technical elements (of software and hardware) and associated organizational processes and standards" (de Reuver et al. 2017, p. 126). The organizational perspective differs from the engineering perspective in its view on innovation: The engineering perspective sees platform innovation as unbounded, but the organizational view acknowledges that the innovative capacity is influenced by organizational structures and practices.

However, although the organizational research stream acknowledges the bounded, sociotechnical, and evolutionary characteristics of digital platforms, it has several limitations. First, most extant studies adopt an owner-centric view, focusing on the way that platform owners design platforms to maximize their own profits (Spagnoletti et al. 2015). Thus, the organizational stream gives limited attention to the challenges and opportunities that digital platforms give user organizations. Without insight into the way end-users and their organizations develop and use digital platforms, it is difficult to understand how different configurations of platforms and user organizations might enhance or impede innovation. Second, the owner-centric focus gives little attention to the way in which legacy systems affect platform evolution. Today's business organizations often utilize and build on many different platforms and cloud services as an integral part of their infrastructure. For example, Rolland et al. (2018) report on a longitudinal case study of a digital platform for news production in a media company. In this study, the digital platform provides novel opportunities for expanding the company's digital infrastructure and supporting new services and ways of organizing journalists' work. At the same time, this research shows the challenging dilemmas and the balancing act between managing the inertia of legacy systems and the vast opportunities provided by the platform ecosystem. Third, although the organizational stream emphasizes the evolutionary characteristics of digital platforms, it pays limited attention to how these platforms come into existence. Most extant studies examine the evolution of established platforms (Eaton et al. 2015).

In response to these shortcomings, a growing number of studies are adopting a platformization perspective (Bygstad and Hanseth 2018; Islind et al. 2016; Poell et al. 2019; Törmer and Henningsson 2018), examining the way in which digital platforms are created, how they evolve, and the way in which they interact with existing infrastructures.

3 Theorizing platformization

Platformization has become a popular strategy for breaking away from a drifting digital infrastructure (Törmer and Henningsson 2018) and making the digital infrastructure more flexible (Battleson et al. 2016). Helmond (2015) uses the term to refer to the rise of the platform as the dominant infrastructural and economic model of the social web, while Islind et al. (2016) describe platformization as the socio-technical process of creating a platform. Bygstad and Hanseth (2018) define platformization as a process where silo solutions are decoupled into a platform-oriented infrastructure.

In this paper, we also see platformization as the decoupling of systems into a platform-oriented structure. However, we define platformization not only as the process of picking apart but also as the process of putting back together. This implies that platformization consists of two basic processes. The first process decouples systems, information, and activities into modular components, and the second process comes from being able to recouple components that would otherwise be difficult or expensive to combine (Normann 2001).

These insights are compatible with service-dominant logic (Lusch and Nambisan 2015), where innovation is seen to happen through the interrelated processes of 'resource liquefaction' and 'resource integration.' From this perspective, resource liquefaction is defined as the decoupling of information from its physical media. As information is decoupled, it becomes a resource that can more easily be transmitted and recombined by other actors. The availability of resources is measured in terms of 'density,' where maximum density occurs when the best combination of resources is mobilized for a particular context (Normann 2001). Value is created and co-created through the integration of resources.

However, while service-dominant logic sees resource density as the decoupling of information, we take a broader view and include the decoupling of systems. The reason is related to the accessibility of information stored in legacy systems: Although the information is accessible in principle (Kallinikos et al. 2013), poorly designed interfaces combined with tightly coupled systems in many cases make information difficult or impossible to access and modify. To increase the availability of information for external modification and use, legacy systems should be decoupled into smaller components with clearly defined interfaces. For organizations with a large legacy of information systems, resource density can be increased in two ways: either through the decoupling of information from its physical media or through the decoupling of legacy systems into modular components.

Digital platforms with their layered modular structure are particularly well suited for increasing resource density and facilitating resource integration (Yoo et al. 2012). This is because digital platforms are characterized by standardized interfaces between components—making them easy to access and combine within and across layers. The platform also provides a venue where actors can access and exchange resources, thus enabling resource integration and innovation.

However, the platform structure with its stable core and modular periphery not only leverages resource density and enhances resource integration. The platform also provides a fluid structure that enables a dynamic reconfiguration of value paths. As work processes are decoupled into modular applications, the processes can be reconfigured and reallocated to the most appropriate part of the organization. This allows for a dynamic reconfiguration of work practices and organizational arrangement by interleaving the decoupling and recoupling processes. Novel value paths will emerge—gradually transforming the organization. Eric Evans (2006) describes the challenges involved in achieving such modularity:

> The goal of the most ambitious enterprise system is a tightly interconnected system spanning the entire business. Yet the entire business model for almost any such organization is too large and complex to manage or even understand as a single unit. The system must be broken into smaller parts, in both concept and implementation. The challenge is to accomplish this modularity without losing the benefits of integration, allowing different parts of the system to interoperate to support the coordination of various business operations.

Concepts	*Activities involved*	*References*
Decoupling	Decoupling information from its physical media (Liquefaction)	(Normann 2001; Lusch and Nambisan 2015)
	Decoupling legacy systems into modular applications	(Bygstad and Hanseth 2018)
Recoupling	Recombining knowledge and skills into new organizational forms and practices	(Fitzgerald and Stol 2017; Brown and Duguid 1991)
	Recombining applications and teams into domains	(Evans 2006; Zammuto et al. 2007)

Table 1. Platformization theorized.

4 Research method

4.1 Case background

Fieldwork was conducted within the IT department at NAV. NAV was established in 2006 following reform of the Norwegian welfare system. The reform involved a merger of the formerly separate Employment Services and National Insurance Services. In addition, the reform involved a formal collaboration between NAV and municipal social services. In total, the organization employs approximately 19,000 people, of whom 14,000 are central government employees, and 5,000 are municipality employees.

NAV has a dual responsibility: The agency stimulates the population's work ability to increase the number of citizens in active employment and supports them economically during periods when they are unable to support themselves. NAV redistributes close to one third of the Norwegian national budget through programs such as unemployment benefits, age pensions, and sick leave benefits. NAV forms the backbone of the Norwegian welfare state, and most Norwegian citizens have contact with the agency at some point.

The NAV IT department develops, maintains, and operates the information systems that support the organization. The IT department has approximately 700 employees and 400 consultants, and maintains and operates close to 300 applications. The application portfolio is made up of several generations of solutions; from mainframe systems to newer web-oriented applications, as well as standard systems that support operations such as accounting, payroll, and document production. Efforts to increase efficiency and automation have made applications increasingly interconnected. Historically, these dependencies have been addressed through centralized control and staged deliveries, and software development has been outsourced to external suppliers. However, although the number of errors has been reduced, the delivery method has proven inflexible and expensive.

Therefore, in 2017, the organization changed its development and sourcing strategy. To increase flexibility and reduce costs, NAV moved from centralized control and outsourcing of software development to internal ownership and a distributed governance model. This study focuses on how this transition was enabled by changes in the digital infrastructure and organizational structure through the process of platformization.

4.2 Data collection

Most data were collected by the first author between March 2017 and May 2019. We used three methods to collect data: interviews, participant observations, and document analysis. First, we conducted a total of 38 interviews (see Table 2). Of these interviews, 23 were recorded and transcribed. The interviews lasted between 45 and 60 minutes. Because of their sensitive nature, not all interviews could be recorded. In these cases, we took notes and added more extensive remarks shortly after the interviews ended.

Informants were chosen from across the IT department based on a snowballing strategy, where one informant suggested the next. Most informants worked on a large ongoing modernization project (the Parental Benefit project) that developed software

Roles	*IT architecture section*	*Parental Benefit project*	*Other sections*	*Total*
Senior executives	1	2	3	6
Project managers		4		4
Team leads		4	2	6
Developers		8	4	12
Architects	3	4	1	8
Case workers			2	2
Total	4	22	12	38

Table 2. Overview of interviews

support for improved management of parental benefits. Parental benefit is a welfare benefit intended to compensate parents for loss of income in relation to the birth or adoption of a child. Parental benefits payments were administered on an IBM mainframe solution introduced in 1978, and the project was a step toward replacing the old system with more flexible and integrated systems support. With an estimated cost of close to 1.3 billion NOK, the Parental Benefit project was the largest ongoing development project within the Norwegian public sector at the time of data collection and provided unique insight into the challenges of replacing legacy systems. The project also gave us insight into the transition from staged to continuous development practices in large-scale organizational settings. In addition to the Parental Benefit project, several informants worked in the section for the IT architecture section. The IT architecture function in NAV has traditionally been responsible for ensuring consistency and sustainability across projects and has had considerable influence on decisions related to technology and architecture. With the transition from a centralized to a distributed decision model, the IT architects were assigned to development teams, consequently losing some of their influence over technology and architecture decisions.

Another important data source was participant observations. The first author was able to move freely within the IT department. She could also attend most meetings and social gatherings. Considerable insight was gained through informal conversations by the coffee machine and encounters in the hallway. Observations and conversations were documented extensively in a field diary.

The data also included a significant volume of documents collected from internal and external websites and archiving systems. Among these documents were government white papers, procurement documents, design specifications, project reports, and project websites. The first author was given an e-mail address and had access to most internal documents, including calendars, project wikis (Confluence), and issue tracking systems (JIRA). In addition, online conference presentations by NAV employees were transcribed and analyzed. The transformation of NAV has attracted considerable attention in the software development community, and NAV has presented the change process at a series of practitioner conferences in Norway.

4.3 Data analysis

Data analysis was iterative and overlapped with data collection, thus granting flexibility to respond to emergent themes (Eisenhardt 1989). Specifically, the data analysis can be described as an iterative four-step process. First, we deductively started from a platformization framework derived from the platform literature. This framework indicated that platformization consisted of two separate but related processes: One involved dismantling legacy systems (Bygstad and Hanseth 2018), and the other involved an altered organizing logic (Yoo et al. 2010). We labeled the first process 'decoupling,' and the second 'recoupling,' where decoupling referred to the dismantling of systems, and recoupling referred to the associated organizational changes.

Second, we used an open coding procedure to discover concepts and their properties and dimensions (Charmaz 2014). We manually coded the data using colors and annotations. We developed descriptive codes capturing the informants' views and reflections on the organizational and technological changes leading up to the transition. Following the actor-centric principle of interpretive research, we identified what was perceived as a problem, by whom it was perceived as a problem, and solutions proposed. For instance, an informant described the problems involved in operating several technical platforms simultaneously. He suggested that these problems had been solved through standardization on a single application platform, separating the application layer from its underlying technical resources. From this, we arrived at the code 'Separating technical infrastructure from the presentation layer.' Similarly, an informant described how tightly coupled systems required extensive coordination between projects. This was, according to the informant, addressed by dismantling legacy systems into modular components that could be developed and managed independently. From this, we derived the code 'Legacy systems are dismantled into smaller components.'

