Lassenius, Casper; Mohagheghi, Parastoo; Seide Molléri, Jefferson

# Bringing it home : successful backsourcing of software development in the public sector

*Published in:*
Empirical Software Engineering

*DOI:*
10.1007/s10664-025-10722-1

Published: 01/12/2025

*Document Version*
Publisher's PDF, also known as Version of record

Empirical Software Engineering (2025) 30:170
https://doi.org/10.1007/s10664-025-10722-1

# Bringing it home: successful backsourcing of software development in the public sector

**Casper Lassenius**[1,2] · **Parastoo Mohagheghi**[3] · **Jefferson Seide Molléri**[4]

Received: 30 August 2024 / Accepted: 19 August 2025


**Abstract**

Reversing an outsourcing decision and retaking control of the development and operations of IT systems can be a viable option for organizations facing outsourcing-related challenges, such as problematic vendor relationships, high costs, or the recognition of software development as a core competence. Despite its increasing practical importance, backsourcing has received little attention in the software engineering literature. This paper presents an in-depth case study of a large public organization that successfully backsourced the development of the majority of its large portfolio of software systems. Based on interviews with 61 representatives of the backsourcing organization and its vendors and an analysis of corporate documentation, we present the rationale for backsourcing, the six main guiding principles shaping the approach, and the outcomes and lessons learned from the initiative. The primary reason for backsourcing in the case organization was cost reduction. The IT department led the effort, granting various application areas considerable autonomy in the practical implementation of backsourcing. The effort was characterized by close and constructive collaboration with the outsourcing vendors, who were being phased out, an incremental approach to assuming responsibility for the applications, and the concurrent implementation of agile development at scale. Over three years, the organization successfully backsourced over 100 systems and established an agile development organization with over 300 people. As a result of backsourcing, the organization reported an increased sense of system ownership, increased personnel motivation, faster deliveries, and better quality-in-use. We were unable to find clear evidence of significant cost savings. Key challenges included managing the change effort, building and retaining the necessary competencies, and addressing organizational tensions resulting from the IT department's central role in the backsourcing effort.

**Keywords** Backsourcing · Software engineering management · Large-scale agile software development

---

Communicated by: Slinger Jansen.

---

Extended author information available on the last page of the article

## 1 Introduction

Software development outsourcing became a popular strategy starting in the late 90s, for both private and public sector organizations seeking to modernize their systems and improve their digital services. Outsourcing can provide access to specialized skills, save costs, and facilitate increased flexibility in project management. Outsourcing allows organizations to focus on their core functions and competencies, while delegating non-core tasks to external vendors (see, e.g., Dibbern et al. 2004). However, outsourcing can also introduce challenges related to software quality and control of the development of systems and lock organizations into using outdated technology or vendors with competence issues, all of which can ultimately impact the performance. Challenges with vendor relationships, unrealized cost savings, or changes in strategy or management can make organizations reevaluate existing outsourcing relationships (Chakrabarty and Wakefield 2010; Molléri et al. 2023).

As a response to problems with outsourcing, backsourcing — the process of returning outsourced functions in-house — has emerged as a potential solution (von Bary 2018; Wong 2008).

While the literature on outsourcing in software engineering is abundant, often under the heading of global software engineering, the topic of backsourcing software development has barely been broached. A recent literature review found only a handful of cases in the software engineering literature, none of which provided deep insights into the phenomenon (Molléri et al. 2023).

As backsourcing concerns moving the development and maintenance of systems to a new organization, the topic is very relevant to software engineering, and we think that a sound empirical basis for backsourcing is needed. To contribute, we present an in-depth case study of successful backsourcing in a large public sector organization. We conducted the study using predominantly qualitative research methods, including semi-structured interviews with key stakeholders and an analysis of internal documentation. In addition, we analyzed internal databases on the frequency of releases, cost, and quality-in-use. Our findings reveal that the main reasons for backsourcing were the need to reduce costs, followed by the desire to take back control. We describe the approach to backsourcing using six overarching principles and discuss the challenges, outcomes, and lessons learned from the backsourcing effort.

This paper contributes to the existing literature on backsourcing by providing the first in-depth case study exploring the reasons, processes, outcomes, and challenges of backsourcing in the field of software engineering. The insights gained from this study can therefore aid other organizations considering backsourcing, as well as provide a baseline case to which further case studies can be compared and contrasted by other researchers, thus contributing towards theory building on software development backsourcing.

The rest of the paper is structured as follows. Section 2 discusses the extant literature on backsourcing in software engineering and IT. Section 3 presents the case organization and the background for the study from that perspective. Section 4 discusses the methodology employed. Due to their length, the results are split into several sections. Section 5 discusses the reason behind the backsourcing decision, Section 6 discusses the approach taken to backsourcing written up as six guiding principles, Section 7 reports the challenges encountered during the outsourcing, Section 8 presents the results of the outsourcing, and finally, Section 9 discusses the lessons learned from the effort. The presentation of the results is followed by a discussion and comparison to the literature in Section 10, and the conclusions are presented in Section 11. The interview guide used is exposed in Appendix A.

## 2 Related Work

### 2.1 Defining Backsourcing

We define backsourcing as *the process of bringing previously outsourced activities back in-house* (Molléri et al. 2023). Stated differently, it is the process of reversing an outsourcing arrangement, retaking the responsibility for and conducting the activities of the outsourced entity in-house; it is the process of transitioning from outsourcing to insourcing (Leyh et al. 2018). In our context, we consider the backsourcing process to start with a decision to backsource, and end when the backsourced activity has been insourced, meaning that the ownership and responsibility for the activities are back in-house.

In the software engineering literature, backsourcing has been referred to more generally as a "return from collaboration" (Šmite et al. 2014). However, this definition is rather vague and can be interpreted to mean that collaboration with the vendors that used to be outsourcing partners would end, which is not necessarily the case. In the broader literature, backsourcing sometimes overlaps with related concepts such as insourcing, reverse outsourcing, reshoring, and backshoring (Bergkvist and Fredriksson 2008; Nujen et al. 2015; von Bary 2018). These terms address different aspects of the sourcing relationship, such as the geographic location or whether the service is delivered by an external provider or within the same company (Šmite et al. 2014).

To ensure consistency in using sourcing-related terminology (e.g., outsourcing, insourcing, multisourcing), we adopt the empirically grounded taxonomy proposed by Šmite et al. (2014). However, for terms such as in-house and vendor, we use the definitions in our earlier work (Molléri et al. 2023) rather than those proposed in Šmite et al. (2014) (i.e., internal for in-house, external provider for vendor).

### 2.2 Backsourcing in ICT

While outsourcing has received much research attention in the global software engineering literature, the act of reversing it has not. There are a few studies of backsourcing in the broad field of IT, but virtually no studies focusing on software engineering. In fact, a recent systematic literature review identified only three studies on the topic in the field of software engineering, and not a single detailed case study (Molléri et al. 2023). Looking at more recent literature, published after 2021, did not change the overall picture.

In the following sections, we summarize the lessons learned from the literature on backsourcing in the IT field, discussing reasons, backsourcing activities, attributes, practices, and reported outcomes, based on a systematic literature review covering studies until 2021 (Molléri et al. 2023), as well as later studies (Falasco et al. 2024; Rajini and Kaluarachchi 2024; Misztal 2024).

### 2.3 Reasons

Not surprisingly, the reasons companies started to retake an outsourced activity in-house were mainly negative, signifying some level of dissatisfaction with an existing outsourcing relationship.

The three main reasons for backsourcing in the literature were quality issues, high cost, and lack of control of the outsourced process (Kotlarsky and Bognar 2012; Nujen et al. 2018; Barney et al. 2013; Butler et al. 2011; Hirschheim and Lacity 2006; Moe et al. 2014; Solli-Sæther 2016; Wong 2008). These findings are supported by more recent work identifying drivers of backsourcing decisions (Falasco et al. 2024; Rajini and Kaluarachchi 2024). Other reported reasons included a poor client-vendor relationship, changes in the client's strategy, lack of vendor competence, new regulations, and outdated technology (Lacity et al. 2007; Leyh et al. 2018; von Bary 2018; von Bary et al. 2018; Wong 2008). The decision to backsource was usually not based on a single reason but a combination of several of the factors mentioned above. One of the three main reasons mentioned above seems to be involved in all reported backsourcing decisions in the literature.

As backsourcing is a major decision with significant implications for the organization, there are obviously factors that argue against it. They include a lack of internal capabilities, high switching costs, and high dependency on the external provider (Barney et al. 2013; Kotlarsky and Bognar 2012; Moe et al. 2014; Nujen et al. 2018). Interestingly, the systematic literature study by Molléri et al. (2023) identified only one case of failed backsourcing (Petalidis 2018), in which the dependency on the vendor kept the client locked into an undesired contract. However, this might be explained both through publication bias — cases of negative backsourcing efforts might not be reported, and by search criteria — they are unlikely to use the term backsourcing or insourcing.

### 2.4 Process

In our literature review of backsourcing (Molléri et al. 2023), we pointed out that the backsourcing process seems to vary greatly due to differences in the context and the circumstances in which backsourcing occurred. Despite this, we were able to identify a set of activities that were shared across backsourcing cases. We grouped them into five categories, as follows:

1. *Change management:* comprises activities related to planning the backsourcing, communicating the decision internally, and managing the overall change process (Hirschheim and Lacity 2006; Petalidis 2018; Wong 2006). These also include post-outsourcing activities like monitoring the outcomes of backsourcing, and planning further expansions (Moe et al. 2014; Wong 2006).
2. *Vendor relationship management:* consists of activities related to contract termination (Butler et al. 2011), including in some cases to draw up a backsourcing agreement (Kotlarsky and Bognar 2012). These activities aim at ensuring a smooth backsourcing process by handling risks associated with the termination or change of the vendor relationship.

3. *Competence building:* focuses on acquiring the necessary competence for knowledge transfer and assuming responsibility for the previously outsourced activities (Kotlarsky and Bognar 2012; Moe et al. 2014; Petalidis 2018). Training own employees is an essential activity at this stage as it ensures the skills and competencies are transferred back home (Moe et al. 2014; Nujen et al. 2018, 2015).
4. *Organizational build-up:* consists of building a new organization that will take over the responsibility for the outsourced activities. Essential activities related to this sub-process are setting up an in-house infrastructure (Butler et al. 2011; Kotlarsky and Bognar 2012; Petalidis 2018), and maintaining a pool of skilled professionals by means of recruitment and personnel retention (Butler et al. 2011; Moe et al. 2014).
5. *Transfer of ownership:* comprises the activities for managing the transferred knowledge (Moe et al. 2014; Wong 2006), and ensuring the continuity of services in-house (Butler et al. 2011; Kotlarsky and Bognar 2012).

These five categories outline areas in which activities seem necessary in all backsourcing cases, based on the reviewed literature. In Molléri et al. (2023), we provide a detailed description of these categories, along with associated activities, attributes, artifacts, and practices. Since the literature on backsourcing, in particular in software engineering, is rather scarce, we do not claim that this list of categories is complete; indeed, future work might well extend and improve it.

### 2.5 Outcomes

Both positive and negative outcomes of backsourcing have been reported. Positive ones include recovering knowledge transferred to the vendor under the outsourcing agreement (Nujen et al. 2015, 2018), lower costs to operate and maintain the systems (Kotlarsky and Bognar 2012; Wong 2008; Solli-Sæther 2016; Hirschheim and Lacity 2000, 2006), better control over the activities (Moe et al. 2014), acquisition of skilled professionals to cover the needs for specific competencies (Kotlarsky and Bognar 2012; Wong 2006, 2008; Butler et al. 2011; Moe et al. 2014), and better service quality (Moe et al. 2014).

Some studies (Kotlarsky and Bognar 2012; Moe et al. 2014) reported post-backsourcing benefits that were achieved long after the company took ownership of the outsourced service. Such benefits include market advantages due to proximity to the customers, new business competencies by hosting services in-house, and expanding the business using new offshore organizations.

The literature also reported negative outcomes of backsourcing. Many were associated with extra costs incurred through the backsourcing process, such as hiring, training, and relocating personnel (Moe et al. 2014; Solli-Sæther 2016). This is indeed not unexpected, as it is clear that making such a large change will incur costs. Moreover, a failure to build a new organization could lead to the loss of crucial competencies and interruption of services (Butler et al. 2011).

Another recurring theme in the literature is the difficulty of maintaining a working vendor relationship during backsourcing. Companies that experienced a relationship breakdown had to terminate the outsourcing agreement early, which came at a cost. Without the vendor's support during the backsourcing process, clients faced a risk of unsuccessful knowledge transfer and disruption of the development process (Butler et al. 2011; Kotlarsky and Bognar 2012), risking project failure.

Building a new organization to take over the work previously handled by a vendor is another major challenge. There are reported cases of companies that considered avoiding backsourcing due to factors such as missing staff, lack of technical knowledge, and lack of support from the vendor in knowledge transfer (Barney et al. 2013; von Bary 2018). One logical but rather aggressive strategy was to recruit from the vendor (Butler et al. 2011; Hirschheim and Lacity 2006; Kotlarsky and Bognar 2012), as the vendor's personnel already has the necessary skills and competencies. This strategy has the risk of alienating the vendor, making knowledge transfer even more difficult, and might require negotiating better work agreements, as the vendor might make competitive counteroffers to keep their talent (Butler et al. 2011; Kotlarsky and Bognar 2012).

Most of the backsourcing studies reported success stories and positive outcomes. It is worth noting that the assessment of success in the existing studies was based solely on the perceptions of managers and other stakeholders participating in the process. We could not identify any papers that would have provided a quantitative analysis of the cost-benefits related to backsourcing or quantified other effects, such as quality improvements. Further, the length of the backsourcing process was not clearly specified in the literature, but most papers report cases that spanned multiple years.

Another gap in the literature is related to defining what successful backsourcing entails. We thus need to define it ourselves. In this paper, we define backsourcing as successful when the *responsibility for the object of backsourcing (a system, set of systems, organizational function, etc.) has been transferred back in-house, and the (new) internal organization is able to carry out the previously outsourced activities.*

We discuss this in more depth in Sections 8 and 10.

### 2.6 Backsourcing in Other Fields

Outsourcing and backsourcing as phenomena, are naturally not restricted to the fields of information technology or software development. Indeed, backsourcing has been discussed across fields such as supply chain management, manufacturing, international business, and healthcare. In these domains, it is often a strategic response to the limitations or failures of prior outsourcing arrangements. Common reasons for backsourcing in healthcare include rising operational costs, quality concerns, lack of control, or shifting organizational priorities (Young and Macinati 2012). In the public sector, backsourcing was motivated by managerial concerns rather than political ambitions (Jansson et al. 2021). In contrast, private-sector firms may be driven by the need for innovation, proximity, or strategic agility (Misztal 2024).

The backsourcing process in these domains involves efforts to manage organizational change, rebuild internal structures, and reintegrate previously outsourced knowledge. This reintegration is particularly complex due to the lack of knowledge symmetry between clients and former vendors, requiring organizational planning and capability development (Ejodame and Oshri 2018; Kotlarsky and Bognar 2012). In manufacturing and logistics, this included rehiring or retraining staff, rebuilding supply chain infrastructure, and re-establishing in-house competencies (Nujen et al. 2015).

Reported outcomes of backsourcing vary across sectors but generally include improved control, cost savings, and strategic flexibility. Some studies also indicate positive market reactions to backsourcing decisions, reinforcing its viability as a deliberate and valuable strategic shift (Nicholas-Donald and Osei-Bryson 2017). Overall, backsourcing should not be viewed in isolation, but rather as part of an evolving sourcing strategy shaped by both internal and external dynamics (Jansson et al. 2021).

While a full comparison of backsourcing beween software engineering and other fields is outside the scope of this paper, we note that there are similarities outside software engineering and IT regarding the reasons behind bacsourcing, the process, and the outcomes. A more extensive comparison between the IT and software engineering literature, and research in other fields, is left for future work.

## 3 Case Background

As the software engineering literature did not contain any detailed accounts of backsourcing, and we were knowledgeable about, and able to get access to a large case of backsourcing in the public sector in Norway, we decided to conduct an in-depth single case study. In this section, we present the selected organization, focusing on the situation before backsourcing was initiated.

The case organization, the Norwegian Labour and Welfare Administration, or NAV, is responsible for implementing labor market policy and providing social security and pension services to the population of Norway. NAV manages one-third of the Norwegian state budget and offers more than 40 different benefits related to, e.g., unemployment, maternity leave, sickness and disability leave, and retirement pension. In 2022, NAV had more than 22,000 employees, of which 850 were in the IT department.

NAV was established in 2006 by merging two public organizations that handled labor and welfare services and by moving the responsibility for several welfare benefits from local authorities to NAV. The purpose was to create a single organization that supports citizens in all stages of their lives and to increase the synergy between the services offered by the public sector in Norway. The organization handles more than three million benefit cases annually, and IT systems are used in both the back office to manage the benefits and to provide services directly to the population through digital interfaces. End-users include private citizens, employers, NAV employees who handle benefit cases, and other organizations that exchange data with NAV or participate in value chains across organizations, both in the public sector, such as the tax offices, police and municipalities, and in the private sector, for example, for providing health care services and labor market activities.

At the top level, NAV was organized into eight departments and a separate unit for internal audits. The departments included the front-facing departments for benefits management, work and labor services, communication, and knowledge. Internally focused departments included HR, economy and management, IT, and the legal department.

The IT department was responsible for developing, maintaining, and operating the IT systems, whereas the other departments owned the systems. Ownership in this context meant defining strategic and operational requirements for the systems, and verifying their implementation, assigning budgets and resources, and helping development teams to clarify and document what the systems do, similar to the role of product owner in Scrum (Schwaber and Sutherland 2020). In the rest of this paper, we refer to these non-IT departments as *business departments*.

### 3.1 NAV IT

At its founding, NAV inherited a portfolio of approximately 300 IT systems, which were developed and maintained by a combination of internal developers, consultants, and external vendors. The new organization faced a major challenge in taking control of these systems and the approximately 600 related contracts related to their development and maintenance. In addition, several large legacy systems were considered to be in dire need of modernization. To complicate matters further, the earlier welfare organization had experienced a major IT project failure, even leading to public hearings in the parliament, which left deep wounds in the organization.

As a result of these factors, the IT director of NAV decided to outsource the maintenance and development of most of its software systems. Additional justification an underlying assumption that the best knowledge is available in the open market and that NAV should or could not compete in this market. Interestingly, this decision to outsource was made by the IT organization, and was not considered strategic at the time[1]. Thus, as a result, NAV took the responsibility for the operation of the systems, while development and maintenance were outsourced.

### 3.2 IT Organization

Prior to backsourcing, the IT department was organized around its three main responsibilities: to plan, build, and run the systems, as illustrated in Fig. 1. Planning included managing the IT budget and capacity, project, and release planning. Building included contract management and participation in development and maintenance activities on the client side. Running consisted of acceptance and deployment of releases, operating the applications and the infrastructure, and first-line user support. The portfolio was owned by the board of directors, who prioritized activities and assigned budgets.

The development of new systems and features was primarily organized as projects planned in collaboration with the respective vendor, and had a traditional subcontracting project setup. Many roles were doubled, e.g., both the client and the vendor had their own project and team managers, and testing was conducted by both the vendor and the client. Vendors provided whole teams instead of individual experts. Vendor teams were physically seated on the client premises but in their own areas, separate from the NAV employees, and the interactions were mainly handled through formal channels. Figure 2 shows the project setup and related organizations and roles.

As seen in the picture, a project had a business owner from a business department and internal project resources from the IT department. The vendor provided the development

[1] Interview with the first director general of NAV

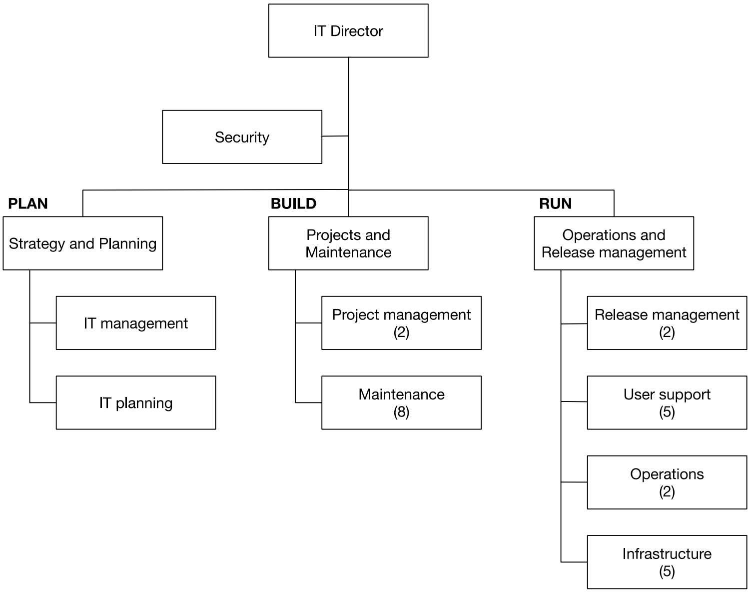



**Fig. 1** Simplified figure of NAV IT Organization (Numbers indicate the number of sub-organizations under the departments)

teams with their structure and leadership. Additionally, the IT organization provided project support through release planning and management, as well as assisted with testing.

Experts from business departments, who had in-depth knowledge of benefits such as pensions or childcare, analyzed laws and regulations, specified requirements, funded development and maintenance efforts, and functioned as requirement owners in the projects. The IT department served as a liaison between the business departments and the vendors.

The main roles in the IT department that assisted the business departments in contact with vendors were functional advisors who had knowledge of the business domain and assisted with tasks such as analysis and estimation, and technical advisors who focused on non-functional and technological aspects of the systems. In addition, software architects assisted projects with architectural decisions, project managers with project execution, and testers with integration and acceptance testing. The operations department was responsible for first-line support, system operation, and incident management. Employees were assigned either full or part-time to the projects, and an employee could simultaneously be involved in several projects.

The vendor teams consisted of team leads, developers, testers, and Scrum masters since the vendors mainly used the Scrum methodology (Schwaber and Sutherland 2020) in the development phase. Vendor teams outside of the projects handled maintenance through long-term maintenance contracts. In this model, deployment was primarily conducted in large, quarterly releases on weekends, necessitated by the requirement for downtime during the deployment process.

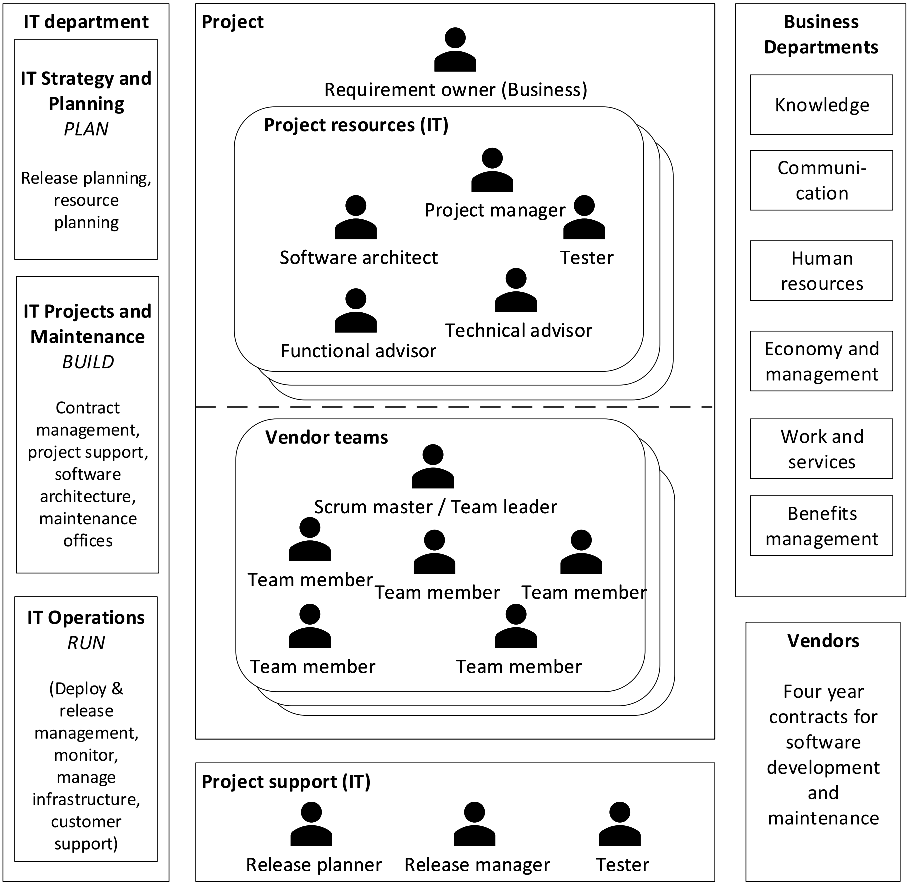



**Fig. 2** Organization of Development Projects (from Mohagheghi and Lassenius 2021a)

### 3.3 Application Areas

From a management and organizational perspective, the applications were grouped into application areas. An area grouped applications together based either on their business domain, as a common service, or their internal role. Typically, the applications in an application area were handled through the same contract. Table 1 shows an overview of application areas that were initially outsourced and subsequently backsourced.

The area names reflects the type of applications: the UI/Self service area contains the systems that provide the external interfaces visible through the official web site and used by private citizens in Norway. The Benefit systems A and B represent systems managing the two largest benefit areas, family and sick leave, and pension and diability benefits. The Knowledge area contains systems that develop various statistics for government and internal use. The Shared systems A-E represent various systems that are used by several benefits for case management. Finally, the IT area represents systems that are owned by the IT department and provide platform and support services.

**Table 1** Application areas

| Area | Owner | Systems | People | Description |
|---|---|---|---|---|
| UI/Self-service | Communication | >70 | 25 | Generation of the organizational website and interfaces for private citizens |
| Benefit A | Benefits | <5 | 50 | Benefits related to family and sick leave. Large legacy systems. |
| Benefit B | Benefits | 1 | 80 | Pension and disability benefits. Large monolith with 7 components |
| Knowledge | Knowledge | >30 | 50 | Generation of statistics for government and organizational use |
| Shared A | Benefits | <5 | 20 | Off-the-shelf component for generating letters and documents, and an archive and distribution solution |
| Shared B | Benefits | <5 | 25 | Large legacy case management system |
| Shared C | Benefits | >30 | 40 | Shared databases with user data from internal and external sources |
| Shared D | Economy | <10 | 20 | Shared off-the-shelf component tailored to collect case management results of work-related benefits (such as unemployment) and manage payments. |
| Shared E | Economy | 3 | 60 | Components collecting case management results of all benefits except work-related ones, calculating and managing payments. |
| IT | IT | >10 | 40 | IT internal systems, e.g., security, data exchange, integration, and a service-oriented platform. |

The owner column names the department that owned the systems in the area, i.e., the Communication Department, the Benefits department, etc. The number of systems represent the number of distinct IT systems managed by the area, and the number of people is the approximate number of external and internal people involved in the area when the backsourcing process started. Finally, the description column provides a short desdcription of each area.

### 3.4 Contract Model

The case organization used a mixture of contract types for different types of work: paying vendors by the hour for non-development activities (such as helping the client in the analysis phase), using an agile variant of a target-price contract for new development or extensions of the existing software, and a fixed-price contract model for software maintenance.

The target-price contract model was designed with agile development in mind. At the contract time, the client was supposed to provide a rough description of the software, and the contract was signed based on overall goals in the form of a target picture and an assumed scope in terms of time and cost. Each delivery was later defined incrementally: the client specified user stories at the beginning of a development iteration, the vendor provided and estimated solution descriptions, and the client prioritized them continuously. User stories were realized using a target price based on the estimated cost, with shared risk for cost overruns (often 50-50) and without an upper limit. This meant that vendors were paid for at least 50% of any cost overrun. In practice, there was little deviation between the estimated and actual time spent. The client preliminarily approved the user stories at a control point in each iteration and the whole delivery according to the delivery plans.

Development contracts with vendors were signed for four years with the possibility of extending them twice with an optional year; thus, for a maximum of six years. The client specified the required competency in the contract, and vendors were expected to demonstrate capacity and the necessary competence during the contract period.

## 4 Methodology

To study backsourcing in the selected organization, we opted for a case study approach. A case study is appropriate when studying a contemporary phenomenon in-depth in its real-life context when the boundaries between the phenomenon and context are not clearly evident (Yin 2009). This is the case in our study, as, e.g., the relationship between the actual activities and results of backsourcing are difficult to distangle from the ones involved in building up and implementing new agile development processes. Further, as the expectant literature lacks in-depth case studies on backsourcing in software engineering, even a single case study can provide valuable insights into this mostly unexplored phenomenon.

To this end, we conducted a case study with the overall goal to **describe and understand the backsourcing effort in the case organization**. We posed the following more specific research questions:

- RQ1: Why did the organization initiate backsourcing of software development and maintenance?
- RQ2: How did the backsourcing proceed?
- RQ3: What challenges did the organization face during backsourcing?
- RQ4: What were the outcomes of the backsourcing effort?
- RQ5: What lessons learned did the organization identify?