Third, we merged the descriptive codes with the theoretical constructs derived in the first step of the analysis by identifying empirical examples of decoupling and recoupling, thus describing how these processes unfolded in the organization. Through discussions with colleagues, we were made aware of the similarity between our conceptualization of decoupling and recoupling and the concepts resource liquefaction and resource integration found in service-dominant logic (Lusch and Nambisan 2015). This insight was fed into the analysis, thus triggering an understanding of decoupling as a process of picking apart, and recoupling as a process of putting the pieces back together. A mapping between theoretical constructs (decoupling and recoupling) and their associated descriptive codes is given in Table 3.

5 Results

In the following, we present the results of the study. We begin by describing the process of decoupling, before describing how recoupling unfolded in the study. Figure 1 provides a timeline of the most important events identified in relation to each process.

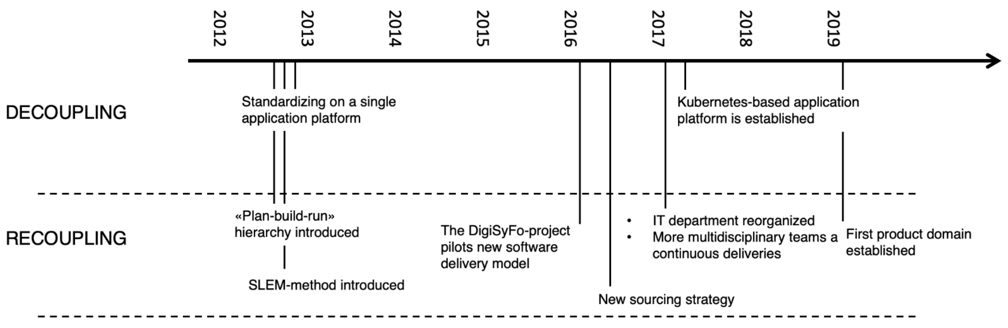

2012
2013
2014
2015
2016
2017
2018
2019
DECOUPLING
Standardizing on a single application platform
Kubernetes-based application platform is established
RECOUPLING
«Plan-build-run» hierarchy introduced
SLEM-method introduced
The DigiSyFo-project pilots new software delivery model
New sourcing strategy
• IT department reorganized
• More multidisciplinary teams a continuous deliveries
First product domain established

Figure 1. Events relevant to the platformization processes in NAV.

Constructs	*Descriptive code*	*Excerpts*
Decoupling	Decoupling between technical infrastructure and presentation layer	"We abstracted away some of the complexity by saying that if you are going to build an application, then you should not have to think about which operating system and network protocols to use, and low-level technical things." (Architect IT, section for IT architecture)
	Decoupling of legacy systems into smaller components	"Systems should be broken up into products then that can be developed, tested and operated separately. Rather than one big lump, you need several separate components. In this way, you avoid all the coordination which takes a lot of time. If my component is dependent on your component, we have to make sure that your component is released before my components and so on. This is very demanding." (Software developer, Parental benefit project)
Recoupling	Establishing independent teams	"This project aimed to develop digital solutions for sick leave services targeting employees, employers, health personnel and NAV employees managing the services. The project was organized in two teams and it was decided to give the teams 'autonomy'; defined as giving the team members freedom to work in close cooperation with product owners and users to decide the scope of the work, to take ownership of their processes and practices, and to take responsibility for the interfaces with other systems in a proactive manner." (Team member from the first independent team).
	Establishing product domains	"The idea behind domain-driven design is that it is much more important to organize for flow: Optimal flow in your organization and in your code. And you do this by minimizing coordination—chattiness across [the organization]. You want to cluster things that naturally belong together and keep things that do not belong together apart. You cluster people and code into domains and domain areas. Then you get a closer connection between people within a domain." (Senior executive)

Table 3. Examples of theoretical constructs and associated codes

5.1 Decoupling legacy systems (2012-present)

The digital infrastructure was decoupled in two stages: First, the organization was standardized on hardware and middleware through the introduction of a common application platform. Second, legacy systems were rewritten as modular applications.

First decoupling phase: Establishing a platform core. The first phase began in 2012 when NAV introduced the agency's first application platform. Previously, each project had been responsible for specifying and establishing its own technical infrastructure, resulting in a multitude of different technological platforms. Technical heterogeneity, combined with manual deployment and provisioning, made application management error-prone and time-consuming. To reduce error rates and increase efficiency, NAV decided to standardize on a single platform and streamline operations management. The application platform, which was named 'the Cloud,' was based on virtual server technology, where JBoss and WebSphere middleware ran on a Red Hat Linux operating system. Software development projects were instructed to use the platform, thus reducing heterogeneity and simplifying deployment and provisioning. The introduction of virtual server technology reduced hardware costs, and semi-automated provisioning reduced the setup time from weeks to minutes. By standardizing on a single application platform, projects no longer had to spend resources on choosing and setting up technical infrastructure.

> Let me give you an example. In the first project I worked in - it must have been in 2006, we used a lot of time, not only to decide which application platform to use but also choosing the operating system it would run on. A project destined to build a case tool would spend a lot of time deciding whether the application should run on Windows or Linux, or if it should run on a mainframe solution. We spent a lot of time doing this. And projects don't spend time doing this anymore. It's been standardized. Everyone just has to relate to it (Architect in IT architecture section).

Most systems were migrated to the application platform, and by the end of 2015, the plethora of hardware and middleware technologies had been reduced to three technical platforms: 1) the application platform (the Cloud); 2) Arena, an Oracle forms-based system for managing follow-up activities related to employment; and 3) InfoTrygd, an IBM mainframe system from 1978 used to manage individual benefits.

However, although the platform abstracted the underlying complexity and simplified provisioning and deployment, the applications were still large and interconnected.

To manage this complexity, the organization centralized operations and banned projects from releasing their own applications. Deployments were restricted to four yearly releases, where all changes had to be approved and deployed by the Operations department. A single release could include 80 000 development hours.

> In 2014, we had a major delivery of approximately 80 000 development hours. [...] it's the biggest we've had. Most of them are around 40-50 000 (Project manager, NAV).

Coordinated releases and centralized control reduced the number of errors, but it could be months from the time a feature was developed until it was available to users and development teams. It also increased complexity as developers were forced to anticipate needs, and "think about everything" upfront.

Second decoupling phase: Establishing a variable periphery of applications. To reduce the complexity of the digital infrastructure, in 2017 NAV began to decouple monolithic systems into modular applications. To facilitate the process, the organization introduced a second application platform. The application platform was based on Kubernetes, an open-source framework for managing software containers. In this way, NAV was able to design its own platform, offering only a subset of the functionality available on the Kubernetes platform. A software container has its own filesystem, CPU, memory, and process space, and as they are decoupled from the underlying infrastructure, they are portable across clouds and operating systems. By containerizing applications, they could easily be migrated between platforms.

As part of the migration process, large systems were also broken down into smaller applications— thus reducing complexity. The modularized architecture made it possible to deploy and manage applications independently. This was also one of the main goals for introducing the new application platform: to introduce a distributed governance model where each team could deploy and manage their own applications. In this way, applications could be developed and released at the discretion of the team, thus increasing the rate of change. With the new governance model, release rates were increased from once every few months to several times a day. As release rates increased, so did the feedback and development speed. One manager described the transition from the old to the new structure as "taking a super-tanker and splitting it up into 100 speedboats" (Section manager in the IT department).

An important prerequisite for the distributed governance was automation of operations management: The platform provided fully automated services for tasks such as provisioning, deployment, and load balancing. This meant that applications could be managed without expert knowledge in operations management, enabling teams to take responsibility for their own applications. A team lead for the platform team stated:

> Kubernetes is the open-source framework that comes from Google. It is all of Google's experience over the last 15 years with how to manage infrastructure rewritten by the same people. It's like taking the world's best operations person and fully automating him. That's what Kubernetes is. It provides a lot of tools for running in production, which makes it more robust, and more scalable and everything.

To ensure that the transition from centralized to decentralized governance did not result in chaos, NAV introduced the concept 'white-listing.' Applications were white-listed if they were sufficiently low in complexity and had few external dependencies. Only white-listed applications could be managed by the development teams. Complex applications still had to be managed by the Operations department.

5.2 Recoupling the organization (2016-present)

The recoupling of NAV can be seen to progress in two overlapping stages: In the first stage (2016 and onward), NAV introduced a distributed service delivery model, where multidisciplinary teams began to develop and maintain their own applications. In the second stage, teams and applications were recombined into product domains.

First recoupling phase: Establishing multidisciplinary teams. In 2016, NAV initiated a project that would pilot a new software delivery model. The project, which we called DigiSyFO (short in Norwegian for Digital sick leave follow-up), was responsible for digitizing follow-up activities related to sick leave. The project had two goals: First, the project would digitize follow-up, transferring communication between the employer, general practitioners, and NAV from analogue to digital media. Second, the project would pilot a new software delivery model, where staged deliveries and centralized control were replaced by multidisciplinary teams that were responsible for the entire software delivery cycle. Software would be developed and released continuously. Whereas other projects employed a staged delivery model with a handover between stages, the Digital Health project took responsibility for all parts of the development cycle, interleaving activities such as design, development, and operations.

> The project was initially set up with a single team with less than ten members consisting of two UX designers, two subject matter experts, one software architect, one developer, operations support, and agile coaches. This grew to approximately 22 project members nine months into the project (Agile coach in the IT department).

Although most of the developers were hired consultants, the project was planned and managed by NAV employees. This differed from other projects, where software development and maintenance were outsourced to consultant companies through so-called responsibility contracts. The project was deemed a huge success, receiving the government's digitalization reward based on outstanding achievement. The chairman of the jury summarized the project as follows:

> This year's winners are a good example of how things can be done in new ways and deliver services that streamline public management and make life easier for users. Both the government and Difi [short in Norwegian for Agency for Public Management and eGovernment] have high expectations for digitalization of the public sector (Director at the Norwegian Digitalization Agency).

Following the success of the Digital Health project, NAV decided to implement the new delivery strategy across the organization. This required a series of changes: First, the agency changed the sourcing strategy, moving from outsourcing to insourcing of software development. As responsibility contracts expired, they were replaced by capacity contracts where consultants were hired by the hour.

> Development has been outsourced to consultant companies through traditional management contracts where the supplier has had independent responsibility for the system and maintenance. Now, NAV is taking over this. We have a different sourcing model where we hire people depending on our capacity instead of giving the supplier total responsibility (Agile coach in the IT department).

Second, the IT department was reorganized, replacing the hierarchical organization with a matrix structure. Whereas employees in the old structure were organized according to their role in the deployment cycle (planning, development, or operations), in the new structure, employees were grouped according to their competence field.

Third, software was developed by multidisciplinary teams that assumed responsibility for the entire software development cycle. In this way, development teams were able to learn from their actions and improve accordingly.

Fourth, the organization began to employ developers. Until this point, NAV had had few software developers of its own. Instead, the IT department was staffed to control suppliers and supervise projects. Development and maintenance of the NAV core systems had been outsourced to 15-20 different consultant companies, each pursuing its own goals and targets.

> We [NAV] had decided not to code ourselves. So, there we were, with 200-250 person-years, supervising 300-400 consultants. We were busy specifying needs and running tendering competitions and stuff like that, and sometimes, the needs proved too expensive. Then, we had to go back and clarify some more needs. So, there we were, managing and coordinating, being an intermediary [between the business side of NAV and the consultant company]. With roles that made make sure that the system was technically sound and things like that. But of course, it is quite difficult to control 15-20 suppliers, who all have their own interests and needs. How do you control them across 300 systems? How do you ensure the quality of the code base? Well, you can't. That's impossible (Manager in the IT department).