To answer the research questions, we selected a case organization based on public knowledge, as it had received wide visibility in the national press as a successful case of backsourcing. In addition, we gained in-depth access by involving an employee of the organization, the second author of the research, which allowed us to conduct an information-rich study (Yin 2009). Thus, the case selection was based on a purposeful sampling logic (Patton 2014).

We collected data primarily through semi-structured interviews, which we complemented with an analysis of corporate documents such as presentations, reports, meeting notes, and quantitative data on cost, quality, and software deliveries. The use of several different sources of data provided us with the opportunity for data-source triangulation (Jick 1979). We analyzed the data using qualitative analysis (Corbin and Strauss 2014; Patton 2014) and simple descriptive statistics and visualizations for the quantitative data.

Runeson and Höst (2008) classifies case studies into exploratory, descriptive, and explanatory, based on their overall objectives. This study is mainly descriptive, describing how backsourcing was conducted in a large public sector organization. However, it also has explanatory aspects, as we attempt to describe the relationship between the backsourcing activities and the outcomes the organization experienced based on qualitative and quantitative data analysis.

Next, we describe the data collection and analysis in more detail.

## 4.1 Data Collection

### 4.1.1 Interviews

We collected data through interviews between October 2019 and August 2021. To gain a deep understanding of the backsourcing effort and to be able to triangulate our findings (Jick 1979), we interviewed several people in different roles. Table 2 shows the number of interviewees and their roles, and the number of interviews performed. As we explored a new phenomenon and wanted to create an in-depth understanding, we opted for a semi-structured qualitative interview format (Legard et al. 2003).

We started our interviews by identifying a small set of key informants (Patton 2014) at the organizational level, which included the current and previous IT directors. This gave us an understanding of the rationale behind the decision, high-level decision-making, and the overall process for organizing the effort.

Based on these interviews and in collaboration with the IT director, we selected the next set of interviewees who could describe the backsourcing effort in more detail for the application areas.

To get a good insight into the backsourcing process for each application area, we interviewed people in different roles: the product owner or product coordinator (a role coordinating several product owners for a business domain), the team lead, and team members in other roles. Furthermore, we interviewed managers at various levels and individuals responsible for contract management. For each application area, we interviewed between three and seven people.

Furthermore, we conducted one interview with each of the six vendor companies to get their perspective. These interviews had one or two participants and covered the application areas in which they were involved.

We conducted the interviews using a general interview guide approach (Legard et al. 2003; Patton 2014) to be able to adapt the specific questions to the roles and individual experiences of the interviewees while simultaneously making sure that the relevant topics were explored. An overview of the interview questions can be found in Appendix A. The interview guide was sent to the interviewees in advance, requesting their approval to participate and be recorded during the interviews.

**Table 2** Interviewees and their roles

| Role | Interviewees | Interviews |
|---|---|---|
| Directors | 4 | 4 |
| Section and office managers | 6 | 6 |
| Team leads | 10 | 10 |
| Product owners or product coordinators | 14 | 13 |
| Team members in other roles: developer, tester, tech lead, agile coach | 11 | 10 |
| Contract management and analysts | 5 | 5 |
| Union representatives | 2 | 1 |
| Vendors | 9 | 6 |
| **Total** | **61** | **57** |

We conducted individual interviews, with the exception of six, which included three with vendor representatives, one with two representatives from employee organizations, and two where a team member accompanied product owners to the interview.

We had planned to conduct all interviews in person, and all interviews in 2019 were conducted on-site. However, the COVID pandemic forced people to work from home since March 2020 and as a result, most interviews during 2020 and 2021 were conducted remotely using video conferencing.

The interviews lasted 45-145 minutes, and all but ten were done jointly by the first and second authors. At the start of each interview, we presented the study's goals and asked for consent to record, store, and use the data for our research. During the interview, one researcher took the role of interview lead, asking most of the questions, whereas the other researcher took notes and asked clarifying questions when he or she thought it was needed. The interviews were recorded and transcribed for analysis.

#### 4.1.2 Additional Data Sources

In many interviews, our interviewees referenced and showed corporate material, such as presentations and spreadsheets, and in some cases, they drew pictures on a whiteboard. We collected these materials for analysis in addition to the interview transcripts.

Other internal documents we received included organizational charts before, during, and after backsourcing, presentations, reports, and meeting notes used to support decision-making and evaluations related to backsourcing. In addition, we collected reports prepared by external consultancies and media coverage related to the change.

Finally, we had several meetings with people in different roles who were not directly involved in the backsourcing effort, such as software architects, release support, management support, and office leads for collecting quantitative data and interpreting the analysis results. Through these meetings, we were given data on all releases done since 2016, all incidents reported in the field from 2016 onwards, and various calculations aimed at analyzing the financial aspects of backsourcing. Additionally, we received data on the number of employees in different IT-related roles since 2016. This additional data collection was done during 2022 and 2023, and required several meetings with personnel from the case organization to identify, interpret, and clean the data.

### 4.2 Data Analysis

#### 4.2.1 Qualitative Data Analysis

The major part of the analysis was done with the help of the qualitative data analysis tool NVivo (QSR 2020). We defined three a priori high-level temporal categories: pre-transformation, intra-transformation, and post-transformation. Under these main headings, we thematically coded passages relevant to our research questions. These included passages that described the reasons behind backsourcing (RQ1), descriptions of how the backsourcing proceeded (RQ2), challenges encountered during backsourcing (RQ3), results and outcomes (RQ4), and lessons learned (RQ5). We used open coding under these predefined high-level categories, letting detailed codes emerge from the data, as suggested by Corbin and Strauss (2014). During this initial coding round, we created codes describing the state-

**Table 3** High-level categories and number of codes

| Category | Coded passages |
|---|---|
| Pre-backsourcing status and decision | 569 |
| Backsourcing implementation | 830 |
| Backsourcing outcomes | 1094 |
| Lessons learned | 577 |

ments the respondents gave at a detailed level using the constant comparative method (Corbin and Strauss 2014). We created tentative codes as we read through the transcripts. When coding new transcripts, we compared the new relevant text passages with the existing codes. Based on this, we either coded the new passage using an existing code, renamed the code to describe the phenomenon better, or created a new code to describe an additional phenomenon or aspect. Naturally, this way of coding quickly creates many codes that can be challenging to manage. To alleviate this, while coding, we started to identify potential categories that grouped several codes. Also, for the categories, we constantly reviewed and altered them to keep the code book manageable and logical. In addition to keeping the code tree manageable, the categories helped raise the abstraction level for analysis and sensitized us to identify new codes. Table 3 shows the number of coded passages under the top-level categories, and Table 4 shows an example of interview passages and their related codes and themes.

**Table 4** Coding examples showing high-level codes for category and theme

| Category | Theme | Code | Passage |
|---|---|---|---|
| Backsourcing implementation | Vendor relationship management | Good collaboration | *So I really liked it because it was very clear, and, yes, we cooperated very well indeed. The supplier ... they were very, very accommodating.* |
| | | Informing vendors early | *Yes, we were informed early on ... that is, when the new IT director came in, we had a good dialogue with him about what the strategy was.* |
| | | | *...it was very clearly communicated centrally from NAV what the goals of NAV were and how NAV envisioned the future within the development department. And within that, it was also clear that they eventually wanted to take over responsibility for the areas we had a management contract.* |

We conducted the initial coding in parallel with data collection and held several meetings to discuss the evolving codebook. The first and second authors coded and discussed the first interviews, after which the second author did most of the initial coding. The first author checked and commented on the coding during the process.

Following the initial coding and categorization, the authors jointly discussed and structured the resulting large code tree into a cleaner structure, aggregating, renaming, merging, splitting, and moving codes as necessary, identifying key themes, a process also referred to as axial coding (Corbin and Strauss 2014). To aid in interpreting the higher-level codes, we did not remove low-level codes but kept them as leaves in the code tree. Due to the size of the codebook, we do not expose it further here. A high-level version of the code tree is publicly available[2].

### 4.2.2 Quantitative Data Analysis

In addition to the interviews and corporate documents, we received two large data sets from the organization. The first dataset contained a list of all deployments done for all systems between 2016 and 2020. The second set had data on externally reported issues to the user support system. For our analysis, we imported the data sets into a relational database. Both data sets required some manual work to be usable for analysis. The third author cleaned, analyzed, and reported the results based on the quantitative data, with support from the other two authors. The collection and cleaning of the quantitative data was done during 2022 and 2023, and involved several meetings with personnel from the case organization, as locating, interpreting, and combining data from various sources required substantial effort from both the researchers and the organization.

The dataset on deployments contained information at the application level, and we manually had to map the applications to their related application areas for analysis. We investigated trends in the number of deployments, their timing, and any significant changes over the observed period. Further, we plotted the changes in the number and timing of deployments and correlated the identified patterns with strategic shifts in the backsourcing process.

The data on externally reported issues was more challenging, as the organization had changed both the reporting system and reporting process, including fault categorization, during the period we were interested in. Due to this, we only show the most recent dataset with a starting date of June 2017 in our analysis. We extracted data on the frequency, duration, and nature of the reported issues to the user support system. Our analysis involved creating histograms and time-series analysis to understand the distribution and evolution of incidents over time. Special attention was given to critical incidents, assessing their impact and resolution timeline.

While we would have been interested in analyzing the pre-release quality during development before and after the backsourcing effort, the development teams had stopped reporting issues identified during development, making such an effort unfeasible.

The following sections present the results organized around the research questions. Section 5 discusses the reason behind the backsourcing decision, Section 6 presents the six principles used in the backsourcing approach, Section 7 reports the challenges encountered, Section 8 presents the results of the outsourcing, and Section 9 the lessons learned.

[2] https://doi.org/10.6084/m9.figshare.29264501.v1

## 5 Reasons for Backsourcing

In this section, we answer the first research question: *Why did the organization initiate backsourcing of software development?*

### 5.1 Reduce Cost

There was one salient driver for embarking on backsourcing: the perceived high IT development and maintenance cost and the explicit need to reduce it. While already a known problem, widely discussed internally, additional impetus came from an external consultancy report ordered by the NAV director that criticized the high costs of development and how software was managed[3]. The IT director at the time stated:

> *The main argument for the new sourcing strategy was plain and simple that we needed to reduce the cost.* — IT Director

The main factors explaining the high cost were the price of consultants compared to internal personnel and process inefficiencies related to the contracting and development model used. The first factor is easy to analyze: according to the IT director, the savings per developer was 1 million NOK (ca 100 000 USD) per year. However, the second factor, process inefficiencies, is more challenging to quantify.

Our informants mentioned several issues that created extra costs due to using consultants, including the need for bidding, creating detailed requirements for estimation, redundant activities and roles, and additional bureaucracy required to manage the outsourcing process. In the words of the IT director at the time:

> *and then you have the increased process efficiency in that you could start working according to agile principles, in closer collaboration with the business side.*
>
> — IT director
>
> One of the vendors also raised the extra work involved in the old way of working:*The contract we had was divided into three parts. You had fixed price maintenance, and then you had construction at what is called a target price, where you have to estimate and you can say a lot about it, and then there was assistance. Assistance was things that were not construction. It's that kind of planning, design, that kind of thing Yes. In 2015, we spent more time assisting (the client) than we did on construction. In other words, more time is spent on planning and doing everything around and reporting than was spent on making things.* — Vendor representative

The way the outsourcing was handled led to many double roles: both the client and the supplier had their own project managers, team leads, test leads, testers, and contract managers. The cost for this was naturally carried by the client.

Maintenance was handled through fixed-price maintenance contracts that included bug fixing. However, many interviewees stated that much overhead was involved in negotiating

[3] Report by McKinsey, 2014

whether a necessary change was a bug fix falling under the maintenance contract or a new development or change, which would be priced separately.

Finally, there was a sense in the organization that the consultants overcharged — this was seen in statements like “everyone knows that a consultant chases hours”, and the fact that the minimum billing time of a consultant was 30 min even if the actual work involved to do a change was only a few minutes.

## 5.2 Secondary Reasons

In addition to the main impetus of reducing cost, the respondents mentioned several secondary motives behind backsourcing. These included the need to get full ownership and control of the IT systems in-house, improving the delivery speed, and reducing technical debt.

Several respondents mentioned the need for the organization to take ownership of the IT systems and to be less reliant on consultants. For example, an IT department lead stated:

> *but it was also ownership you can say there are two interrelated streams here. On the one hand, what do we want to accomplish in the large and how do we want to work. Until now, we have not been able to take ownership. And then we don’t want to continue to work with projects that have deliveries as their goal. We want to work with end-user value and we want to develop products.* — IT Department lead

> While there was a strong understanding of the need to gain control, there was quite some frustration related to being too reliant on the vendors and not being able to understand, let alone challenge them, often leading to what was considered unnecessary extra cost, as the following quote illustrates:*[The] suppliers provided detailed solution descriptions that we, not being developers, could not understand and therefore had to trust [the vendors] And then [the solution] was delivered, and the intention that they would take responsibility for and correct errors of their own accord [did not realize] as they were responsible for maintenance, they just pointed out that “Yes, but you approved the solution specification, you wanted us to make it this way. Therefore, we will treat this as a change request.”* — Middle manager

> In addition to bringing extra cost, the long discussions on whether a problem was a fault or a change request made bug fixing very slow and was felt to be very unmotivating: *It took many steps to get where you wanted. So we felt that things took a lot*

> *It [the discussion] took so insanely much time and motivation.* — Product owner/Team lead

Changing between vendors was expensive for the client since, despite lots of documentation, knowledge was accumulated in the vendor organizations rather than by the client. Client employees were not, for example, allowed to change any code. The contracts specified activities for knowledge transfer to the next vendor in case of such change and the previous vendor was supposed to support knowledge transfer one year after the contract ended, albeit at a fee.

The third factor we identified was the need for increased development speed. Twelve interviewees discussed challenges with development and delivery speed as a factor motivating backsourcing. As discussed previously, the process with the vendors contained a lot of overhead, and the organization was in a model with four annual main deliveries and smaller updates and fixes, “hotfixes” more often.

There was a strong push towards agile development and to introduce continuous deliveries. While vendors used Scrum during development, the overall development process with separate phases for requirement definition, analysis, development, and testing resembled a waterfall process. While certainly possible to do agile development also within an outsourcing model, the organization considered that it would be easier to accomplish with the development under its own control and with less need for negotiations with vendors.

Finally, the need to improve internal quality was mentioned by 14 respondents.

> *“Suppliers did not have incentives to reduce technical debt, and over time, they built up technical debt”* — Office lead

> In the outsourcing model with development done in projects, the budget was mainly assigned to necessary development activities, with tight time and cost goals. Often, no explicit funds for reducing technical debt were included in the projects. The IT department had a small annual budget for maintenance that was used for minor improvements. Our interviewees reported that there was a need for major investments to remove technical debt. However, technical debt was not measured, nor were there any contractual clauses or processes in place to encourage the vendors to develop high-quality code.*Because of how the contract was designed if a developer from a vendor saw something in the code that they thought should be changed, changing it was only a risk. There was nothing to gain from making improvements because if you try it and it turns out that you introduce a fault, then the contract says that the vendor has to cover the costs and fix it.* — Team lead

Interestingly, our interviewees did not mention quality in use as a significant issue.

## 6 Backsourcing Approach

In this section, we present the overall backsourcing approach used at NAV, answering the second research question. We first give an overall timeline and overview of the process carried out in the organization. We then describe the approach taken using six guiding principles in the subsequent subsections.

### 6.1 Overview

The backsourcing process started with the development of a sourcing strategy in 2016, which classified the systems into different target sourcing models. We describe the sourcing strategy in Section 6.2 below.

As discussed in Section 3, the IT system development and maintenance was organized into different areas, which grouped systems into logical groups based on their business areas, such as retirement benefits, or technical area, such as document preparation, data warehousing, or self-service for citizens.

Based on the approved sourcing models, most areas were tasked with backsourcing the relevant systems. The areas were given broad leeway in how to conduct backsourcing, but we could identify a common set of principles they employed, discussed below. While there was no requirement for a defined process for each area, one area developed a model that some other areas adopted it to their own needs as this model identifies several essential activities related to backsourcing.

Figure 3 shows the overall timeline for the backsourcing effort, illustrating the timespan for the various areas. Each bar in the figure represents the overall length of the backsourcing effort. The bars with diagonal lines show the time period used for knowledge transfer between the vendor and NAV. Backsourcing started in practice in the beginning of 2017 when the new organizational model was in place and was mainly finished by the end of 2019, except for one application area with extended contract until beginning of 2021.

We defined the starting point for backsourcing within an area as the point in time when the decision to backsource was made and communicated internally to the organizational unit managing the area, and the buildup of the receiving organization started. We considered the backsourcing effort finished when ownership of all the systems within an area was formally transferred to NAV. All backsourcing activities later described took place during this time period. The white diamonds indicate the original termination time for the related contract(s), and the black diamonds show the use of contractual extension options. These are further discussed in Section 6.6

As seen in Fig. 3, the backsourcing efforts in the various areas lasted from six months to over two years. There are several reasons behind some areas taking longer to backsource fully, including larger numbers of systems, higher system complexity, less knowledge of the solutions, and difficulty in recruiting and building internal teams to take over.

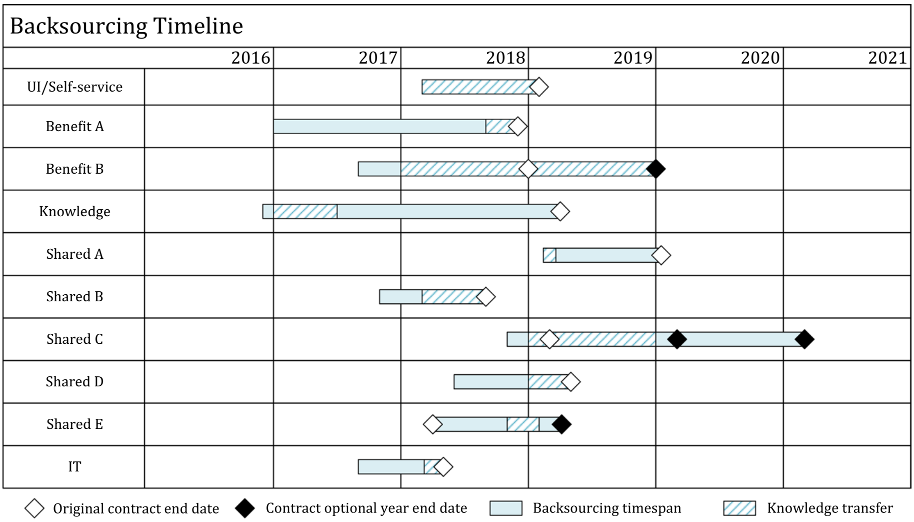



**Fig. 3** Backsourcing Timeline showing the system areas involved

In this section, we describe the overall approach taken, organized as six guiding principles that drove the organization's backsourcing effort. The principles are summarized in Table 5.

## 6.2 Principle 1: Create and Communicate a Clear Backsourcing Strategy

The first principle we identified was to develop and communicate a clear strategy for sourcing within the organization.

The sourcing strategy in this case was initially developed by a working group in the IT department under the guidance of a newly appointed IT Director. The strategy was presented to the top management team in August 2016 and approved in January 2017. The sourcing strategy was developed to provide the organization with better knowledge about the systems, facilitate faster decision-making and better control of development.

**Table 5** Backsourcing principles

| Principle | Description | Motivation |
|---|---|---|
| 1. Create and Communicate a Clear Backsourcing Strategy | Create a well-anchored strategy describing what to backsource and why | Make sure the reasoning and need to backsource is well understood in the organization; align the different parts of the organization |
| 2. Give Autonomy to Organizational Units to Implement Backsourcing | Set goals for the effort and communicate them clearly. Give autonomy to the units to implement the effort according to their needs | Avoid planning paralysis. Show trust and get buy-in. Recognize the differences between different organizational areas and systems |
| 3. Build a Competent Internal Organization | Recruit from the market. Retrain existing personnel. Conduct knowledge transfer in collaboration with the vendors. Integrate IT and business people in the same teams | Build teams able to do end-to-end development. Ensure that business takes ownership of the backsourced systems |
| 4. Provide Organizational Support for Key Processes | Support key processes, e.g., recruiting and contract management. Support employees in building new competencies and changing roles or teams. Support adopting an agile culture and process | Centralize processes that require specialized knowledge. Relieve unit and team leads. Reduce organizational tensions by using centralized processes |
| 5. Treat Vendors as Partners | Avoid terminating contracts. Use contract extensions as needed. Avoid direct recruiting of vendor personnel | Avoid alienating vendors to secure their support during and after backsourcing. Keep good relations for future collaboration |
| 6. Take Over Incrementally | Take over one or a few system/subsystem at a time rather than everything at once | Avoid risks related to big-bang changes. Facilitate gradual knowledge build-up |

A central component of the new strategy was a sourcing framework, depicted in Fig. 4, which classified sourcing into one of five distinct models. In this framework, the level of responsibility given outside the organization, i.e., the degree of outsourcing, varies from fully insourced (model 1), in which the organization internally develops and maintains a system or service and uses vendor resources only for capacity reasons, to business as a service (model 5), in which a complete business process is outsourced, and the organization is not involved at all in providing the service. The organization retained the responsibility for ownership and strategic management in all models. In models 1-3, the organization also held technical management and architecture responsibility. However, in models 4 and 5, the technical solutions were left to the vendors, although strict criteria must be adhered to, e.g., related to storage, security, and infrastructure. The strategy explicitly excluded the operations of the systems from its scope, meaning that for models 1-3, outsourcing operations, which at the time were in-house, was kept as an option.

When preparing the future sourcing model for the systems, five factors were considered: the uniqueness of the organizational needs served by the system, the degree to which the system solved a core task for the organization, the criticality of the system, system complexity, and the use of strategically preferred technologies. The more salient these factors were, the stronger the case for backsourcing the system.

Figure 5 depicts the state of sourcing in 2016 (Original), the recommendations made when the strategy was developed (Planned), as well as the outcome of the backsourcing effort (Realized).

While the analysis details are out of this paper's scope, the overall logic embodied in the strategy was to develop and maintain internally (model 1) core systems for which no commercially viable alternatives existed and which used modern technologies. Core systems that relied on old technology stacks or were to be phased out, were recommended to remain in or move to model 3, i.e., remain developed and maintained by vendors. Other systems were classified on a case-by-case basis, with the guiding principle that off-the-shelf components and software-as-a-service would be used as much as possible.

As our focus is on backsourcing, we discuss transitioning from model 3, "Outsourced development and maintenance," to model 1, "Fully insourced". However, we touch upon

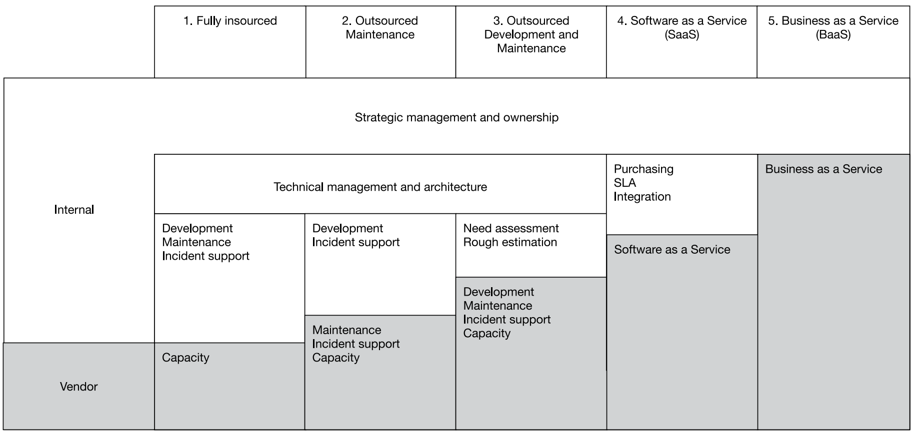



**Fig. 4** Sourcing framework developed in 2016

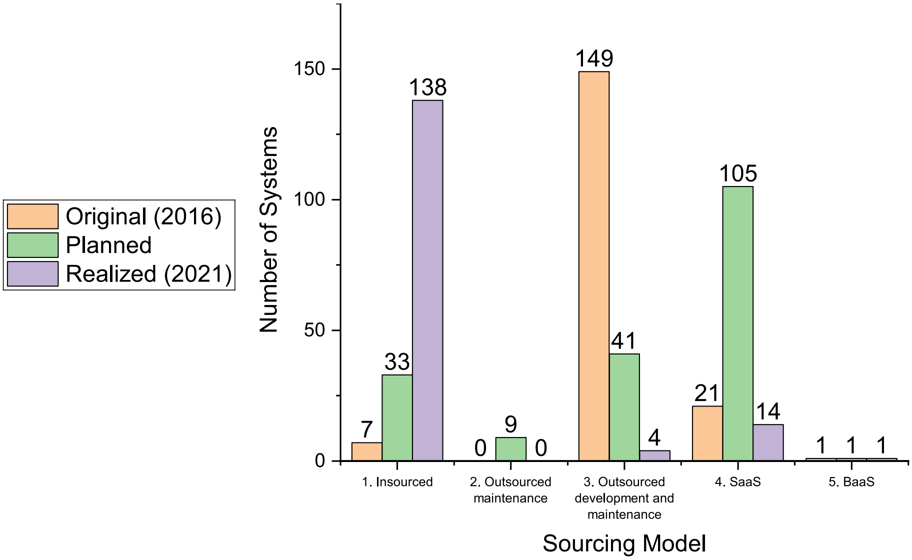



**Fig. 5** Original, planned and realized sourcing models

the overall picture again in Section 8.1 where we discuss the overall sourcing situation at the end of our study.

Once the IT department had developed the strategy, it was formally approved by top management and widely communicated in the organization. This was considered necessary as a means of securing support for implementation and to help reduce organizational resistance to change. However, it is worth noting that the overall backsourcing effort was carried out by the IT department without significant input from the business units.

### 6.3 Principle 2: Give Autonomy to Organizational Units to Implement the Backsourcing Strategy

The second principle was to give broad mandates to the different application areas to implement the sourcing strategy. Thus, on the whole, the implementation of the backsourcing effort was neither centrally managed nor strictly controlled at the organizational level. The main motivations behind this was to avoid "analysis paralysis" and get started with the effort as quickly as possible. Further, there was a sense that it was too many unknowns to facilitate detailed planning beforehand, and that there were significant differences between the areas that would anyhow require differences in approach. Giving autonomy to the areas was also considered a way of getting stronger buy-in for the effort among employees.

Thus, the different application areas, depicted in Fig. 3, were given autonomy to decide when and how to proceed with backsourcing. The existing contracts with the vendors had a typical timeline of 4+1+1 years, meaning that the primary contract period was four years, with the option to extend it twice by one more year. As depicted in the figure, in most cases, the areas took charge of the development either before or by the end of the main contract period. In some cases, the teams used the optional year as well, when more time was needed to ramp up the organization and develop the necessary competency to fully take responsibility for the systems. No contracts were terminated before their expiration date.

The applications were managed by different units (sub-departments and offices) in the IT department, and some coordination was done at that leveOur findings reinforce earlier research identifying high development costs and lack of control as primary reasons for backsourcing (Kotlarsky and Bognar 2012; Moe et al. 2014; Molléri et al. 2023). Consistent with (Nujen et al. 2018), our respondents also emphasized knowledge asymmetry with vendors as a major concern. However, in contrast to cases reported in other domains (Young and Macinati 2012; Falasco et al. 2024), improving quality was not regarded as a primary motivator in our study, although some respondents reported challenges with internal quality as a result of backsourcing.

Other factors mentioned as reasons for backsourcing in the literature include issues with vendor competencies, new regulations making backsourcing difficult or impossible, vendor lock-in with outdated technology, as well as changes in leadership or strategy. In this case study, none of these factors were drivers of the backsourcing decision. The organization got a new IT Director tasked with backsourcing and became the driving force behind it, but this was not a driver for the decision, but rather a result of the organization considering the development costs to be too high. l, mostly from the point of view of sharing experiences. As explained by one IT director:

> *Each unit [sub-department or office] had its own maintenance contracts to work with and then we had some coordination across the units, we had some discussions with the management team, but as a general rule, the whole process related to backsourcing was defined at the unit level* — IT Director

On the positive side, the areas felt empowered to make decisions and run the implementation based on their needs, which differed significantly since there was a wide variance in the type of systems they managed, the degree of technical knowledge they already had and the resulting need for knowledge transfer.

On the negative side, the units that were pioneers in this effort considered the lack of more guidance from higher-level management challenging.

> *I would like to have a little more assistance from the management then. Could feel that I have had a support device in my back. I feel that I have been very alone in the whole implementation.* — Team lead

> As a reaction to the lack of guidance from top-level management, one department defined a "process flow" depicting, in their view, the most salient activities and roles involved in backsourcing. This model was shown to us by representatives of several areas, but it was felt to be unnecessarily complicated and, therefore, simplified.*Yes, [one manager] made this master plan with decision points and dependencies, and we noticed that it it did not work too well, because it forced you to every time assess in what order to do what. So, in that last one that I did, we removed the decision points and dependencies and it was more like "these are the four, five main tracks [activity areas] that we need.* — Manager

The simplified model divided the necessary activities into four flows: contracting, competencies and capacity, teams, and change readiness. While used as inspiration in some areas, in the end, it was not widely adopted.