With the strategic shift, NAV gradually assumed responsibility for developing and maintaining the core systems. The goal was to increase the in-house development capacity and use consultants only during peak periods and to provide expert knowledge. The focus changed from controlling subcontractors to developing and maintaining the systems.

Second recoupling phase: Establishing product domains. As the applications became increasingly decoupled, the development teams could work more independently. However, many applications were part of larger value paths, and had to be developed in close relation to other systems and teams. A developer for the Parental Benefit project stated:

> In my experience, when people don't see the value chain they're contributing to, they begin to sub-optimize. It is artificial to say that you have a truly autonomous team in the area of Parental Benefit. All components and all applications support the same decision process.

To address dependencies, applications were grouped into functional domains, where a domain would contain several applications and teams.

> The idea behind domain-driven design is that it is more important to organize for flow. Optimal flow in the organization and in the code. And this is done by minimizing coordination and chattiness. You must bundle the things that belong together and keep the things that don't belong together apart. You bundle people and code in domains. People within a domain will be more closely linked (Manager in IT department).

By focusing on domains, NAV was able to establish value paths that transcended the organization. However, the domains had to be established gradually, one at a time. In this way, they could learn from the experiences from one domain before establishing the next.

> The goal is to remove boundaries between departments. But you can't do that by changing everything at once. All 19 000 employees. Because that won't work. Instead, you can do what we are doing. Establish one domain at a time. Gain experience and prove to the organization that it works. Then you establish the next domain (Senior executive in the Parental Benefit project).

As a principle, no legacy systems were brought into the domains. Domain teams would manage only new applications. In practice, NAV found that they had to make exceptions to this rule. In some cases, it was more practical to give domain teams responsibility for legacy systems while they were being dismantled and reimplemented.

By creating cross-cutting domains, NAV was able to improve flow and collaboration, without physically reorganizing employees. And by establishing one domain at a time, NAV could learn and adapt, gradually restructuring the organization and redesigning the service delivery. At the time of writing (September 2019), the organization had established three domains.

6 Discussion

The purpose of this study was to provide insight into how platformization unfolds within organizations. Platformization is often described as a process of establishing core services and an ecosystem of complementors (Benlian et al. 2018; Bygstad and Hanseth 2018; Cusumano 2010). However, we take a broader view, and draw on the concepts

Concepts	*Scope of theorizing*	*Manifestation in the case*
Decoupling	Decoupling legacy systems into a platform-oriented structure	Monolithic systems are gradually reimplemented as modular applications through application of microservices
	Decoupling information from its physical media (Liquefaction)	As systems are reimplemented, a growing number of manual operations are automated—liquefying data through the Kubernetes platform
Recoupling	Recombining knowledge and skills into new organizational forms and practices	Knowledge and skills are recombined through the formation of multidisciplinary teams
	Recombining applications and teams into domains	Applications and teams with stronger dependencies are recombined into product domains
Interaction of decoupling and recoupling	Cyclic interaction of decoupling and recoupling	Decoupling provides a stable platform for new combinations of knowledge and skills to emerge. In turn, recoupling provides new flexible organizational forms and practices for changing existing infrastructure into a platform architecture

Table 1. Platformization theorized.

of decoupling and recoupling to theorize how digital platforms enable new ways of organizing and delivering services in an interorganizational setting characterized by monolithic legacy systems and hierarchical structures. Empirically, we conducted a longitudinal case study of a platformization process in a Norwegian government organization, NAV, covering a period of seven years. Our detailed account of the platformization process in the case uncovers the way in which digital platforms pave the way for a reorganization of service delivery where outsourcing of development activities and a staged delivery model are replaced by continuous software development and cross-organizational collaboration. In the following, we draw on these empirical insights (as summarized in Table 4) and on extant literature to advance theory about platformization. In so doing, this paper provides three distinct contributions. Our first contribution is our theorization of how platformization involves processes of decoupling that can transform legacy systems into a modular platform architecture. In this way, decoupling does not merely provide a platform in addition to or on top of the existing digital infrastructure as presented in extant literature (Bygstad and Hanseth 2018; Islind et al. 2016), but instead, transforms the digital infrastructure into a working platform. Our second contribution is related to how platformization through processes of recoupling affords recombination of knowledge and skills into new organizational forms and practices. This gives an organization the capability to change infrastructures flexibly. The third contribution is our proposal that platformization requires cyclic interaction between processes of decoupling and recoupling in the sense that decoupling provides increased stability (i.e., a platform core) for new knowledge and skills to emerge, and in turn, recoupling implies increased flexibility and competence for change. Henceforth, the processes of decoupling and recoupling, although distinct, feed each other cyclically, so that if organizations have one without the other, they will not be able to implement platformization.

We begin by discussing the concept of decoupling before recoupling. Finally, we discuss how the concepts of decoupling and recoupling interact over time.

6.1 Decoupling: transforming legacy IS into platforms

By emphasizing decoupling as a strategy for renewing technology, we contribute to existing literature by suggesting that digital platforms provide a means for *replacing* legacy systems, thus addressing issues related to innovation along the periphery and at the core. Viewing platformization as a strategy for replacing legacy systems complements studies that see platformization either as a process of establishing platforms from scratch (Benlian et al. 2018; Islind et al. 2016) or as a process of masking legacy systems behind applications' programmable interfaces in an effort to increase innovation at the periphery (Bygstad and Hanseth 2018).

Decoupling corresponds to Islind et al.'s (2016) description of platformization as the process of establishing a digital platform. However, whereas Islind et al. describe platformization as the process of establishing a platform alongside existing infrastructure, we see decoupling as a process of establishing a platform structure across the existing infrastructure. Therefore, Islind et al. describe a case of green-field development, where the platform is established de neuvo—not having to relate to existing systems and practices. In contrast, the present study is of brown-field development, where the platform is tailored to the existing infrastructure. Our observations correspond to decoupling as described by Benlian et al. (2018), who identify it as unfolding on the infrastructure and application levels. Benlian et al. see decoupling as a general trend and a consequence of cloud computing. However, we take a more specific approach and see it as a strategy of technological renewal, enabling the gradual introduction of platform-oriented logic in existing infrastructures. This difference is important, because although the efficiency and transformative impact of platform structures have been proven through successful implementations, such as Apple (Eaton et al. 2015), large-scale organizations with a legacy of existing systems and practices are finding it hard to transition from tightly coupled architectures. In this vein, our conception of decoupling provides a strategy for gradually and continuously renewing technology which is applicable to even the most entrenched infrastructures. At NAV, decoupling progressed in two steps. In the first step, application systems were decoupled from their underlying digital resources through the use of virtual servers. Virtualization technologies provided increased elasticity as resources could be dynamically scaled up and down (Benlian et al. 2018). In the second step, legacy systems were decoupled into smaller applications. The container platform enabled reuse of third-party services and facilitated decoupling through the use of web protocols (i.e., SOAP and REST).

The use of platformization as a means for increased flexibility and innovation in digital infrastructures has been addressed by Bygstad and Hanseth (2018). In a multilevel study of a large e-health initiative, they examine the way in which a layered architecture, where legacy systems are encapsulated in the platform core, creates a platform-oriented infrastructure. Bygstad and Hanseth describe a process where legacy systems are encapsulated and hidden from application developers. However, we describe a process where legacy systems are gradually replaced. Rather than seeing these differing strategies as different and opposing, they should be seen as complementary and potentially mutually enabling, where the encapsulation of legacy systems might be seen as an aid, or a preliminary stage on the way toward renewing and replacing legacy systems. Although the approach proposed by Bygstad and Hanseth increases external innovation (offering complementors services which enable innovation), it leaves legacy systems mostly unchanged. Therefore, the strategy fails to improve the maintainability and evolvability of the legacy systems, which was seen as the main goal of the decoupling process as it unfolded at NAV. Decoupling legacy systems provided the agency with a set of modular components that could be combined and recombined into new products and services (Henfridsson et al. 2018), thus increasing resource density and the probability of innovation (Lusch and Nambisan 2015). The modular structure enabled the reallocation of applications to the most suitable actor in the organization (Normann 2001). In practice, with a large legacy of aging systems, it might not be realistic or desirable to replace all systems. Therefore, renewal be achieved through a combined strategy where some systems are hidden while others are replaced.

6.2 Recoupling: enabling new forms of organizing

In this study, we found that decoupling the legacy systems paved the way for an alternative method of organizing: Centralized control was replaced by a distributed structure where independent development teams were responsible for developing and managing applications. Teams were composed of technology experts and business domain representatives. This enabled innovation through the recombination of skills and knowledge across departments and subject domains (Brown and Duguid 1991). Thus, we propose that the process of recoupling allows for recombining knowledge and skills that enable flexible change in organizations.

At NAV, as most applications belonged to larger value chains, teams that were responsible for developing related functionality were combined into service domains. In this way, decoupling the digital infrastructure enabled the formation of a relatively self-contained, self-adjusting system of loosely coupled actors (Lusch and Nambisan 2015), where actors contributing to common value chains were more closely connected than actors contributing to different value chains (Evans 2006). Teams and domains transcended the matrix organization in a fluid structure, where employees could be dynamically combined and recombined in response to emergent needs (Ciborra 1996; Schreyögg and Sydow 2010).

These findings complement the platform literature by highlighting the way in which digital platforms enable restructuring of organizations through the dynamic recombination of teams and applications. By simultaneously opening the black boxes of technology and organization (Zammuto et al. 2007), we move beyond an inter-organizational perspective (Eaton et al. 2015; Yoo et al. 2010) and explore the way in which digital platforms facilitate innovation *within* organizations. From this intra-organizational view, digital platforms enable the recombination of not only digital resources (Henfridsson et al. 2018) but also knowledge and skills across sections and departments, further increasing the agency's potential for innovation.

The teams were composed of business and technology experts, providing the teams with the skills and knowledge to develop and manage applications independently. Combined with a modular platform where applications could be deployed in isolation, the teams were able to assume responsibility for all parts of the development process—from the inception of an idea until a service was eventually turned off. In this way, the organization was able to move from staged development, where different departments were responsible for different parts of the development process, to a continuous process in which one team was responsible for the entire software development cycle (Fitzgerald and Stol 2017). By shifting from the staged development practice to a continuous and network-oriented approach, NAV facilitated innovation across teams (Lusch and Nambisan 2015). These findings confirm insights from the software engineering literature where practices and continuous development are seen to enhance innovation through feedback and learning (Fitzgerald and Stol 2017). We complement these insights by exploring the way in which digital platforms can enable continuous development practices on a large scale. Practices of continuous development enabled feedback from users to be recombined into future versions of the service—thus adding an additional stakeholder in the recoupling process.

Although the concept of recombination is not new (Henfridsson et al. 2018; Lusch and Nambisan 2015), this research contributes by highlighting the interrelation between social organization and technical infrastructure, simultaneously opening the back boxes of technology and organization (Zammuto et al. 2007). As the teams are formed across existing structures, organizational recoupling is less prone to the knowledge disruption associated with structural recombination (Karim and Kaul 2015). However, at NAV, the transition from centralized to distributed control inferred a considerable shift in the power structures—where decision authority was transferred from centralized and coordinating roles to development teams. Therefore, the transition faced considerable resistance from parts of the organization, requiring coordinated efforts and persuasion at all levels of the organization. However, these issues are beyond the scope of this research.

6.3 Interaction between decoupling and recoupling

Grounded in this research, we theorize that processes of decoupling and recoupling interact cyclically, where decoupling increases the potential for recoupling and vice versa. Specifically, the analysis revealed that the process of decoupling provided new ways of organizing the development of information systems—thus increasing the organization's capacity to recouple. Similarly, the recoupling of the organization produced new organizational capabilities for renewing information systems, which, in turn, facilitated further decoupling (see Figure 2).

Consistent with previous research, this study showed that the decoupling of the digital infrastructure enabled a recombination of services (Benlian et al. 2018) and the introduction of an alternative organizational logic (Yoo et al. 2010). This research complements these studies by emphasizing platformization as an emergent phenomenon,

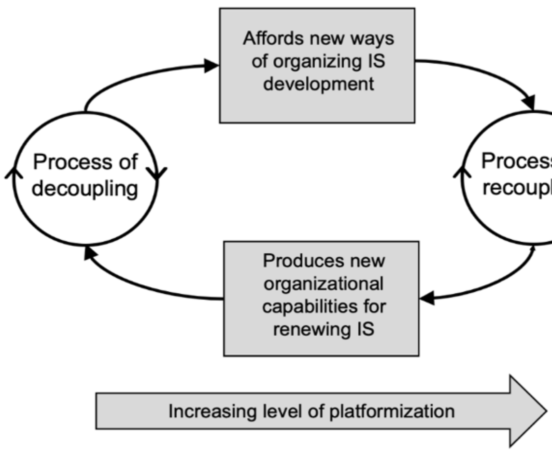

Affords new ways of organizing IS development
Process of decoupling
Process of recoupling
Produces new organizational capabilities for renewing IS
Increasing level of platformization

Figure 2. Interaction between decoupling and recoupling over time.

exploring the way in which the hierarchical organization is gradually and incrementally replaced by a distributed model.

NAV began by establishing one domain, and based on these experiences, continued to establish others. This incremental approach also faced resistance, as a proven track record (where independent development teams outperformed traditional project deliveries in terms of efficiency and flexibility) was a powerful and convincing argument in discussions with sceptics. In this manner, the incremental approach reduced resistance and increased the likelihood of a successful transition. With this incremental approach, the platformization process became a process of continuous organizational improvement, where feedback from one cycle was fed into subsequent cycles, allowing for a gradual and knowledge-based transformation of the organization.

7 Conclusion

This research presented a longitudinal study of the platformization of a large digital infrastructure. We found that through the decoupling of the digital infrastructure, the organization was able to dynamically recouple software components, skills, and knowledge into new and innovative services. We also found that the platformization unfolded cyclically, where the decoupling of applications enabled the recoupling of the organization, and the recoupling of the organization enabled further decoupling of the infrastructure.

As this research is based on a single case study, it is inevitably subject to several limitations. For example, the results cannot be validated across cases (Yin 2013). In addition, change processes of the magnitude described in this case study take a long time.

Although the transformation of NAV seemed successful, a two-year study is too short to draw any decisive conclusions. Thus, additional fieldwork is needed.

We observed the change process as it was unfolding—following events over a two-year period. Therefore, the findings suggest a path but are not conclusive regarding the outcome of the change process. Further studies are needed to provide detailed insights into the precise shortcomings and advantages of the decoupling process, where decoupling is investigated in a later stage of evolution.

Acknowledgment

We would like to thank NAV, and especially Petter Hafskjold and Ida Kristine Dørum for answering our questions and granting us access to the sites. We would also like to thank all informants for taking the time to talk to use. In addition, we want to acknowledge Eric Monteiro for providing extensive and valuable comments throughout the writing process.

This work was supported by the project Agile 2.0 supported by the Research Council of Norway through grant 236759 and by the companies DNV GL, Equinor, Kantega, Kongsberg Defence and Aerospace, Sopra Steria and Sticos.

References

Aanestad, M., and Jensen, T. B., (2011). Building nation-wide information infrastructures in healthcare through modular implementation strategies. *The Journal of Strategic Information Systems*, (20:2): 161-176.

Baldwin, C. Y., and Woodard, C. J., (2009). The architecture of platforms: A unified view. *Platforms, markets and innovation*): 19-44.

Battleson, D. A., West, B. C., Kim, J., Ramesh, B., and Robinson, P. S., (2016). Achieving dynamic capabilities with cloud computing: an empirical investigation. *European Journal of Information Systems*, (25:3): 209-230.

Benlian, A., Kettinger, W. J., Sunyaev, A., and Winkler, T. J., (2018). Special Section: The Transformative Value of Cloud Computing: A Decoupling, Platformization, and Recombination Theoretical Framework. *Journal of Management Information Systems*, (35:3): 719-739.

Boudreau, K., (2010). Open Platform Strategies and Innovation: Granting Access vs. Devolving Control. *Management Science*, (56:10): 1849-1872.

Brown, J. S., and Duguid, P., (1991). Organizational learning and communities-of-practice: Toward a unified view of working, learning, and innovation. *Organization science*, (2:1): 40-57.

Bygstad, B., and Hanseth, O., (2018). TRANSFORMING DIGITAL INFRASTRUCTURES THROUGH PLATFORMIZATION.).

Cennamo, C., and Santaló, J., (2019). Generativity tension and value creation in platform ecosystems. *Organization Science*, (30:3): 617-641.

Charmaz, K., (2014). *Constructing grounded theory*, Sage.

Ciborra, C., (2000). *From control to drift: the dynamics of corporate information infastructures*, Oxford University Press on Demand.

Ciborra, C. U., (1996). The platform organization: Recombining strategies, structures, and surprises. *Organization science*, (7:2): 103-118.

Constantinides, P., Henfridsson, O., and Parker, G. G. (2018). Introduction—Platforms and Infrastructures in the Digital Age: INFORMS.

Cusumano, M., (2010). Cloud computing and SaaS as new computing platforms. *Communications of the ACM*, (53:4): 27-29.

de Reuver, M., Sørensen, C., and Basole, R. C., (2017). The digital platform: a research agenda. *Journal of Information Technology*).

Eaton, B., Elaluf-Calderwood, S., Sorensen, C., and Yoo, Y., (2015). Distributed tuning of boundary resources: the case of Apple's iOS service system. *MIS Quarterly: Management Information Systems*, (39:1): 217-243.

Eisenhardt, K. M., (1989). Building theories from case study research. *Academy of management review*, (14:4): 532-550.

Evans, E., (2004). *Domain-driven design: tackling complexity in the heart of software*, Addison-Wesley Professional.

Fitzgerald, B., and Stol, K.-J., (2017). Continuous software engineering: A roadmap and agenda. *Journal of Systems and Software*, (123): 176-189.

Gawer, A., (2014). Bridging differing perspectives on technological platforms: Toward an integrative framework. *Research Policy*, (43:7): 1239-1249.

Gawer, A., and Cusumano, M. A., (2014). Industry platforms and ecosystem innovation. *Journal of Product Innovation Management*, (31:3): 417-433.

Ghazawneh, A., and Henfridsson, O., (2013). Balancing platform control and external contribution in third-party development: the boundary resources model. *Information Systems Journal*, (23:2): 173-192.

Hanseth, O., (2001). Gateways—just as important as standards: How the internet won the "religious war" over standards in Scandinavia. *Knowledge, technology & policy*, (14:3): 71-89.

Hanseth, O., and Lyytinen, K., (2010). Design theory for dynamic complexity in information infrastructures: the case of building internet. *Journal of Information Technology*, (25:1): 1-19.

Helmond, A., (2015). The platformization of the web: Making web data platform ready. *Social Media+ Society*, (1:2): 2056305115603080.

Henfridsson, O., and Bygstad, B., (2013). The generative mechanisms of digital infrastructure evolution. *MIS quarterly*, (37:3).

Henfridsson, O., Nandhakumar, J., Scarbrough, H., and Panourgias, N., (2018). Recombination in the open-ended value landscape of digital innovation. *Information and Organization*, (28:2): 89-100.

Islind, A. S., Lindroth, T., Snis, U. L., and Sørensen, C., (2016). Co-creation and fine-tuning of boundary resources in small-scale platformization, *Scandinavian conference on information systems*, Springer, pp. 149-162.

Jacobides, M. G., Cennamo, C., and Gawer, A., (2018). Towards a theory of ecosystems. *Strategic Management Journal*).

Kallinikos, J., Aaltonen, A., and Marton, A., (2013). The ambivalent ontology of digital artifacts. *Mis Quarterly*): 357-370.

Karim, S., and Kaul, A., (2015). Structural recombination and innovation: Unlocking intraorganizational knowledge synergy through structural change. *Organization Science*, (26:2): 439-455.

Kornberger, M., Pflueger, D., and Mouritsen, J., (2017). Evaluative infrastructures: Accounting for platform organization. *Accounting, Organizations and Society*, (60): 79-95.

Lusch, R. F., and Nambisan, S., (2015). Service innovation: A service-dominant logic perspective. *MIS quarterly*, (39:1).

Normann, R., (2001). *Reframing business: When the map changes the landscape*, John Wiley & Sons.

Parker, G. G., Van Alstyne, M. W., and Choudary, S. P., (2016). *Platform Revolution: How Networked Markets Are Transforming the Economyand How to Make Them Work for You*, WW Norton & Company.

Poell, T., Nieborg, D., and van Dijck, J., (2019). Platformisation. *Internet Policy Review*, (8:4): 1-13.

Rolland, K. H. M., Lars; Rai, Arun, (2018). Managing digital platforms in user organizations: The interactions between digital options and digital debt. *Information Systems Research*, (29:2): 419-443.

Schreyögg, G., and Sydow, J., (2010). Crossroads—organizing for fluidity? Dilemmas of new organizational forms. *Organization science*, (21:6): 1251-1262.

Spagnoletti, P., Resca, A., and Lee, G., (2015). A design theory for digital platforms supporting online communities: a multiple case study. *Journal of Information technology*, (30:4): 364-380.

Tan, H., and Wright, J., (2018). A Price Theory of Multi-Sided Platforms: Comment. *American Economic Review*, (108:9): 2758-2760.

Tiwana, A., (2013). *Platform ecosystems: aligning architecture, governance, and strategy*, Newnes.

Tiwana, A., Konsynski, B., and Bush, A. A., (2010). Research commentary—Platform evolution: Coevolution of platform architecture, governance, and environmental dynamics. *Information Systems Research*, (21:4): 675-687.

Törmer, R. L., and Henningsson, S., (2018). From Drift To Central Guidance: A Path Constitution Perspective On The Platformization Of An Information Infrastructure, *2018 European Conference on Information Systems*, Association for Information Systems. AIS Electronic Library (AISeL).