### 6.4 Principle 3: Build a Competent Internal Organization

The third principle relates to building a competent internal organization to handle the development and maintenance of the IT systems. The main idea is for business to continue to own the backsourced systems but collaborate closely with a newly built development organization using teams able to do end-to-end development for the various systems. This led to the idea of building cross-functional teams with autonomy in managing the handover process and related activities, such as selecting their own development process, tools and managing competence development.

However, as the organization lacked own development competence, achieving this in practice required recruiting new talent and retraining existing people before establishing the teams. Finally, to get the organization up and running, conducting knowledge transfer activities from the vendors to the new internal organization was critical.

We describe this overarching principle through four strategies: recruiting externally, retaining through retraining, building mixed teams combining vendor and internal resources, and conducting collaborative knowledge transfer.

**Recruit Externally** Recruiting was done from the open market. This was an important decision, which, e.g., meant that vendor employees were not directly targeted as potential employees but had to apply in the same manner as everyone else.

The market for people with IT expertise is very competitive in Norway. According to our interviewees, NAV ran recruiting campaigns focused on the value provided to society, which proved to have a positive effect on recruiting. Also, it was essential to offer acceptable salaries to IT workers, which were higher than the salaries otherwise in the organization. This way, the IT department managed to attract some experienced and well-known seniors from the private market, which made the organization even more attractive to others.

A significant lesson learned was that it was easier to hire for specific roles like a "Java developer for the pension benefit system" rather than just a "senior developer," as stated by one interviewee:

> *We have recruited from the market We started to look for people typically like "senior developers", with general announcements. And it worked interview the beginning, but then it started to wane off then we began to go [recruit to] specific teams, and to sell ideas for what they would do. Like "Do you want to work in the daily allowance [one benefit] team?"* — Middle manager

Overall, the results of this push were very positive, as illustrated in Fig. 6. The figure shows the number of employees in various roles in the IT organization during 2017-2021. During this period, the number of developers grew from 90 to 309.

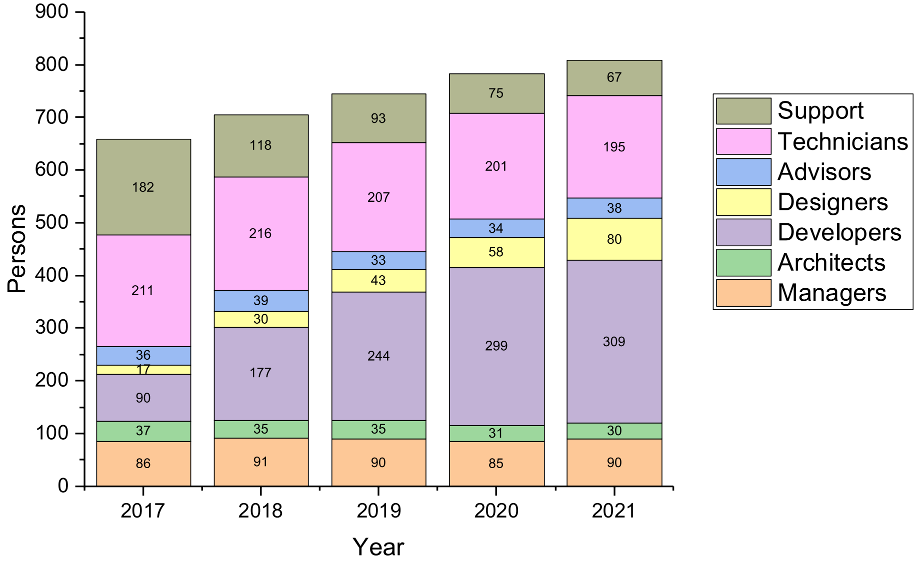



**Fig. 6** Organizational buildup 2017-2021

Although there never was any direct attempts at attracting vendor resources, some people with in-depth knowledge of the legacy systems applied to become employees and continue working on the same systems. These included senior developers and people with mainframe knowledge who were much appreciated by team leads and team members. However, our interviewees stressed that this was not a targeted effort, but rather a side effect of the general recruiting campaign.

**Retaining through Retraining** During the backsourcing effort, many people in the organization changed roles, both on the business side and in the IT department. Many domain experts on the business side became product owners in the cross-functional teams while employees in the IT department changed roles based on their background and interests; for example, from functional advisors to product owners or agile coaches, from project managers and office leads to team leads; and from technical advisors to tool experts, data experts, and developers.

The policy was described by one line manager as:

> *Those who make a transition must be trained in the unit to which they move*
>
> — Line manager

The employee turnover rate stayed low, between 2 and 5%, showing that most people continued their careers in the new organizational model.

While mostly successful, the extent of focus on developing competency varied between units, especially when it came to people who needed to change their competencies, and those who lost their roles during the backsourcing process.

> *There was suddenly no use for many people, who went there for a period and were frustrated and discouraged and It was tough. It was tough for the teams and for everyone. Some were moved to other teams that hadn't progressed as far as us, and it really just moved the problem around. So it was half a year, which was quite tough.*
>
> — Team lead

Regarding people leaving the organization, the interviewees particularly mentioned the loss of people in the old role of functional advisors as a negative point, as they thought the organization lost a lot of domain knowledge as a result.

**Build Mixed Teams Combining Internal and Vendor Resources** During the transition phase, teams were formed, in which internal and external resources were mixed for competency building. The teams also included product owners, taking steps towards building full cross-functional teams. The idea was to build team that were able to quickly develop the necessary competences to take over the systems and to do end-to-end development.

> *At the same time as we start phasing out suppliers, we are building teams to bundle ownership of the products and the expertise around it.* — Middle manager

In 2017, the IT department was structured in a matrix organization with a hierarchical and stable line organization housing "people" and organizing them in competency pools, and a flexible and dynamic network organization housing "teams" responsible for products and services. All employees were associated with a professional line function, and most were assigned to teams for delivery as well. A decision was made to run fewer projects as well, emphasizing continuous development of products.

Although the number and responsibilities of lower-level organizational units and teams have undergone many changes, the matrix organization model is still in use. We describe this organization in more detail in Mohagheghi and Lassenius (2021b)

**Conduct Collaborative Knowledge-transfer** Knowledge transfer was planned jointly with the vendors and included workshops, training sessions, and other activities for knowledge transfer, including working together. The knowledge transfer activities were monitored on a weekly or monthly basis. From the vendors' point of view, knowledge transfer activities were normal, chargeable hours, helping to reduce any potential friction further.

The teams we interviewed had different levels of knowledge of their systems. In four of the application areas, the interviewees thought they had enough knowledge of the systems and two even mentioned that they had better knowledge than the vendors about the systems. They lacked programming skills, and a few consultants were hired through the normal process or continued as vendor resources in the teams after the handover. Therefore, there was no need for extensive knowledge transfer activities.

There was a strong need for knowledge transfer in four areas, even though they felt they had good domain knowledge. In the outsourcing model, technical knowledge was accumulated by vendor resources and transferred between the companies. The same contractual mechanism supporting this was employed by the client during backsourcing, as explained in Section 6.5. The knowledge transfer period varied from 2–6 months, although some respondents claimed it takes up to two years to become fully productive.

> *We have a large portfolio. We prioritized the most important applications, and experienced that the internals also thought it was at a good enough level* — Team lead

Regarding explicit knowledge, system documentation was reviewed and improved, and vendors made analyzes of remaining faults and technical debt. Six of the areas arranged knowledge transfer workshops and training sessions.

Transferring implicit knowledge requires social interaction and solving problems together, which was done by mixing internal and vendor resources and by doing pair programming.

> *[In our team] Knowledge transfer has been: "You and you sit together and work."* — Team lead

In general, teams expressed satisfaction with knowledge transfer and considered that the vendors were very helpful in planning and execution og knowledge transfer activities. System owners on the business side were not involved in the knowledge transfer activities, and the product owners we interviewed commented that the sessions were technical and that they did not have the time or find it interesting to participate. Subsequently, it was recognized that product owners needed training tailored to their role. The organization started the "PO-Academy" for product owners, which has supported the 140 people currently in this role.

### 6.5 Principle 4: Provide Organizational Support for Key Processes

The fourth principle was to provide organizational-level support for key processes related to the backsourcing effort. The four most critical areas that received support were recruiting, contractual support, competence development, and adoption of agile development and an agile culture. Providing central guidelines and supporting leaders during transition reduced the burden on team leads and the affected people.

**Recruiting** The organization had very few developers when the backsourcing process started, so recruiting received much attention. Due to the central role of recruiting to the success of the backsourcing strategy, the IT Director at the time took it as his mission to alter the public image of the organization in the IT market:

> *Who wanted to work for NAV? NAV did not have a good reputation when I started – it was actually quite the opposite because we were known for very rigid processes and a bad atmosphere [as a result of previous project failures]. I spent a lot of time on*

> *recruiting and selling the message. And it was a challenge to strengthen the reputation of NAV we tried plain and simple to get the message out that we are becoming something completely different We are becoming a much cooler employer; we will work with cool technologies, and we want to be a great place to learn for the most skilled [software] craftspersons in the country.* — IT Director

To this end, the organization worked hard to be visible in the media and developer communities through advertisements and media stunts and using social media. It actively started to use Norwegian conferences and other annual events for developers and IT students. Furthermore, the organization was very active in offering intern positions to students to raise awareness among the generation of up-and-coming developers.

In addition to raising awareness about NAV as a potential employer for experts in the IT field, the organizational level also handled the practical recruiting process. The competence needs, i.e., what roles and skills to recruit for came from the application areas, and they were also involved in the hiring process, e.g., in interviewing the candidates.

**Contractual Support** The second area which was handled centrally was contract management. The contract management team in the IT department was responsible for the old contracts and developing and implementing a new contract model for use after backsourcing. During backsourcing, the contract team supported the areas in using possibilities in the old contracts to support backsourcing. Several tactics were applied:

- The old contract model included activities for transferring systems to a new vendor. The organization negotiated with the vendors to do the same activities for transferring the systems to the client.
- The organization relieved economic sanctions defined in the contracts in case of critical faults, so employees got access to code before the formal termination of the contracts. This was necessary to be able to build knowledge during the handover phase.
- Since vendors spent less time on development and maintenance during the gradual transfer of ownership, they were encouraged to use time on knowledge transfer activities instead. Not all vendors were willing to do so since they could be paid time-based for transfer activities for one year.

One vendor discusses this:

> *Because we had a very formal client-vendor-oriented contract, we had to rewrite it partly to fit the transfer, especially regarding the sharing of risk. When we accepted client resources to work with us in development, we had to change the responsibility for faults. Therefore, the economic sanctions were taken out of the contract* — Vendor representative

In addition, a new capacity contract was developed, gradually replacing the target-price development and fixed-price maintenance contracts. The new model identified different knowledge domains, such as user interaction or case management. The organization signed contracts with two or three vendors for each domain to provide resources based on mini-competitions for specific needs.

**Supporting Competence and Culture Development** Organizational development during backsourcing required establishing new teams and changing roles, and acquiring new competencies for many employees. Examples of organizational-level activities to support this were establishing a "Product owner forum", weekly open technical demos, and encouraging employees to take responsibility for their competence development and develop other forums and channels as needed.

It was also essential to have support for onboarding new team members to increase the sense of belonging and avoid disruptions and loss of productivity.

**Support Adopting an Agile Culture and Process** While not strictly part of backsourcing but an important facilitator, the organization adopted an agile culture and process. There were several reasons for this, both internal and external.

As mentioned, the organization had a crucial need to improve its image as an employer to facilitate the recruiting of skilled developers. Adopting agile development was seen as one crucial piece in this.

Further, agile development was thought to ensure working collaboration between the new development teams and the business departments owning the systems, particularly by involving people from the business departments as product owners in cross-functional teams.

A third factor supporting adopting agile development was the close teamwork with people from business departments, vendors, and newly recruited developers would facilitate knowledge transfer through learning by doing.

Finally, adopting agile was seen as a natural way to help increase the delivery speed through continuous integration and deployment.

While the path to implementing agile development in the organization is interesting and multi-faceted; we do not describe it in detail here but refer the interested reader to our earlier papers (Mohagheghi et al. 2020; Mohagheghi and Lassenius 2021b).

### 6.6 Principle 5: Treat Vendors as Partners

Maintaining good relationships with the vendors was key to the success of the backsourcing effort. During the whole process, the vendors from whom the work was being transferred were treated as valuable partners. In many cases, the organization continued to work with the same partners afterward, albeit using the new capacity contract type and a completely new way of working, as discussed in more detail later.

Whereas the areas were given much freedom in how to conduct the practical backsourcing activities, vendor relationship management was mainly handled by the central contract management office and involved actions taken to communicate the decision about backsourcing to the vendors, management, and development of the contracts, and in some cases, changes to the contracts during the process of backsourcing, as described in Section 6.5.

In practice, the implementation of this principle could be seen in a set of strategies used in working with the vendors. We describe them next.

**Do not Terminate Contracts** Instead of terminating contracts early, NAV started the backsourcing effort in various areas well before planned contract expiration to, as far as possible, complete the backsourcing within an area by the end of the natural contract expiration date. In practice, no contracts were terminated before the end of the main contract period.

As mentioned in Section 6.1, the existing contracts included provisions for up to two so-called "option years", allowing NAV to continue the contract beyond its expiration. As shown in Fig. 3, the possibility to extend the contract was used in three areas; in one case consisting of many systems, both optional years were required to build the organization and competencies needed for the takeover.

**Inform the Vendors Early** A natural result of the previous strategy was informing the vendors early about the backsourcing decision. Vendors were aware of the internal discussions in the organization that started after hiring the new IT director. The consultants were working within or tightly with the existing teams and, in some cases, were even team leads. They also had access to internal communication channels such as Confluence and the IT blog, where discussions about planning and realization of the new organization were going on. Our interviewees on the vendor side reported helping the organization during the process of becoming agile and building knowledge.

> *We were informed. When the new IT director came in, we had a good dialogue with him about the [sourcing] strategy, and was also involved in the sourcing strategy of NAV. And we immediately saw that the area we had was certainly the most mature. We had come a long way in terms of agile development with NAV developers, and it was probably the most natural area to start with.* — Vendor representative

> *Then we had a longer period where we tried, also within the old management agreement to see what adaptations we can make in the way we work, the way we cooperate within the existing contract, which takes us in the direction we want.* — Vendor representative

**Collaborate During Backsourcing** During the backsourcing process, the role of the vendor teams changed, and they were finally disbanded and replaced with internal teams. However, this change was done gradually. As explained later, rather than creating new internal teams "competing" with the vendor teams, NAV employees were put in the vendor teams, creating a mix of resources and providing a natural place for onboarding recruits and conducting knowledge transfer. The vendor and experienced internal personnel worked together to help the new people get up to speed, as described in more detail below in the section on knowledge transfer.