Vassilakopoulou, P., Grisot, M., Blegind Jensen, T., Sellberg, N., Eltes, J., Thorseng, A. A., and Aanestad, M., (2017). Building national ehealth platforms: the challenge of inclusiveness, *Proceedings of the International Conference on Information Systems-Transforming Society with Digital Innovation, ICIS 2017*.

Wareham, J., Fox, P. B., and Cano Giner, J. L., (2014). Technology ecosystem governance. *Organization Science*, (25:4): 1195-1215.

Yin, R. K., (2013). *Case study research: Design and methods*, Sage publications.

Yoo, Y., Boland Jr, R. J., Lyytinen, K., and Majchrzak, A., (2012). Organizing for innovation in the digitized world. *Organization science*, (23:5): 1398-1408.

Yoo, Y., Henfridsson, O., and Lyytinen, K., (2010). Research commentary—the new organizing logic of digital innovation: an agenda for information systems research. *Information systems research*, (21:4): 724-735.

Zammuto, R. F., Griffith, T. L., Majchrzak, A., Dougherty, D. J., and Faraj, S., (2007). Information technology and the changing fabric of organization. *Organization science*, (18:5): 749-762.

PAPER V

Using digital platforms to promote a service-oriented logic in public sector organizations: A case study

Kathrine Vestues
Norwegian University of Science and Technology (NTNU)
kathrine.vestues@ntnu.no

Marius Mikalsen
SINTEF, Trondheim, Norway.
Norwegian University of Science and Technology (NTNU)
marius.mikalsen@sintef.no

Eric Monteiro
Norwegian University of Science and Technology (NTNU)
eric.monteiro@ntnu.no

Abstract

A growing number of scholars and practitioners have recognized that value is defined and cocreated by citizens and that citizens must be involved in the service delivery process to improve the quality and efficacy of public services. Central to this service-dominant logic is that public sector organizations cannot manufacture value for citizens; they can only make a value proposition that the citizen might choose to use. Hence, value must be cocreated. However, this cocreation entails accommodating cocreation practices with millions of users. Currently, cocreation is often limited to involving a carefully selected set of users in crafting requirements early and/or measuring user satisfaction upon service launch. There is an empirical blindspot in the current literature in terms of how to shape service delivery in a way that is capable of effectively capturing emergent and process-oriented value cocreation across large user groups. Through a longitudinal case study of the IT department at the Norwegian Labour and Welfare Administration (NAV), which provides services to millions of users, this paper explores how digital platforms are used to transform value cocreation into a process of continuous improvement. We find that adopting a process-oriented approach for cocreation within public sector organizations requires structural changes, including sourcing strategy and governance structure. We also show the importance of digital platforms in increasing the efficiency of cocreation. We discuss how these structural changes were made and the role played by digital platforms in achieving these changes.

1. Introduction

Public sector organizations are under strong and increasing pressure to improve their service delivery. In particular, issues have been raised about inadequate response to emergent demands [1, 2] and lack of citizen involvement [3].

This calls for a transformation of public sector organizations where they become more attuned to citizens' demand for emergent service delivery, with a focus on value creation as a process where value is cocreated and negotiated through the ongoing collaboration between public sector organizations and citizens. Crucially, this underscores the importance of recognizing the context-dependent and emergent nature of value, where the perceived value of a service will change in line with changing user expectations and knowledge. Successful service delivery therefore requires a longer-term, process-oriented approach where public sector organizations continuously seek knowledge, feedback and information from citizens, which in turn are used to continuously improve service delivery.

In practice, adopting a process-oriented approach has proven difficult. Citizen input and feedback are generally used to cocreate requirements at the beginning of a project or measure user satisfaction after services are launched [4]. This signifies the remains of a manufacturing-oriented logic that effectively obstructs public sector organizations´ capability to respond to emergent citizen needs [3].

The existing literature predominantly focuses on cocreation during early design and specification phases, where user feedback is directly transmitted from citizens to service providers [5-8]. However, to achieve the promises of a service dominant logic [3, 9], there is a need to address cocreation as an ongoing process, where cocreation is mediated throughout the entire service delivery cycle. Public administrators are therefore exploring novel means to achieve more agile and continuous value cocreation [10]. In this regard, digital platforms have significant potential to realize a more process-oriented approach. This is due to the digital platform's ability to mediate between service providers and users and scale up user engagement through mediated forms of cocreation [11, 12].

Recently, calls have been made for improved insight into the ways in which feedback from users is captured and reintegrated at a service level and the role of technology in such forms of value cocreation [5, 13]. Answering to these calls, this paper examines the following research question: *How do digital platforms promote process-oriented, mediated value cocreation in public sector organizations?*

To answer this question, we draw on insights from a longitudinal case study of the IT department in the Norwegian Labour and Welfare Administration (NAV). NAV serves millions of users, has almost 19 000 employees and is responsible for redistributing one third of the national budget through schemes such as age pension, sick-benefit, and disability benefit. In 2017, NAV made radical changes to its service delivery model, moving from a manufacturing-oriented approach towards more process-oriented service delivery. We aim to contribute by explaining how adopting a process-oriented approach for value cocreation in NAV required structural changes, including sourcing strategy and governance structure. We also show the key role of digital platforms in capturing and reintegrating feedback into subsequent service delivery cycles.

The rest of the paper is organized as follows: In section 2, we present an overview of the literature on value cocreation, followed by a review of the literature on digital platforms within the public sector, before we present the theoretical framework that was used to analyze our data. In section 3, we describe the research setting and methods, while section 4 presents our findings. Finally, in section 5, we discuss how these structural changes were possible and the role of digital platforms in achieving these changes before making concluding remarks.

2. Theoretical background

Value cocreation denotes a logic of value creation where value is seen as created in the interaction between provider and users [9, 14]. In the following sections, we begin by discussing value cocreation in the public administration literature before exploring the way in which digital platforms might affect value cocreation. Finally, we present our theoretical framework.

2.1 Cocreating public services

Public sector organizations have traditionally been dominated by a manufacturing logic, where value is seen as created by a service organization and delivered to citizens who take the role of passive consumers. This logic has, however, come under increasing criticism for failing to address the complex, fragmented, and emergent needs of citizens [3, 13, 15]

As a consequence, researchers have identified an alternative logic, where value is seen as cocreated in the interaction between public sector organizations and citizens [9, 15]. Central to this *service-dominant logic* is that public sector organizations cannot create value for citizens—they can only make a "value proposition" that the citizen might choose to use [3, 9]. Hence, value is created in use ("value-in-use") [3].

Furthermore, service-dominant logic emphasizes that value propositions and their potential to create value for citizens depend on the social context in which the service is offered [16]. As the context changes, for instance, as citizens acquire new knowledge or appropriate new technology, preferences and needs will change. If services are to be perceived as valuable *over time,* public sector organizations must therefore continuously seek feedback from citizens and improve value propositions accordingly. The ability to sense and respond to evolving needs requires agility and responsiveness on the part of public sector organizations, often contradicting established structures that favor internal efficiency over external efficacy [13].

A founding idea in service-dominant logic is that value is cocreated through the interaction between suppliers and users. It pinpoints the challenge of shifting from a supply-side focus in the delivery of public services to a demand-side focus. There are, however, three shortcomings in the manner in which value cocreation has been employed in the context of public sector services. First, the cocreation between citizens and the supply side is assumed to take the form of direct engagement and interaction. For all its merit, direct involvement of citizens is only feasible for small populations; the scaling of participatory methods of technology development by necessity needs to find indirect, mediated forms of representing citizens' voices [17, 18]. Second, common for much of this research is an emphasis on "the involvement of citizens in the initiation and/or design of public services" [8], most often neglecting cocreation in later stages of service delivery [13]. Third, the extant literature pays little attention to the role of communication technology in promoting value cocreation in public sector organizations [5, 13].

Recently, digital platforms have emerged as a promising approach to transforming public sector organizations and increasing the capacity for cocreation. In the following section, we give a short overview of the way digital platforms have been addressed in the public administration literature and discuss the way in which platforms might enable increased value cocreation.

2.2 Digital platforms in the public sector

Digital platforms enable innovation [12], value cocreation [19], and user involvement [11] and have been studied as a means for increasing public and private value creation [20]. From an economic perspective, platforms create value by acting as mediators between two or more categories of users who would otherwise not connect [21, 22], while they, from an engineering perspective, are seen as technology foundations that enhance generativity and innovation through their layered modular structure [12, 23].

The advent of platform ecosystems is radically transforming the way private and public sector organizations interact with their users. Digital platforms let governments tap into existing communication channels [24, 25], thereby engaging citizens in the arenas they know. For instance, Hand and Ching [26] examine how social media platforms such as Facebook and Twitter let citizens engage with police agencies, while Nam [27] explores the way in which digital platforms enable discussions about rule making between citizens and other stakeholders. Similarly, many studies explore the challenges and opportunities relating to open government data, focusing on issues such as innovation [28], civic engagement [29], and the design of open data platforms [30]. Public sector platforms can potentially increase both transparency and efficiency by exposing public sector data and engaging citizens in cocreation [31].

While the debate on digital platforms has proven useful, much of the existing literature has focused on digital platforms as a means for communication between public sector organizations and citizens. As an exception to this trend, Dunleavy et al. [32] argue that we have entered an era of digital governance, where public sector developments revolve around changes in digital technologies and alterations in information systems. By reintegrating public service, digital technologies are enabling a "needs-based holism" where end-to-end processes and agile practices are increasing public sector organizations' ability to respond to emerging citizen needs [32]. Similarly, Fishenden and Thompson [33] propose that digital platforms and open architectures enable a reaggregation of digital services, promoting a service-dominant approach where citizens become an integral part of the value creation process. Central to this transformative potential is the platform's ability to mediate between different user groups and offer resources that can be recombined into new and improved services. Hence, the platform becomes a venue where citizens and public sector organizations can interact and exchange services and information. Digital platforms are thus important in public sector service delivery for at least two reasons. First, digital platforms facilitate the exchange of services and information between citizens and public sector organizations. Second, platforms enable a rapid and ongoing reintegration of this information into new and improved value offerings [14].

2.3. Processual perspective on value cocreation

Digitally enabled participation and production of services is changing citizens' expectations about public sector services [34]. To ensure continued trust in governments, public sector organizations need to move from anticipating citizens' needs (manufacturing-oriented approaches) to approaches where services are developed in response to the actual needs of citizens. Although prior literature on value cocreation has recognized the need for an alternative logic in public sector organizations, it provides little insight into how such value cocreation can be achieved in practice [3, 13]. We argue that traditional forms of cocreation are poorly suited for the large-scale and dynamic context of public sector organizations.

To close this gap, there are several assumptions worth making. First, direct involvement as the sole means for capturing citizen feedback is insufficient for collecting the needs of large and heterogeneous user groups. Instead, organizations need to adopt practices that enable indirect and mediated forms of interaction where feedback can be gathered from large user groups. Second, feedback must be collected and reintegrated *throughout* the service delivery cycle, not only during initiation and design. Third, to adopt cocreation *across* organizations, traditional structures of centralization and control must be replaced by more flexible technical and organizational structures that enable agility and innovation.

To further our understanding of how public sector organizations can achieve ongoing value cocreation, we have conducted a case study of NAV. During the past few years, NAV has undergone radical changes to the way it develops and delivers public services, moving from a manufacturing-oriented approach towards a more service-dominant logic. In the following sections, we describe the methods used to investigate the transformation and the results we obtained.

3. Research setting and methods

Fieldwork was conducted within the IT department at the Norwegian Labour and Welfare Administration (NAV). The IT department consists of approximately 700 employees, 400 consultants, and operates and maintains close to 300 applications.

NAV was established in 2006, following the merger of the Employment Agency, the National Insurance Agency, and Social Services. NAV is responsible for increasing the population's work ability, as well as supporting citizens economically during periods when they are unable to support themselves. Among the services they provide are age pensions, unemployment benefits, sick benefits, and disability benefits. Most Norwegian citizens will at some point come in contact with NAV. The organization has almost 2.8 million active users at any given time.

In 2015, an expert committee criticized NAV for failing to improve digital services in response to emergent needs and for paying too little attention to user experiences [35]. As a response to this criticism, NAV made radical changes to its sourcing strategy, technical infrastructure, and governance model.

To examine these changes, we performed an interpretive longitudinal case study. Data were collected over a two-year period from January 2017 to May 2019 and consisted of document analysis, participant observation, and semi structured interviews. The study of the ongoing change process was complemented by a historical reconstruction based on archived documents and informants' recollection of the past.

First, we conducted a total of 38 interviews. We chose informants using a snowballing strategy, where one informant suggested the next. In this way, we gradually traversed the IT department, including informants from all levels of the organization. Among informants were the former and present CTO (2), program and department managers (4), project managers (4), team leads (6), IT architects (8), software developers (12), and case workers (2). These differing perspectives were important to capture both the strategic motivations behind the change and its practical implications. For instance, CTOs, senior executives, and managers were able to shed light on the motivations and larger context, whereas IT architects, team leads, and developers provided insight into the technical implementation and their consequences.

Of the 38 interviews, 23 were recorded and transcribed. Because of their sensitive nature, not all interviews could be recorded. In these cases, we took notes during interviews and added more extensive remarks after the interviews ended. Interviews lasted between 45 and 60 minutes.

Second, participant observation was another important source of information. The first author was able to move freely within the IT department and could also attend most meetings and social gatherings. She has a background as a software developer and IT consultant and could easily blend into the environment. Considerable insight was gained through informal conversations by the coffee machine and encounters in the hallway. Many of the informants were recruited through this informal relation building. Observations and conversations were extensively documented in a field diary.

Third, our study included numerous documents collected from internal and external web sites and archiving systems. Among these were governmental white papers, procurement documents, design specifications, project reports, and websites. The first author was given an internal account and could access most internal documents, including calendars, project wikis, and issue tracking systems. In addition, online conference presentations held by NAV employees were transcribed and analyzed. Since the digital platform used to facilitate the shift was exposed as open-source code on GitHub, we were able to examine its functionality in great detail (www.nais.io), including features relating to monitoring and feedback.

Data collection and data analysis were performed in tandem to benefit from the understanding emerging from recursively iterating between theoretical conceptions and the empirical material [36]. Specifically, our data analysis can be described as an iterative three-step process. First, interviews were transcribed and coded. We used descriptive codes, capturing the informant's views and reflections on the transformation. For instance, the code "The platform is used to change the organizational culture" captures the interaction between technology and organization, where the introduction of the platform was seen to enable social change. Codes were later merged into 3 themes that captured relevant aspects of the transformation of NAV. The themes are listed in Tables 1 and 2.

Second, we used visual mapping to display the progression of events between 2012 and 2019. By using a method of temporal bracketing [37], we identified two periods in which service delivery was approached in distinctly different ways: The first period (2012 – 2016) was dominated by large projects with staged development and limited user input, whereas the second period (2017 – 2019) was characterized by incremental approaches where user feedback was continuously monitored and reintegrated into subsequent service delivery.

Third, we iterated between theoretical abstractions relating to service-dominant logic and themes uncovered in the previous phase of analysis. Elements of manufacturing logic mapped accurately to the first period, whereas the last period was characterized by service-dominant logic. Based on the analysis, we inferred that NAV had transitioned from a manufacturing logic to a service-dominant logic and that the change was captured by elements relating to 1) sourcing strategy, 2) technical platform, and 3) governance model.

4. Results

In the following, we present the two time periods uncovered in our analysis and describe the alternative ways in which the sourcing strategy, technical platform, and governance model were addressed in each of the two periods.

4.1. Manufacturing logic (2012 - 2016)

From 2012 to 2016, software development in NAV was organized as large projects where development and maintenance were outsourced to consultant companies. Information systems were large and interconnected, and projects followed a staged delivery model where requirements elicitation and user involvement were isolated to early stages of the development process. Dependencies were managed through centralized control and coordinated releases. The elements are summarized in Table 1 and elaborated on in three subsections.

Table 1. Elements of manufacturing logic

Element	Contents
Sourcing strategy	Service development was organized as large projects where the software development was outsourced to consultant companies
Technical platform	Large and interdependent IT systems required coordination and control
Governance strategy	Staged development methods and centralized control restricted user involvement to early stages

4.1.1. Sourcing strategy. In the period from 2012 to 2016, IT development was organized as large projects where the development and maintenance of information systems were outsourced to consultant companies. To ensure predictability and control, NAV introduced a clear separation between customers and suppliers, where requirements elicitation and user involvement were isolated to early stages of the development process. Changes to the agreed-upon requirements often required formal approval and additional funding, limiting the organization's ability to respond to emergent needs.

In line with public sector procurement regulations, maintenance contracts were put out to tender every 4 - 8 years. In this way, suppliers were replaced at regular intervals, causing discontinuity and loss of key competence. "*At any given time, NAV would have 15-20 distinct suppliers developing and maintaining its core systems. These suppliers had to be coordinated and controlled"* (CTO).

A significant part of IT modernization in NAV was funded over the national budget. To minimize the administrative overhead associated with such funding, project proposals would contain a large and dispersed collection of prospective needs, increasing both complexity and risk.

A prominent example of this funding model was a large modernization program that was initiated in 2012. The main purpose of the program was to renew NAV's IT portfolio and increase efficiency through automation and self-service solutions. The program had an estimated cost of 3.3 billion Norwegian Kroner (approx. 349 million U.S. dollars) and would be performed through three consecutive projects—lasting from 2012 until 2018.

4.1.2. Technical platform. After NAV was established in 2006, the system portfolio consisted of large and heterogeneous systems. To reduce technical heterogeneity and simplify operations and management, NAV began to standardize on a single application platform. By 2016, most systems were running on a single application platform. The platform was based on JBoss application servers running on a Red Hat Linux operating system and virtual servers. The goal was to eventually run all applications on the same platform. However, two of the core systems were too large and too tightly connected to the underlying hardware for migration to occur. Thus, by the end of 2016, NAV had three technical platforms: 1) Infotrygd - an IBM mainframe from 1978, 2) Arena - an Oracle forms-based system introduced in 2001, and 3) a JBoss application server running on a Red Hat Linux operating system.

Although technical heterogeneity was reduced, systems were still large and interdependent. To manage these dependencies and ensure stable operations, release management was centralized and coordinated across projects. All software releases had to be tested and approved by the operations department. For maximum resource utilization, deployments were bundled into four yearly releases. Although the strategy provided predictability and internal efficiency, it reduced the flexibility and responsiveness of development teams: It could take months from when a feature was developed until it became available to end-users, and teams tried to predict future needs as a means for reducing response times.

4.1.3. Governance strategy. Software development was organized as distinct and nonoverlapping stages, where different departments were responsible for different stages of the development process. For instance, design and specification had to be completed before the project could begin to develop the system, and development had to be finalized and approved before the application could be released into production.

The development strategy reduced the responsiveness of development teams in several ways. First, user input was isolated to early stages of the development process. Second, changes to initial specifications required formal approval and possibly additional funding. Third, it could take years from project initiation until the system was completed and available. During this time, the needs and expectations of users would evolve, and the completed system could become obsolete.

To ensure consistency across suppliers and projects, NAV introduced a centralized governance model and a technology "catalog" listing approved technologies. Any decision to appropriate new technologies or use old technologies in new ways had to be approved by an IT architecture decision board. The strategy increased predictability but effectively reduced local initiatives and innovation.

4.2. Service-dominant logic

Following the criticism of the expert committee in 2015 [35], NAV made several changes to its digital service strategy. First, the outsourcing strategy was replaced by an insourcing strategy. Second, monolithic systems were gradually dismantled into more loosely coupled applications. Third, the staged software development method was developed by an iterative approach where development teams were developed and maintained by independent teams responsible for the entire service delivery cycle. Table 2 summarizes these changes.

Table 2. Elements of a service-dominant logic

Element	**Contents**
Sourcing strategy	Insourcing of software development where software development activities are funded over the operating budget
Technical platform	Monolithic and interdependent applications are dismantled into more loosely coupled applications
Governance strategy	Independent teams assume responsibility for the entire software development cycle

4.2.1. Sourcing strategy. NAV changed its sourcing strategy in 2017. The outsourcing of software development was replaced by an insourcing strategy where NAV would assume responsibility for developing and maintaining core systems. As the old contracts expired, responsibility contracts were replaced by capacity contracts where consultants were hired per hour. The long-term objective was that consultants would only be used during peak periods and to provide specialized competence.

To accommodate the new sourcing strategy, NAV began an aggressive recruitment campaign, aiming to employ hundreds of software developers within a few years. During the two years the study lasted, NAV recruited close to 200 developers. Competitive salaries and promises of modern technologies made NAV an attractive employer. A key objective behind the insourcing strategy was to strengthen internal competence and provide continuity and learning.

The altered strategy also affected the funding model: Although service development still required external funding, the funding was used to finance existing teams. By maintaining stable teams with stable responsibilities, continuity and predictability were increased. This stood in stark contrast to the manufacturing-oriented approach, where periods of intense activity were followed by periods of relative calm. The long-term goal was for the organization to become less dependent on external funding and that most development activities be financed over the operating budget.

4.2.2. Technical platform. To increase the flexibility and maintainability, NAV began to dismantle legacy systems into more loosely coupled applications. To facilitate the dismantling of legacy systems, NAV introduced a second-generation application platform in 2017. The application platform was called "NAIS", short for NAV's Application Infrastructure Service and was based on Kubernetes. Kubernetes is an open-source framework developed by Google. The platform offers fully automated services for tasks such as provisioning and deployment. As expressed by a member of the platform development team, *"Kubernetes is the open source framework that comes from Google. It is all of Google's experience over the last 15 years with how to manage infrastructure - rewritten by the same people. It is such as taking the world's best operations person and fully automating him. That is what Kubernetes is. It provides many tools for running in production, which makes it more robust and more scalable and everything"*.