Also, our interviewees frequently mentioned good collaboration with the vendors during the process of backsourcing. Some interviewees mentioned that the vendors wanted to continue the good relationship with the organization to be suppliers in the future using the new contract model.

> *They were very, very accommodating. So they were very understanding that we were going into a new model.* — Team lead

> *The supplier tried everything they could to make themselves look good, so that we would continue to use them as a consulting supplier.* — Middle manager

Our interviewees mentioned two main activities that the vendors and internal personnel conducted during the backsourcing process: knowledge transfer and system status walkthrough. We discuss knowledge transfer in Section 6.4, and system walkthrough in Section 6.7.

**Retain Good Relationships with the Vendors after Backsourcing** Despite the intention to take over the development and maintenance of the applications internally by its own personnel, NAV actively worked on retaining close and working relationships with the vendors.

In practice, in parallel to phasing out the old contract model, a new, so-called capacity contracting model was phased in, as mentioned in Section 6.5. In this model, the organization started hiring consultants from the same vendors but on an individual rather than a team basis. Also, the focus was on hiring people for technical roles and keeping management and administration internally.

In practice, this change to a new contracting model resulted in many consultants from various vendors continuing to work with the same systems they had in the old model, but now as equal team members in the newly formed internal teams. This helped retain and transfer knowledge to the new teams in the long run.

Another aspect of retaining good relationships with the vendors was to avoid direct recruiting of vendor personnel. If vendor personnel wanted to be hired directly by NAV, they had to apply for the open positions like everyone else. Several individuals from vendors applied and got hired by NAV, but this was part of the normal recruitment process, not any attempt to grab resources from the vendors.

### 6.7 Principle 6: Take Over Incrementally

The sixth principle we identified was to take an incremental approach to transfer the ownership of the systems within an area. The idea was to avoid big-bang transfers in favor of a gradual approach. In areas with many smaller systems, this meant transferring the ownership of one or a few systems at a time, and in areas with large systems, one system at a time.

As previously discussed, knowledge was transferred to the new teams for each system and employees started working with the systems while being supported by the vendors. Often, when their knowledge had reached a sufficient level and a team was established to take over; a formal handover was done, after which the full responsibility for the system was moved from the vendor to the new team.

As part of the knowledge transfer, the vendors and the teams did a walk-through of the systems, going through the documentation, the remaining faults, and any technical debt or other issues remaining in the systems. Since, in many cases, some of the vendor's people continued to work in the new teams as team members under the new capacity contract model, the handovers went largely without problems.

## 7 Backsourcing Challenges

This section discusses the organization's challenges during the backsourcing process, answering our third research question. We discuss the challenges under three main categories: change management, competencies and resourcing, and organizational tensions. Table 6 summarizes the challenges, and the next subsections discuss them in more detail.

### 7.1 Change Management

The first challenge was understanding and communicating the extensive scope of the backsourcing effort. While heavily driven by the IT organization, the fact that backsourcing is a process that affects large parts, if not all, of the organization was not well understood in the beginning. In the words of a manager:

> *I used [a model of backsourcing activities] a lot because [we] believed that [back] sourcing was only about taking over things. But it affected so many things. It included recruiting, how we work, collaboration between IT and business, and change in how we achieve our goals.* — IT Manager

From a change management point of view, the strong lead taken by the IT department led to a situation in which other parts of the organization, particularly the business units that owned the systems were involved too late and too little. A Product Owner explained:*Yes, we were involved, but I really wish we were more involved But, because this initiative was taken by [the IT department], and they drove the process, they naturally focused first and foremost on themselves. So, for example, the compe-*

**Table 6** Main challenges during the transformation

| Category | Challenge |
|---|---|
| Change management | Poor understanding and communication of scope |
| | Business units engaged too little and too late |
| | Lack of management support |
| | Personnel experienced pressure and felt insecure |
| | Difficulty adapting the financing model to support agile development |
| Competences and Resourcing | Difficulty in recruiting |
| | Retention of own and vendor personnel |
| | Poor handling of retraining and reskilling |
| | Difficulties in defining and developing competencies for the product owner role |
| | Difficulty in allocating new people to legacy systems |
| Organizational tensions | Heavily IT-led transformation |
| | IT overstepped boundaries to business units |
| | IT grew, business units did not |
| | Unwillingness of business units to provide product owners to teams |
| | Lack of communication between IT and business units |

> *tence development for product owners should have been better, and we [on the business side] should probably have taken more responsibility...So we were invited into something that somebody else owned and knew more about so we were perhaps a little bit skeptical, and, well, a little more anxious about making this change, and afraid that we would lose delivery momentum We have a strong focus on delivery, as we often are the [system or service] owners.* — Product Owner

Many respondents stated that stronger management support would have been needed and that they sometimes felt "alone" when working with the backsourcing effort. Things were further complicated by periods of lack of trust at the top levels of management.

Overall, communication challenges were abundant, and many people reported feeling uneasiness during backsourcing, as little information was available, e.g., about what would happen to people whose roles became redundant due to the backsourcing and reorganization done in conjunction with building up the new organization. In the words of a team lead from one vendor organization:

> *it was both amongst our [the vendor] people who were to enter new roles in the new organization, or even the same role, but what would it mean to be a developer in the new organization compared to what we are used to? There was a lot of uncertainty even about whom to ask if you wonder about something, or even if you should or should not ask. Many things were unclear also to the NAV [host organization] employees about what they were to do now. Many had experiences and roles that suddenly disappeared in the new organization This is what really strikes me there was a lot of uncertainty, and it was it was not considered. It took a long while before things became clear.* — Vendor representative

Many people also reported feeling pressure to do things before they thought they were ready or even that they thought did not make sense. Examples of this included creating cross-functional teams for legacy systems, acting as product owners, and taking over the responsibility for a system before the necessary resources were in place.

Another major challenge was that employees felt that too many changes happened simultaneously. They had to learn systems, learn new ways of working, and build relationships in their new roles and organizational units.

The financing model, i.e., how the teams were allocated money to do their work was frustrating during the backsourcing process:

> *The financing model is very unclear to us after introducing the new [organization]. We don't understand it and don't even know if we have a financing model. So, no, we get no more financing now. We had models for it until two, three years ago, [defining] how the financing was allocated* — PO coordinator

At the beginning of the process, many teams were formed into so-called "IT teams," with a fixed budget for maintenance and minor changes and where the business side had to pay them on a requirement basis for major changes estimated only roughly. This way, teams could eliminate the heavy estimation process and prioritize part of

> their backlog themselves, although not giving them full autonomy:*Up- and downscaling all the time...we don't have a good source of financing for maintenance and IT teams...I need to fire and hire people as single orders come in. It has been completely hopeless...Financing for IT teams is scandalous. It just does not work.* — Vendor representative

Fully financed platform teams and teams with many orders with project financing were able to work in a stable environment. Other teams mentioned challenges in getting funding for reducing technical debt, training new people, and keeping knowledgeable external resources on the team.

Some respondents also let us know that it was unclear to them even how money was divided between the business units and the IT department:

> *But we have not come further than that the money is allocated through the management hierarchy, and it means, e.g., that [a product owner] on the business side does not have any money. But she can work within the limits given by her manager. But it is not like she could use money to prioritize and there are some major discussions going on about what money the business should get and what should go to the IT department. And now, IT has more money than the business side.* — Manager

Later, real cross-functional teams were formed, and they were given full funding for the whole year, making it unnecessary to "sell" their services to the business side, as the cross-functional teams included personnel from the business side as well. This was a clear step in the right direction.

At the end of our data collection, some teams were organized into larger, fully financed product areas, as we described in Mohagheghi and Lassenius (2021b). However, the hierarchical financing model still applies; i.e., business areas are funded, and they must break down their budget and assign it to product areas, while some funding comes from large projects as before.

### 7.2 Competences and Resourcing

Related to competencies and resourcing, the organization faced challenges with retaining knowledgeable personnel, recruiting, retraining and reskilling existing people, introducing the product owner role, and allocating people to work on legacy systems.

In many cases, the plan for how to deal with people whose roles were being phased out was not clear or well communicated. This created challenges with employee motivation and a sense of unsafety, leading to challenges with retention and retraining people to work in new roles. As a result, many people with strong competencies in both the technical aspects of the systems as well as of the functionalities from an IT perspective, e.g., functional advisors, left the organization.

> *It is pretty tough. So many have quit NAV of those people. So many competencies have left.* — Team lead

> *In the rest of the house, it is like: "These must out." And so a lot of technical advisors and functional advisors have quit, which costs a lot and will cost a lot in the future, in development time [to develop similar competences] for others.* — Agile coach

> In addition to challenges with retaining their own competent people, another was to retain critical vendor resources with unique skills needed during and after the transformation. In particular, some people worried that this transformation from relying on knowledgeable consultants to building their own competencies was not understood well enough. When replacing vendor personnel with new people, taking into account the time for knowledge transfer was challenging. In the words of one team lead: *as I told the vendor, I've said that we could have become a superb team that would lift the whole organization [to a new level] if we had been allowed to keep the people from the vendor that we would have liked to. But it boils down to being about the realization of profits [cost cutting]. When you exit [do not renew] the contracts, the cost will go down, but one thing you don't see is that it needs to be a gradual change [from relying on vendor competence to having own competent resources].* — Team lead

> Despite the active attempts at raising the organization's public profile as an attractive employer for people skilled in IT, recruiting enough developers, data scientists, etc., was a challenge, slowing down some backsourcing efforts. The quick remedy, which was planned for and supported through the capacity contracts, was to continue to involve vendor personnel in the teams:*[It was difficult] to get Java developers. And it was a really big challenge. And, it is clear that as long as we could not recruit competent people, it was impossible to do any knowledge transfer because there was no one to transfer to. So, therefore we really needed to continue hiring externals [consultants] using the capacity contracts. It was maybe a little more expensive solution.* — Team lead

Due to the fact that the backsourcing effort was heavily led by the IT department, the business organizations who were tasked with providing product owners to the new teams were not fully committed. This created many challenges in getting the product owner role to work.

Several team leads mentioned the challenge of having access to product owners who could spend significant time in the new teams since business departments were not ready for the product owner role. In some cases, the product owner was simply a named contact person in the business department who helped to put the development team in contact with the real business side customers rather than being a team member. In some cases, the business department simply refused to provide full-time product owners to the teams.

> *We received operating resources. We put the team together as if it were an interdisciplinary product team, but we had problems getting the business side on board, so we were missing the product owner. But otherwise, we had members from the departments we had previously collaborated with.* — Middle manager

One reason for this refusal was, as discussed in the next subsection, the fact that while the IT department was allowed to hire new people, the business departments were not but had to identify product owners amongst their existing personnel, which seems to have created some resistance.

In addition to challenges with getting the business departments to provide product owners, there were challenges in getting the right competencies. In the words of one vendor representative:

> *to be completely honest, the role of product owner has been undervalued. What that role entails has not been understood, and it is a bit random who becomes a product owner. In many cases, the person becoming a product owner is selected due to domain competencies and experience with the system that needs to be implemented or maintained, but not necessarily the competencies a product owner needs.* — Vendor representative

### 7.3 Organizational Tensions

The fact that the backsourcing effort was clearly done under the IT department's leadership, without the business departments' strong involvement, created many organizational tensions. It became a situation in which some people felt that the IT department had the vision, the resources, and the power to decide over others who were at a similar level in the hierarchy. In the words of one employee representative:

> *A challenge at that time was that the IT department progressed very quickly, and the IT director had visionary ideas about where we were going, and they were way ahead of the rest of the departments in the directorate, which presented a big challenge.* — Employee representative

> A high-level manager in the IT department also reflected on this, noting that while building a strong identity for IT was good, there was a risk that they started to look down on the rest of the organization, complicating collaboration: *there was a big challenge that we had this own identity at IT, which was to a high degree different than in the rest of the organization. So I think that [the IT director at the time] was extremely successful in building a strong identity and story around the IT transformation, but it was to a certain extent about defining us in contrast to the rest of the bureaucracy, which was the old, boring gang, who had not really understood anything?* — IT manager

> Naturally, during the backsourcing effort, the IT organization was permitted to hire a lot of new resources to fill new development-related positions. At the same time, the business departments were mainly not allowed to recruit, and several of departments were even downsized. This created a sense of unfairness and jealousy towards the IT organization, regardless of the fact that this probably was the right decision. One employee representative explains: *it is no doubt that people at one stage felt that different departments were treated differently. IT could recruit so many new people, but the other departments experienced a stop in recruiting. They had needs, they too, and it was a bit difficult to motivate this during a period, but as time went on, and*

> *we got more people in cross-functional teams, we started to see that this has been the [right] way to go.* — Employee representative

> However, the opposing opinion was also expressed, i.e., that the business side also should have been strengthened in the transition: *So one weakness of this [backsourcing] was that the money was only used to strengthen the IT organization and not the business side. We could have increased productivity much faster had we strengthened the business side in parallel with IT* — Product coordinator

In particular, a case in which the IT department recruited data scientists and data managers, roles that in the old organization had belonged to business functions, creating friction in the organization, despite the fact that the overall outcome ended up being positive.

There were also fundamental differences in the mindset about how people thought things should work. Simplifying slightly, the business departments wanted more control and rules, in particular around managing product changes. In contrast, IT wanted to implement a fully agile way of working without formal change management procedures but with empowered product owners in the teams, which also was what in the end was implemented.

## 8 Backsourcing Outcomes

In this section, we describe the outcomes of the backsourcing initiative, answering the fourth research question. We first discuss the extent of backsourcing compared to the plan, followed by a discussion of subjective and objective changes to development speed, productivity, quality, and culture and atmosphere. Table 7 shows the main outcomes of the backsourcing effort.

**Table 7** Main outcomes of the backsourcing effort

| Dimension | Outcome |
|---|---|
| Extent of Backsourcing | All planned systems and many others were backsourced |
| Cost and Effort | Overall IT cost did not change significantly. Cost per man-month became lower because of the higher number of internal resources |
| Development Speed | Significant change; from quarterly to continuous deployment |
| Productivity | Subjectively higher, no available hard data. Reuse: mixed results. On one hand internal development facilitated taking use of shared platforms. On the other hand, team autonomy led to freedom of choice and divergence in processes and technologies. |
| Quality | Improved quality-in-use. No hard data on quality in development; reduced technical debt |
| Culture and Atmosphere | Improved product/service ownership, Improved motivation |

## 8.1 Extent of Backsourcing

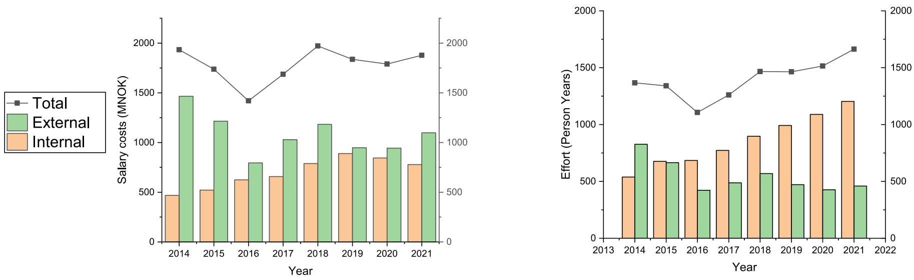
As discussed in Section 6.2, the new sourcing strategy approved in 2016 defined a goal state with respect to how the systems were to be sourced in the future. In Fig. 5, the red series shows the outcome of the backsourcing on the system level, as realized by the end of 2021. We can make several interesting observations based on this.