The NAIS platform also simplified the monitoring of application performance and use. These metrics were displayed on a large screen in the team area, providing development teams with immediate and continuous feedback from systems and users. Through this mediated interaction with citizens, development teams were able to continuously improve services in response to actual use. Mediated feedback from monitoring mechanisms was complemented with traditional forms of direct user input, such as "guerrilla" interviews, surveys, design workshops, and prototyping. Together, these strategies provided the team with rich insight into the application of strong points and shortcomings. The loosely coupled architecture of the platform, combined with functionality for automated provisioning and deployment, enabled development teams to rapidly reintegrate feedback from citizens could into new and improved services.

4.2.3. Governance strategy. The dismantling of legacy systems into a more modular structure enabled a restructuring of the IT department: The staged development model was replaced by an iterative approach where independent teams were responsible for the entire software development cycle. To effectuate this shift, the IT department was reorganized in 2017. The "plan-build-run" hierarchy was replaced by a decentralized control structure where employees were assigned to multidisciplinary service development teams. Team members had various backgrounds, including software developers, interface designers, IT architects, and domain experts.

A leading principle behind the reorganization was that development teams would have the competence and authority to develop services independently of other teams and that they would be responsible for the entire service delivery cycle—from the inception of an idea until the service was eventually turned off.

By duplicating competence across teams and introducing a distributed decision model, development teams could work independently, and release applications as needed. Centralized control and coordinated releases were replaced by decentralized decisions and continuous releases. For many teams, deployment rates increased from once every three months to several times a day. In this way, user feedback was rapidly and continuously reintegrated into service releases.

5. Discussion

The purpose of this study was to investigate how digital platforms promote process-oriented, emergent value cocreation in government organizations. Through the research question - *how do digital platforms promote process-oriented, emergent value cocreation in government organizations* - we have reported findings from an explorative case study of a large public IT department. The study aims to contribute in two important ways. First, we examine the organizational and strategic changes necessary to enable continuous and ongoing value cocreation across large and heterogeneous user groups. Second, we emphasize the role of digital platforms in scaling value cocreation in time and space. Each of these contributions is discussed in further detail below.

5.1. Process perspective on service delivery

Extant research discusses the benefits, drivers, and barriers of cocreation in the public sector [5, 7] with an emphasis on cocreation as part of the initiation or early design [8]. We complement these studies by exploring the structural changes undertaken by NAV to achieve value cocreation across large and heterogeneous user groups throughout the service delivery cycle.

First, NAV changed the *sourcing strategy* - transitioning from an outsourcing strategy to an insourcing strategy. By employing software developers and gradually replacing consultants with internal employees, NAV ensured continuity and predictability, both in terms of financing and competence. While software development had previously been financed through large-scale projects, software would now become a continuous activity performed by internal employees, financed over the operating budget. This provided predictability and continuity, allowing the organization to build the knowledge and skills required to continuously improve services.

Second, they changed the *governance strategy* - replacing top-down control and handovers between departments with a bottom-up approach, where independent, self-organizing teams were responsible for the entire service development cycle [38]. By establishing multidisciplinary teams with the skills, knowledge, and authority to solve problems independently, NAV was able to continuously sense and react to the emergent needs of citizens.

Our findings correspond with insights from service-dominant logic, which suggests that organizations must engage in continuous and ongoing improvements to ensure value cocreation throughout the development cycle [9, 13]. However, our study addresses a blind spot in the current literature by questioning the applicability of direct user interaction as a means for achieving continuous and ongoing value cocreation across large and heterogeneous user groups [17, 18]. In this way, we complement existing studies by emphasizing the context-dependent and emergent nature of value cocreation, arguing that public sector organizations need to radically restructure their service delivery models and employ mediated forms of feedback and learning.

Although other studies have addressed the need for more responsive service delivery methods in public sector organizations [1, 2, 39], these studies either do not address the structural changes needed to adopt such approaches [2, 39] or they view agility and responsiveness as "add-ons" that apply in selected cases [1]. In contrast, our study sees value cocreation as a set of processes and activities that are applied across departments and organizations, radically changing the way public sector organizations organize and deliver service.

5.2. Platforms as enablers

Our findings suggest that digital platforms play a pivotal role in enabling efficient value cocreation within public sector organizations. At NAV, the container-based application platform enabled cocreation in three important ways. First, the modular structure of the platform enabled the formation of independent development teams that could work in relative isolation. As long as application interfaces remained intact, development teams could experiment and innovate inside the boundaries of their applications [40].

Second, the platform provided indirect and mediated feedback from citizens. By monitoring application use and performance, development teams were able to continuously capture the reactions of citizens. Third, the platform simplified provisioning and deployment, thereby enabling continuous and ongoing reintegration of feedback into subsequent service deliveries. These insights comply with insights from service-dominant logic, which suggest that digital platforms increase both the efficiency and effectiveness of resource exchange [14].

Based on our findings, we further suggest that by enabling mediated feedback and rapid reintegration into subsequent service delivery, platforms have the potential to scale cocreation in both time and space. While other studies explore the ways in which digital platforms enable improved communication between citizens and governments within existing structures [26, 41], we thereby take a step further and examine the ways in which platforms might enable the formation of radically new structures and improved forms of service delivery.

Further, we address the relation between the structure of the digital infrastructure and the organization's ability to develop and deliver services, suggesting that the transformation of public sector organizations preconditions a transformation of the digital infrastructure: Only by increasing the flexibility of the infrastructure are organizations able to scale value cocreation across the organization, incorporating feedback from large and heterogeneous user groups over prolonged periods of time.

The focus of our study has been the broad strategic and technical changes needed to move public sector organizations towards more service-dominant logic. To pursue this goal, we have adopted a supply-side focus in the exploration of organizational and technological changes. We have largely ignoring the perceptions of citizens in our exploration of the ongoing transformation. The rationale behind this decision is two-fold: First, capturing both the supply side *and* demand side in a complex case such as NAV was not possible within the constraints of our research projects. Second, many of NAV's services are part of a larger value chain, including a wide array of public and private actors outside NAV. It will therefore take time before the effects of the ongoing transformation propagate out to citizens. We therefore hold the exploration of citizens' opinions and experiences as an opportunity for future research.

In addition, further research is needed to uncover the long-term effects of such transformations. NAV underwent significant changes during the course of our fieldwork, but considerable work remains.

Further, our research lacks details of the specific monitoring and feedback mechanisms used in the delivery process. Exploring the different forms of mediated feedback and the way in which they evolve over time presents another opportunity for future research.

Finally, our findings are limited to one specific case and context. Exploring the applicability of similar approaches in other public sector contexts thus presents another opportunity for future research.

6. Conclusion

In this paper, we have approached value cocreation as a process of ongoing improvement, where public sector organizations must implement the means to apply feedback and learning throughout the entire service development cycle. We have described how digital platforms promote such service-dominant logic by mediating interaction with citizens and facilitating the reintegration of feedback into subsequent service delivery. We found that adopting a process-oriented approach for value cocreation within public sector organizations requires structural changes, including sourcing strategy, governance structure, and more flexible digital infrastructure.

7. Acknowledgments

We would like to thank NAV for granting us access to the sites and all informants for taking the time to talk to us. We would also like to thank Torgeir Dingsøyr for providing valuable comments during the writing of the paper. This work was supported by the project Agile 2.0 supported by the Research Council of Norway through grant 236759 and by the companies DNV GL, Equinor, Kantega, Kongsberg Defense & Aerospace, Sopra Steria and Sticos.

8. References

[1] I. Mergel, N. Edelmann, and N. Haug, “Defining digital transformation: Results from expert interviews,” *Government Information Quarterly,* vol. 36, no. 4, pp. 101385, 2019.

[2] I. Mergel, “Agile innovation management in government: A research agenda,” *Government Information Quarterly,* vol. 33, no. 3, pp. 516-523, 2016.

[3] S. P. Osborne, "From public service-dominant logic to public service logic: are public service organizations capable of co-production and value co-creation?," Taylor & Francis, 2018.

[4] OECD, *Digital Government Review of Norway - Boosting the digital transformation of the public sector*, OECD, OECD Publishing, Paris, 2017.

[5] N. Baptista, H. Alves, and N. Matos, “Public Sector Organizations and Cocreation With Citizens: A Literature Review on Benefits, Drivers, and Barriers,” *Journal of Nonprofit & Public Sector Marketing*, pp. 1-25, 2019.

[6] J. Trischler, S. J. Pervan, S. J. Kelly, and D. R. Scott, “The value of codesign: The effect of customer involvement in service design teams,” *Journal of Service Research,* vol. 21, no. 1, pp. 75-100, 2018.

[7] W. H. Voorberg, V. J. Bekkers, and L. G. Tummers, “A systematic review of co-creation and co-production: Embarking on the social innovation journey,” *Public Management Review,* vol. 17, no. 9, pp. 1333-1357, 2015.

[8] W. Voorberg, V. Bekkers, S. Flemig, K. Timeus, P. Tonurist, and L. Tummers, “Does co-creation impact public service delivery? The importance of state and governance traditions,” *Public Money & Management,* vol. 37, no. 5, pp. 365-372, 2017.

[9] S. L. Vargo, and R. F. Lusch, “Evolving to a new dominant logic for marketing,” *Journal of marketing,* vol. 68, no. 1, pp. 1-17, 2004.

[10] I. Mergel, S. Ganapati, and A. B. Whitford, “Agile: A New Way of Governing,” *Public Administration Review*, 2020.

[11] B. Eaton, S. Elaluf-Calderwood, C. Sorensen, and Y. Yoo, “Distributed tuning of boundary resources: the case of Apple's iOS service system,” *MIS Quarterly: Management Information Systems,* vol. 39, no. 1, pp. 217-243, 2015.

[12] Y. Yoo, O. Henfridsson, and K. Lyytinen, “Research commentary—the new organizing logic of digital innovation: an agenda for information systems research,” *Information systems research,* vol. 21, no. 4, pp. 724-735, 2010.

[13] S. P. Osborne, Z. Radnor, and K. Strokosch, “Co-production and the co-creation of value in public services: a suitable case for treatment?,” *Public Management Review,* vol. 18, no. 5, pp. 639-653, 2016.

[14] R. F. Lusch, and S. Nambisan, “Service innovation: A service-dominant logic perspective,” *MIS quarterly,* vol. 39, no. 1, 2015.

[15] S. P. Osborne, Z. Radnor, and G. Nasi, “A new theory for public service management? Toward a (public) service-dominant approach,” *The American Review of Public Administration,* vol. 43, no. 2, pp. 135-158, 2013.

[16] R. F. Lusch, S. L. Vargo, and M. Tanniru, “Service, value networks and learning,” *Journal of the academy of marketing science,* vol. 38, no. 1, pp. 19-31, 2010.

[17] T. Dingsøyr, D. Falessi, and K. Power, “Agile development at scale: the next frontier,” *IEEE Software,* vol. 36, no. 2, pp. 30-38, 2019.

[18] L. K. S. Roland, Terje Aksel; Sæbø, Johan Ivar; Eric Monteiro, “P for Platform,” *Scandinavian Journal of Information Systems,* vol. 30, no. 2, 2018.

[19]C. Cennamo, and J. Santaló, "Generativity tension and value creation in platform ecosystems," *Organization Science,* vol. 30, no. 3, pp. 617-641, 2019.