In 2016, 91% of the analyzed components were outsourced, while the same amount were insourced in 2021, leaving out the components that are not in use anymore. Thus, the main change has been to backsource most systems. The ambition to use the SaaS (model 4) or use it together with internal adoptions (model 4+1) has not been realized as planned.

We asked a high-level IT leader and a chief architect about the reasons for this apparent change of strategy. According to them, a combination of several factors may describe the strong move to internal development compared to the original plans: a) the IT department has succeeded in recruiting developers, even for mainframe applications, which were not seen as possible in 2016; b) it was over-optimism in 2016 regarding availability and ease of use of SaaS that could cover the organizations use and fulfill security concerns, and c) for some components, model 1 is an intermediate stage until new services or off-the-shelf components replace them. Also, going from outsourced development to SaaS requires a significant project, but going from outsourced to backsourced can mostly be handled through a contractual change and was, therefore, faster and easier.

## 8.2 Cost and Effort

Despite the main driver behind the backsourcing effort being to reduce the cost of IT system development, operations, and maintenance, the internal analysis by 2021 showed no cost reduction. In our study, we got high-level cost and effort data at the organization and person-year levels. This data showed that the overall spending on the IT workforce has remained fairly stable, as illustrated in Fig. 7a. The figure shows the amount paid as salaries and fees to internal and external personnel from 2014 to 2021. However, as discussed below, the amount of person-years being done with that amount of money has increased significantly; thus, doing more for the same money.




(a) Salary costs related to digitalization

(b) Effort spent on digitalization

**Fig. 7** Cost and effort related to digitalization 2014-2021

The figure shows that the total IT salary costs have remained almost the same since 2014, with a slight dip in 2016. The backsourcing effort can be seen in the increase in internal costs and decrease in external ones. However, the reduction in cost for external work is not as dramatic as one could think based on the change that has been done: even in the new state in which most systems have been backsourced, a large share of the cost is still incurred by the use of external consultants. From a cost perspective, the average cost of a person-year was 1.8 MNOK in 2014, and 1.1 MNOK in 2021, a 34% drop, reflecting the smaller portion of external work.

When discussing the effect of backsourcing on cost with our interviewees, most of them raised the issue of the difficulty of making valid calculations due to the many changes in both processes and accounting that had been made as a result of the backsourcing effort. For example, teams do not report hours spent on tasks or categories at all, and many teams have completely stopped estimating and monitoring costs. Thus, there is little data comparable to the situation prior to backsourcing, where vendors reported detailed effort and cost. Attributing salary costs to specific systems and teams also turned out to be futile.

However, the effort spent on digitalization activities, shown in Fig. 7b, gives a more positive outlook: the overall effort has increased from 1400 person-years in 2014 to about 1750 person-years in 2021. Thus, the organization has added 350 yearly person-years of effort without significantly increasing IT spending due to backsourcing. Key to this is the increased internal work share, which increased from 39% in 2014 to 72% in 2021. As a rule of thumb, an external person-year costs 2-3 times as much as an internal one.

While the organization did not have any clear analysis of the monetary effects of backsourcing, our respondents mentioned three factors that they thought have had a positive effect on saving money: less waste in the process, as all administrative work related to managing subcontracting was eliminated; the significantly lower cost of internal vs. external resources, and a sense that it was no longer possible for vendors to suggest and charge for unnecessary work.

They also mentioned factors that they thought help explain the lack of a major reduction in cost: the clear growth in person-years being done at the same cost, investments in new areas and the development of new technology, the strategy to still keep systems that are close to end-of-life outsourced, and the continuing reliance on external consultants as part of the teams.

Overall, all persons we interviewed considered backsourcing worthwhile and successful and were not worried about the financials.

### 8.3 Development Speed

The perhaps most salient change as a result of the backsourcing process can be seen in the change in deployment speed, as measured by the number of and time between system deployments. Most application areas have transitioned from a model with four annual main releases and smaller and more frequent bugfix-“service pack” updates to continuous deployment, in which applications can be deployed directly by developers even several times daily. Figure 8 illustrates the number of deployments per month per application area for the time period March 2016 to March 2022.

In the figure, we can see a significant growth in the number of deployments starting in 2019 and continuing until 2022.

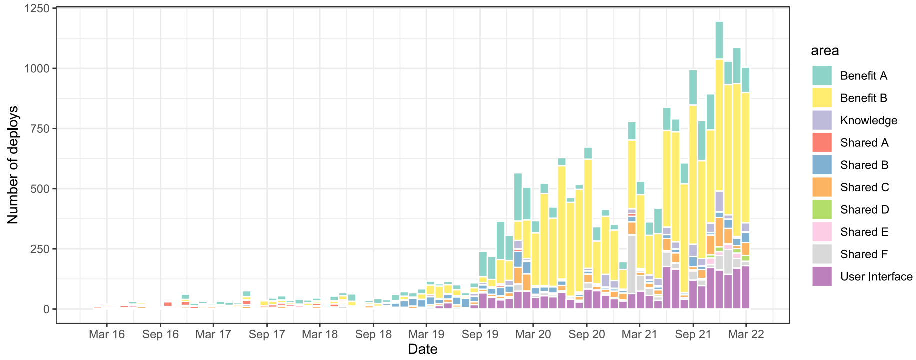



**Fig. 8** Number of deployments for the various application areas for the time span Mar 2016 - Mar 2022

There are several explanations for this change, including the development adoption of a new platform, NAIS, for continuous integration and deployment, significantly reduced waste due to process changes as a result of dropping the old outsourcing model, and full-scale adoption of agile development with autonomous product teams. Another key to the improvement in speed is the empowerment of the teams to make decisions related to their products. Many product teams now have full control of the whole process, from backlog management to deployment, and can make all necessary decisions to deliver a feature or fix a problem as quickly as needed. Unfortunately, as the teams do not systematically collect data on work activities, we were not able to calculate end-to-end flow times, making a quantitative comparison of the change in time-to-market for, e.g., new features infeasible.

## 8.4 Productivity

Regarding productivity, our respondents mostly considered it to be at the same level or higher than before backsourcing. The original premise for the transformation, as touted by the IT Director, was "Double the output for half the price". While we discussed the effect on cost earlier, we were not able to get quantitative data on productivity, as it was not measured before or after backsourcing. This lack of analysis was reported to have created some tension between the IT and economy departments.

Before backsourcing, effort spent by externals was logged and reported by vendors. With the incorporation of internal teams, effort and schedules for work items and features were no longer systematically monitored, and many teams opted not to collect such data.

Thus, for our purposes, we rely on the statements of our respondents. The opinions on the effects of backsourcing on productivity varied widely, with 20 respondents reporting that they think it is higher, eleven considered it lower, and four that it is the same as before. The overall assessments were from "doing more with 10 people now than with 23 before" to very neutral or slightly negative assessments. Many respondents stated that improving productivity was not, in their minds, even that critical a goal of backsourcing. The overall sentiment was that productivity after backsourcing is higher, or at least at the same level as before.

In addition to an overall assessment, we asked our respondents to explain what factors they thought had influenced productivity, see Table 8.

**Table 8** Factors affecting productivity post-backsourcing

| Factor | Effect | Explanation |
|---|---|---|
| No subcontractor interface | Positive | No administration of subcontractors needed. Less time spent on analyzing requirements, estimation and negotiating with the vendor. Less time spent on discussions in case of faults. Double roles and activities removed (e.g., project managers, testing) |
| Direct communication between developers and business | Positive | Fewer misunderstandings, faster response |
| Fully funded agile teams | Positive | Teams do not need to put effort into getting funding and resources for their work |
| Rework | Positive / Negative | Less rework due to direct communication and better-understood requirements. More rework due to lack of up-front analysis and "jumping to development", and less extensive testing |
| Inexperienced new people | Negative | It takes time before new people learn the complex systems and domains, and become productive |
| Fewer effective working hours | Negative | Competence development by internals consumes development time; externals do it outside working hours. A higher ratio of internals to externals decreases effective working time. |
| Technical debt | Negative | Need to work on reducing technical debt |

**Table 9** Factors affecting quality, as reported by our respondents

| Factor | Effect | Comments |
|---|---|---|
| Continuous fault correction | Positive | Faults are corrected immediately: a significantly higher number of errors corrected per time unit than previously |
| Teams directly responsible for quality | Positive | "You code it, you fix it" mentality improved the quality |
| Focus on removing technical debt | Positive | Systems had a high degree of technical debt before, making changes difficult |
| Less systematic manual testing | Negative | No test leads, fewer functional experts, lost testing competencies |
| Less pre-deployment testing | Negative | A higher number of problems make it to production, but are fixed quicker |

We identified three factors that were considered to have a positive effect on productivity, three with a negative impact, and one factor that arguments were made both ways. We discuss each main factor next.

The most salient positive productivity factor was the change in the way of working from a subcontracting process for development done by vendor teams to internal teams. This removal of the subcontracting interface made the process much less bureaucratic and removed many non-value-adding activities. In particular, our respondents reported that not having to discuss whether a problem is a fault or an improvement and not having to prepare orders with detailed requirements, specifications, estimates, and so on saved much time and frustration. A vendor explained (and tried to quantify) this change as it related to small development tasks:

> *in the old contract, we thought that the smallest change...that had to go through the whole process there was nothing less than twenty-some hours. That was the smallest quantum we could get through for the smallest possible change. [Now] when a single person can decide and implement a change from idea to production without involving others, I think it realistically can be done in one, two hours.* — Vendor representative

Indeed, the fact that small improvements and fixes could be done very quickly was mentioned as positive by most respondents. Many reported that the number of fixes shipped had increased by a factor of ten, which corresponds well to the consultant's viewpoint above. Thus, at least for small tasks, the improvement seems to have been clear. When it comes to larger tasks, we were told that they are delivered incrementally and, therefore, disguised as small tasks. However, some respondents also noted a lack of up-front analysis for larger tasks, creating more rework downstream.

Direct communication between developers and business people responsible for the systems was considered an important factor that helped improve productivity through faster decisions and better understanding and iteration of requirements.

Further, the institution of fully funded agile "product" teams was considered to improve productivity. Such teams are funded through a fixed yearly budget and do not have to acquire funding for each piece of development work separately, something the other teams have to do on a requirement-by-requirement or project basis.

Rework was a factor that painted a divided picture. The direct communication lines between business people and developers made requirements better understood during development, reducing the need for rework. However, some respondents were also concerned that the new process was "too flexible and too fast." These concerns were related to, on the one hand, less testing and, on the other hand, jump-starting development without a good enough understanding of the requirements. A few respondents commented that the large number of small issues fixed quickly are, at least partly, due to a lack of sufficient testing. Developers did not consider less testing as a negative factor, and stated that they can fix any faults observed in production fast enough. Jump-starting development was a particular concern for product owners, as sometimes developers started development on a feature without a thorough walkthrough of the requirements with them. Further, there were concerns related to the lack of up-front planning for larger features, speculating that many issues that must be fixed downstream might be due to this.

Regarding factors that lower productivity, the most salient one was adding new, inexperienced people to development, a natural result of backsourcing. Our respondents stated that onboarding people always comes at a cost and that the hires they were able to get, in many cases, were quite junior, reflecting a lack of experience in technology. In addition, the systems and domain were considered highly complex, making onboarding a significant challenge. The transition was made smoother by the fact that some consultants joined as employees and many consultants continued to work in the teams, both due to demand for their skills and difficulties recruiting enough new personnel. One respondent noted that getting new people is not a new phenomenon: developers had also changed in the vendors' development teams, but the vendors had a very efficient process for onboarding new people, something NAV did not, although in practice, one could observe reduced productivity when changing vendor resources, as well. One would expect the effect of onboarding and competence transfer to have a temporary lowering effect on productivity, but unfortunately, we were not able to get any data to confirm or refute this.

Interviewees mentioned that they spend more time in meetings now, both development and non-development-related ones, such as training, communities of practice, and social activities to enhance team culture.

Finally, working on reducing the high level of technical debt was reported to take up significant time for many teams. At the time of our interviews, there was no clear policy for how and when to deal with the technical debt, nor any specific time allocation, making it difficult to assess the significance of this factor. While it was considered important for improving quality, the direct impact on productivity may be negative.

## 8.5 Quality

Measuring possible changes in software quality before and after backsourcing turned out to be challenging due to changes in data collection and poor availability of data.

We intended to analyze both the internal and external quality of the software pre- and post-release before and after backsourcing to see if we could detect any changes that, at least partly, could be attributed to the change in sourcing. This, however, turned out to be futile - perhaps not surprisingly.

Before backsourcing, pre-release bugs were reported and handled systematically, but in the new way of working, bugs internally found are handled directly by the teams and not systematically registered. As a result, we are not able to analyze any potential changes to the amount or types of pre-release defects. Therefore, we focused on external quality in the form of incident reports from the field.

In our interview data, 25 respondents considered the quality to be higher than before, eight respondents reported concerns with lower quality and twelve persons considered that the quality after backsourcing is the same as before. We first discuss the factors affecting quality mentioned by our respondents, followed by a discussion of externally observable changes to the quality.

### 8.5.1 External Quality

In Fig. 9, we can see the number of new incidents reported per area and trimester from June 2017 to May 2022. Unfortunately, we could not get data from the time period before that, as the reporting systems had changed and older data was no longer available.