[20]J. Ju, L. Liu, and Y. Feng, "Public and private value in citizen participation in E-governance: Evidence from a government-sponsored green commuting platform," *Government Information Quarterly,* vol. 36, no. 4, pp. 101400, 2019.

[21]T. Eisenmann, G. Parker, and M. W. Van Alstyne, "Strategies for two-sided markets," *Harvard business review,* vol. 84, no. 10, pp. 92, 2006.

[22]G. G. Parker, M. W. Van Alstyne, and S. P. Choudary, *Platform Revolution: How Networked Markets Are Transforming the Economyand How to Make Them Work for You*: WW Norton & Company, 2016.

[23]A. Tiwana, B. Konsynski, and A. A. Bush, "Research commentary—Platform evolution: Coevolution of platform architecture, governance, and environmental dynamics," *Information Systems Research,* vol. 21, no. 4, pp. 675-687, 2010.

[24]E. Bonsón, S. Royo, and M. Ratkai, "Citizens' engagement on local governments' Facebook sites. An empirical analysis: The impact of different media and content types in Western Europe," *Government information quarterly,* vol. 32, no. 1, pp. 52-62, 2015.

[25]S. M. Zavattaro, P. E. French, and S. D. Mohanty, "A sentiment analysis of US local government tweets: The connection between tone and citizen involvement," *Government information quarterly,* vol. 32, no. 3, pp. 333-341, 2015.

[26]L. C. Hand, and B. D. Ching, "Maintaining neutrality: A sentiment analysis of police agency Facebook pages before and after a fatal officer-involved shooting of a citizen," *Government Information Quarterly,* vol. 37, no. 1, pp. 101420, 2020.

[27]C. Nam, "Behind the interface: Human moderation for deliberative engagement in an eRulemaking discussion," *Government Information Quarterly,* vol. 37, no. 1, pp. 101394, 2020.

[28]L. Danneels, S. Viaene, and J. Van den Bergh, "Open data platforms: Discussing alternative knowledge epistemologies," *Government Information Quarterly,* vol. 34, no. 3, pp. 365-378, 2017/09/01/, 2017.

[29]M. Kassen, "A promising phenomenon of open data: A case study of the Chicago open data project," *Government Information Quarterly,* vol. 30, no. 4, pp. 508-513, 2013.

[30]E. Ruijer, S. Grimmelikhuijsen, and A. Meijer, "Open data for democracy: Developing a theoretical framework for open data use," *Government Information Quarterly,* vol. 34, no. 1, pp. 45-52, 2017.

[31]T. O'Reilly, "Government as a Platform," *Innovations: Technology, Governance, Globalization,* vol. 6, no. 1, pp. 13-40, 2011.

[32]P. Dunleavy, H. Margetts, S. Bastow, and J. Tinkler, "New public management is dead—long live digital-era governance," *Journal of public administration research and theory,* vol. 16, no. 3, pp. 467-494, 2006.

[33]J. Fishenden, and M. Thompson, "Digital government, open architecture, and innovation: why public sector IT will never be the same again," *Journal of public administration research and theory,* vol. 23, no. 4, pp. 977-1004, 2012.

[34]OECD, *OECD Recommendation on Digital Government Strategies*, OECD, OECD Publishing, Paris, 2014.

[35]S. Vågeng, *Et NAV med muligheter*, Ministry of Labor and Social Affairs, https://www.regjeringen.no/no/aktuelt/ekspertgruppens-forslag-til-et-bedre-nav/id2404844/, 2015.

[36]H. K. Klein, and M. D. Myers, "A set of principles for conducting and evaluating interpretive field studies in information systems," *MIS quarterly*, pp. 67-93, 1999.

[37]A. Langley, "Strategies for theorizing from process data," *Academy of Management*, 1999.

[38]M. M. Mikalsen, Nils B; Stray, Viktoria; Nyrud, Helga, "Agile digital transformations: a case study of interdependencies," in ICIS, 2018.

[39]J. Torfing, E. Sørensen, and A. Røiseland, "Transforming the Public Sector Into an Arena for Co-Creation: Barriers, Drivers, Benefits, and Ways Forward," *Administration & Society,* vol. 51, no. 5, pp. 795-825, 2016.

[40]A. Gawer, "Bridging differing perspectives on technological platforms: Toward an integrative framework," *Research Policy,* vol. 43, no. 7, pp. 1239-1249, 2014/09/01/, 2014.

[41]M. D. de Jong, S. Neulen, and S. R. Jansma, "Citizens' intentions to participate in governmental co-creation initiatives: Comparing three co-creation configurations," *Government information quarterly,* vol. 36, no. 3, pp. 490-500, 2019.

NTNU

FACULTY OF INFORMATION TECHNOLOGY AND ELECTRICAL ENGINEERING NORWEGIAN UNIVERSITY OF SCIENCE AND TECHNOLOGY

DECLARATION OF CO-AUTHORSHIP

Declaration of co-authorship is to be included when a thesis containing papers written by more than one author must include this signed declaration that describes the contribution of the candidate and the co-authors of each of the papers. It must be possible to identify the candidate's independent contribution in the work. *(cf. the PhD regulations section 10 or dr.philos regulations section 3).*

Name of candidate: Kathrine Vestues

Title of dissertation: Using digital platforms to promote value cocreation: A case study of a public sector organization

Publication: Vestues, Kathrine; Bjørnson, Finn Olav. (2016). *Agile requirements work in a digital transformation project: Managing diverse and dispersed user needs*. Paper presented at the International Research Workshop on IT Project Management (IRWITPM)

Description of the Candidate's contribution: Kathrine had the initial idea, and had the main responsibility for writing, data collection, and data analysis.

Statement by the co-author:
I hereby confirm that the doctoral candidate's described contribution to this paper is correct, and I consent to including the paper in the named dissertation.

Trondheim, 22.02.2021

[Name and signature of co-author]

DECLARATION OF CO-AUTHORSHIP

Name of candidate: Kathrine Vestues

Title of dissertation: Using digital platforms to promote value cocreation: A case study of a public sector organization

Publication: Dingsøyr, Torgeir; Mikalsen, Marius; Solem, Anniken; Vestues, Katherine. (2018) *Learning in the Large: An Exploratory Study of Retrospectives in Large-Scale Agile Development*. Agile Processes in Software Engineering and Extreme Programming, 19th International Conference, XP 2018, Proceedings

Description of the Candidate's contribution: Kathrine contributed to the data collection and data analysis. In addition, I wrote the case description.

Statement by the co-author:
I hereby confirm that the doctoral candidate's described contribution to this paper is correct, and I consent to including the paper in the named dissertation.

Kristiansand, 22.02.2021

[illegible]

DECLARATION OF CO-AUTHORSHIP

Declaration of co-authorship is to be included when a thesis containing papers written by more than one author must include this signed declaration that describes the contribution of the candidate and the co-authors of each of the papers. It must be possible to identify the candidate's independent contribution in the work. *(cf. the PhD regulations section 10 or dr.philos regulations section 3).*

Name of candidate: Kathrine Vestues

Title of dissertation: Using digital platforms to promote value cocreation: A case study of a public sector organization

Publication: Dingsøyr, Torgeir; Mikalsen, Marius; Solem, Anniken; Vestues, Katherine. (2018) *Learning in the Large: An Exploratory Study of Retrospectives in Large-Scale Agile Development*. Agile Processes in Software Engineering and Extreme Programming, 19th International Conference, XP 2018, Proceedings

Description of the Candidate's contribution: Kathrine contributed to the data collection and data analysis. In addition, I wrote the case description.

Statement by the co-author:
I hereby confirm that the doctoral candidate's described contribution to this paper is correct, and I consent to including the paper in the named dissertation.

Malvik, 22.02.2020

[illegible]

[Name and signature of co-author]

DECLARATION OF CO-AUTHORSHIP

Declaration of co-authorship is to be included when a thesis containing papers written by more than one author must include this signed declaration that describes the contribution of the candidate and the co-authors of each of the papers. It must be possible to identify the candidate's independent contribution in the work. *(cf. the PhD regulations section 10 or dr.philos regulations section 3).*

Name of candidate: Kathrine Vestues

Title of dissertation: Using digital platforms to promote value cocreation: A case study of a public sector organization

Publication: Dingsøyr, Torgeir; Mikalsen, Marius; Solem, Anniken; Vestues, Kathrine. (2018) *Learning in the Large: An Exploratory Study of Retrospectives in Large-Scale Agile Development*. Agile Processes in Software Engineering and Extreme Programming, 19th International Conference, XP 2018, Proceedings

Description of the Candidate's contribution: Kathrine contributed to the data collection and data analysis. In addition, I wrote the case description.

Statement by the co-author:
I hereby confirm that the doctoral candidate's described contribution to this paper is correct, and I consent to including the paper in the named dissertation.

Trondheim, 23.02.2021

Anniken Solem

Anniken Solem

[Name and signature of co-author]

NTNU

DECLARATION OF CO-AUTHORSHIP

Declaration of co-authorship is to be included when a thesis containing papers written by more than one author must include this signed declaration that describes the contribution of the candidate and the co-authors of each of the papers. It must be possible to identify the candidate's independent contribution in the work. *(cf. the PhD regulations section 10 or dr.philos regulations section 3).*

Name of candidate: Kathrine Vestues

Title of dissertation: Using digital platforms to promote value cocreation: A case study of a public sector organization

Publication: Vestues, Kathrine; Mikalsen, Marius; Eric Monteiro (2021), *Using digital platforms to promote a service-oriented logic in public sector organizations: A case study*, Accepted to the 54th Hawaii International Conference on System Sciences.

Description of the Candidate's contribution: Kathrine did the data collection, data analysis, and wrote most of the paper. Marius Mikalsen and Eric Monteiro contributed to the writing of the introduction, theoretical framework, as well as providing comments.

Statement by the co-author:
I hereby confirm that the doctoral candidate's described contribution to this paper is correct, and I consent to including the paper in the named dissertation.

Malvik, 22.02.2020

[illegible]

[Name and signature of co-author]

NTNU

DECLARATION OF CO-AUTHORSHIP

Declaration of co-authorship is to be included when a thesis containing papers written by more than one author must include this signed declaration that describes the contribution of the candidate and the co-authors of each of the papers. It must be possible to identify the candidate's independent contribution in the work. *(cf. the PhD regulations section 10 or dr.philos regulations section 3).*

Name of candidate: Kathrine Vestues

Title of dissertation: Using digital platforms to promote value cocreation: A case study of a public sector organization

Publication: Vestues, Kathrine; Mikalsen, Marius; Eric Monteiro (2021), *Using digital platforms to promote a service-oriented logic in public sector organizations: A case study*, Accepted to the 54th Hawaii International Conference on System Sciences.

Description of the Candidate's contribution: Kathrine did the data collection, data analysis, and wrote most of the paper. Marius Mikalsen and Eric Monteiro contributed to the writing of the introduction, theoretical framework, as well as providing comments.

Statement by the co-author:
I hereby confirm that the doctoral candidate's described contribution to this paper is correct, and I consent to including the paper in the named dissertation.

Place, 22.02.2021

[illegible]

Eric Monteiro