Looking at the figure, we can see a clear decline in the number of new incidents per area, which would indicate an increase in external system quality. While this overall improvement in quality likely has a relationship to backsourcing, based on this data, we cannot rule out other explanations, such as a general maturation of the systems. However, several of our respondents explicitly linked the new way of working and the improved quality. In the words of one team lead:

> *Yes, I really think that we have higher quality, because we have fewer faults. Because we...maybe we work a bit closer together with the business side, and we are focused on delivering a good solution.* — Team lead
>
> Indeed, one of the most significant changes was related to bug fixing and small improvements. In the old model, bugs were reported, analyzed, and negotiated with the vendor before being scheduled for correction, and in many cases, the discussions about who was responsible for the cost of fixing a defect could be difficult. In the new model, teams immediately correct errors as they are found, often being able to deploy corrections within a few hours after the discovery of a bug. Furthermore, in the spirit of continuous deployment, teams are empowered to push directly to production, whereas previously that step could only be taken by the operations organization. One team lead explains:*Yes, everybody is very happy, since you can take something very quickly to production. An error correction, or a change – it does not take many days from the need has emerged until it is in production.* — Team lead

Prior to backsourcing, the vendors had to adhere to strict SLAs for all systems, with predefined response times depending on the severity of the reported incident. As a result of backsourcing, SLAs are no longer used in the organization, and it is up to the product teams to decide how to handle incidents. The practices vary somewhat, with some teams having

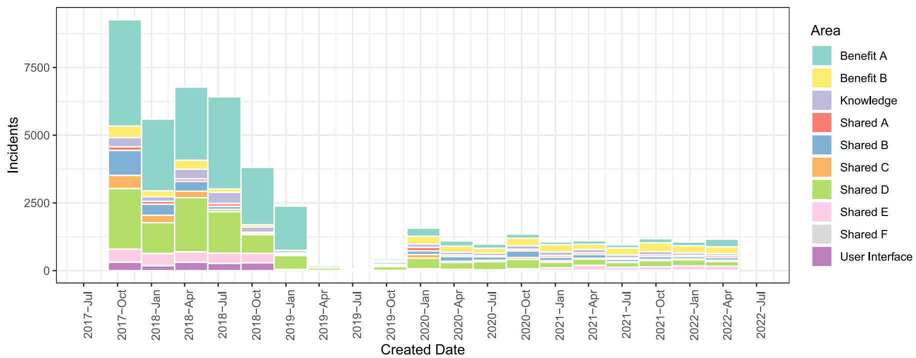



**Fig. 9** Incidents created per trimester per area

people on call, but most teams having no specified rules, and handling incidents ad-hoc as they occur.

#### 8.5.2 Internal Quality

The *internal quality* of many systems was considered poor pre-backsourcing by many of our interviewees, who considered the amount of technical debt a serious problem from the point of view of maintenance and further development. The high level of technical debt is mostly claimed to be due to the outsourcing agreement: consultants had no incentives to reduce technical debt, as it was not a paid-for activity. After building own teams, there has been an increased focus on reducing technical debt.

The teams now work on technical debt continuously. However, there were no clear organization-wide rules for how to do this, e.g., in the form of time allocation specifically for working on technical debt, although some teams reported rules like 30% of time for reducing technical debt, and 70% for developing new features. We were unable to obtain any hard data or analysis of technical debt before and after backsourcing.

#### 8.5.3 Culture and Atmosphere

An aspect difficult to measure but very important is the effect of backsourcing on the organizational culture and atmosphere. In the interviews, most respondents reported that through backsourcing, people in the organization feel an increased sense of motivation and ownership. A leader commented on this as:

> *Building competency and having the responsibility to develop solutions gives quite different motivation than following up the suppliers.* — Middle manager

> The inclusion of product owners from business units as team members has created a change in mindset on the business side, shifting from an ordering mindset to a collaborative approach with a greater focus on end-user needs. In the words of one product owner:*Now we have more ownership of the applications themselves, so we don't have the orderer-supplier role, which means that there are handovers, that you don't really know what you have ordered, and then you have a supplier who may misunderstand the order. Now I actually think that the developers have greater ownership of the applications themselves so that they are more they maybe care a bit more about the application, I have an impression. They feel greater ownership of the portfolio, and this means that we have sparring sessions along the way, which means that we are able to specify what we order, so that we actually get what we should have, instead of something completely different.* — Product owner

> Further, the consultants that now worked in the teams also reported a stronger sense of belonging and identification with the organization. They reported not feeling like outsiders anymore: *Now you are a part of the team. Much more integrated. It makes it [working] much easier, no doubt.* — Vendor representative

Giving teams responsibility for the products they develop and maintain, and autonomy to decide how to work have created high involvement and instilled a sense of pride in the teams.

The organization wishes for a strong development community, and reflects this on its public image as well. This is necessary to be able to recruit people with development-related skills at a high scale. The IT manager who started the process of backsourcing was a front figure for this change and recruited some experts from the private sector that improved the public image even more.

## 9 Lessons Learned

We summarize the lessons learned by the organization in five dimensions, as illustrated in Fig. 10. The lessons emphasize that strategic planning, effective communication, good vendor relations, and good organizational support help smooth the transition from outsourcing to in-house development. The lessons learned are based on what our interviewees explicitly stated, as opposed to our interpretation.

In addition to actual back-sourcing, the organization's adoption of large-scale agile development produced another set of lessons that we do not report here, as they are outside the scope of this paper.

1. **Vendor Relationship Management**
   – Maintain established relationships with vendors to ensure successful backsourcing.
   – Facilitate knowledge transfer by integrating vendor and client resources.
   – Employ collaborative methods, such as pair programming, for practical learning.
2. **Leadership and Management**
   – Ensure the whole organization is aligned with the backsourcing agenda.
   – Develop a shared strategy to motivate and ensure collaboration across departments.
   – Focus on outcomes, strategy, and goal-setting to facilitate insourcing decisions.
3. **Process**
   – Replace outdated management and development processes with light, agile methods.
   – Increase transparency in processes, roles, and decision-making to enhance collaboration.
4. **Communication and Coordination**
   – Develop platforms for experience sharing and coordination among teams.
   – Address shared technical needs with unified solutions to avoid silos between teams and save costs.
5. **Managing Legacy Systems**
   – Implement gradual transfer of ownership to build competency within teams.
   – Use cost savings from backsourcing to phase out expensive legacy systems.
   – Balance developing new functionalities with removing technical debts accrued over time.

**Fig. 10** Lessons learned

## 10 Discussion

To our knowledge, this paper presents the first in-depth case study of backsourcing in software engineering. Thus, a very detailed comparison to other cases is not possible. However, in the following, we discuss the reasons, process, challenges and outcomes we found in contrast to the literature, including non-software development backsourcing.

### 10.1 Reasons for Backsourcing

With respect to reasons for backsourcing, our findings reinforce earlier research identifying high development costs and lack of control as primary reasons for backsourcing (Kotlarsky and Bognar 2012; Moe et al. 2014; Molléri et al. 2023). Consistent with (Nujen et al. 2018), our respondents also emphasized knowledge asymmetry with vendors as a major concern. However, in contrast to cases reported in other domains (Young and Macinati 2012; Falasco et al. 2024), improving quality was not regarded as a primary motivator in our study, although some respondents reported challenges with internal quality as a result of backsourcing.

Other factors mentioned as reasons for backsourcing in the literature include issues with vendor competencies (Hirschheim and Lacity 2000, 2006), new regulations making backsourcing difficult or impossible (von Bary and Westner 2018) vendor lock-in with outdated technology (Wong 2008), as well as changes in leadership or strategy (Raghuram 2016). In this case study, none of these factors were drivers of the backsourcing decision. The organization got a new IT Director tasked with backsourcing and became the driving force behind it, but this was not a driver for the decision, but rather a result of the organization considering the development costs to be too high.

In contrast to many cases in the literature reporting problems with the client-vendor relationship as a driver for backsourcing (e.g., Whitten and Leidner 2006; von Bary and Westner 2018; Moe et al. 2014), neither the client nor the vendors reported such issues in our study. On the contrary, both sides reported that the relationship stayed positive during and, in several cases, after the backsourcing process. This indicates that NAV viewed vendor collaboration as a relational issue, rather than a transactional one. Reflecting on known barriers to backsourcing, it is clear that this close collaborative culture helped lower the barriers related to a lack of internal capabilities, dependency on the vendor, and high switching costs.

In their systematic literature review, Molléri et al. (2023) report that most cases in the literature reported several reasons behind each backsourcing decision. In the case studied in this paper, the clearly most salient driver was the need to reduce cost. This is particularly interesting, as this often is the main driver behind many outsourcing decisions; however, as discussed, it was not the main driver for the original outsourcing decision in this case.

### 10.2 Backsourcing Approach

We outlined the approach to backsourcing based on six key principles. Our findings align with Moe et al. (2014) and Nujen et al. (2015) regarding the importance of incremental takeovers and building internal capacity. However, we found the autonomy granted to each application area to be unusually high compared to the more centralized strategies described by Hirschheim and Lacity (2006).

We next discuss these principles in relation to the sub-processes of backsourcing described in Molléri et al. (2023):

**Change Management** The change management activities include planning, internal communication, and post-backsourcing activities. Related to this process, the organization developed a sourcing strategy to transfer their systems to a higher level of responsibility and ownership (see Section 6.2). The IT director actively communicated the backsourcing decision, which was readily accepted due to previously recognized high IT development and maintenance costs, and also discussed in public forums such as the parliament and national press.

Instead of creating detailed, prescriptive plans for the backsourcing efforts, the organization delegated the responsibility for this to the various application areas (see Section 6.3). However, the efforts were too strongly grounded in the IT department, leading to challenges in integrating the business side of the organization and impacting product ownership and resourcing from the business side. This challenge was previously mentioned in Butler et al. (2011). The organization did not have any clear way of monitoring or systematically assessing the outcomes of the backsourcing, only conducting this analysis two years later.

**Vendor Relationship Management** This case highlights a particularly interesting approach to vendor management, driven only by a need to reduce cost rather than challenges with vendor competencies or relationship issues. Vendor relationship management handled centrally, and vendors were treated very well during the whole process (Section 6.6). Contracts were not terminated; instead, they were left to expire naturally, and optional extension years were utilized to ensure a seamless transfer of ownership without service disruption. During the backsourcing process, the organization began using time-and-material type contracts with many vendors. Under this “capacity contracts” model, the organization hired individual resources for the newly formed development teams post-backsourcing (Section 6.5).

**Competence Building** Competence building was an activity delegated to each application area. Thus, each area planned what competencies were needed and how to acquire them - either through retraining existing personnel or through recruiting (Section 6.4). Knowledge transfer was done mainly at the team level. The main mechanisms included training sessions led by vendors who had developed a system, workshops on specific topics, and collaborative teamwork. A key aspect was forming joint teams consisting of NAV and vendor personnel who sat together. Further, some vendor resources continued in the teams under new contracts, facilitating long-term competence building.

**Organizational Build-up** The organizational build-up was handled similarly to competence building: each area planned how many personnel were needed, what roles they would fill, and how to acquire them. As recruiting was handled centrally, each area developed plans and descriptions for the needed positions to HR and was later involved in the interviewing and hiring process (Section 6.4). Key strategies included avoiding direct recruitment from the vendor and managing personnel retention without requiring staff relocation.

**Transfer of Ownership** As stated by Section 6.7, transfer of ownership was done incrementally, addressing one system at a time, or in the case of large systems, one component at a time. This helped ensure the continuity of services and lower the risk of major issues. System operations, which have been managed in-house since the formation of NAV, have undergone major changes in the responsibility for maintenance and fault correction, now handled by internal teams.

While we do not claim that this list of sub-processes is exhaustive, they cover the main activities of backsourcing as described in the literature. The principles we identified were easily mappable to these sub-processes, and we hypothesize that they can be used to structure and guide backsourcing efforts in other organizations, as well.

### 10.3 Challenges

The main challenges reported in the literature were related to vendor relationship management, contracting, knowledge transfer, building the competencies required, and integrating the new in-house organization, as discussed in Section 7.

In this case, vendor relationship management, contracting, and knowledge transfer activities did not pose significant challenges due to the good relationship with the vendors. Interestingly, the good relationship was mentioned by both the representatives of the backsourcing organization and the vendors.

Building new competencies posed challenges related to hiring enough competent personnel, training them in systems, technologies, and the domain, as well as retraining and retaining existing personnel during the reorganization. All of these issues have been previously mentioned in the literature, as discussed in Section 7

In addition, the organization encountered many challenges related to change management, which were not mentioned in the backsourcing literature but have been covered both in the agile transformation literature (e.g. Dikert et al. 2016; Edison et al. 2021), and the literature on global software development (e.g. Niazi et al. 2016). Our results reinforce the important lesson from change management of involving top management and committing the whole organization to the change effort. In this case, the IT Director was mainly responsible for both the original outsourcing decision as well as for the Backsourcing effort.

### 10.4 Outcomes

As an overall effort, backsourcing was considered very successful within the organization. The fact that a significantly higher number of systems had been backsourced than initially planned, and the fact that the organization successfully took over all backsourced systems without breaking service continuity, can be considered indicators of this. Further, the number of incidents in the field declined after backsourcing. The transfer of ownership and responsibilities can thus be considered successful.

Furthermore, NAV established a new organization by acquiring skilled professionals from the market. Knowledge transfer was conducted successfully, and existing in-house knowledge was largely retained through both continuing relationships with experienced consultants and retraining of existing personnel. As a result of backsourcing, the organization reported having better control of development and related priorities.

Assessing the outcome on quantitative metrics proved difficult, mostly due to the lack of suitable data. Contrasting the outcomes with the reasons for backsourcing, we noted that the main reason behind the effort was the need to reduce costs. However, as discussed in the results, the overall IT costs have not decreased. On the other hand, we can see a marked increase in the effort put into development, and many of our informants reported that they think that productivity now is better, or at least at the same level as before backsourcing. However, we were unable to calculate any meaningful productivity measures due to a lack of data. What we were able to demonstrate is that the number of deliveries has increased significantly and that the quality, as measured by field incidents, has improved.

A positive outcome that we did not find in the literature was the reduction of people involved in managing the contracts and subcontracting activities necessary when working with outsourcing. Our qualitative data showed that many people felt that work had become significantly more productive. Still, as discussed in the results, we were unable to demonstrate productivity increases quantitatively due to a lack of data. In particular, we can see that overall IT costs have not decreased, even though cost savings were the primary motive for initiating back-sourcing.

We were also unable to show lower maintenance costs, as they are no longer separated from the development costs in the new process, where internal teams do both development and maintenance. The operational costs remain relatively stable at the same level as before, which is not surprising, as the operation of the systems has been handled by the case organization both before and after outsourcing. We also did not see lower recruiting costs, as they were significantly higher than before due to the need to establish the organization as an attractive employer and the large number of recruitments that had to be made during the process.

Regarding negative outcomes mentioned in the literature, we saw extra refactoring costs, as the technical debt level was considered very high in many systems. It is, however, debatable whether this cost would have been incurred even if the outsourcing mode had continued, as at some point, the technical debt must be repaid regardless of the sourcing model. There were also additional costs due to reorganization, as many new roles were created, and establishing a new organization incurs various expenses.

Negative outcomes reported in the literature that we did not identify in our case include a lengthy negotiation process for back-sourcing agreements, unsuccessful knowledge transfer, additional management costs, and service interruptions.

Market advantages and expanded offshore business, which are the outcome factors identified in the literature, were not relevant to this case, as the case organization did not have any offshore business and is, by law, the sole provider of the services it delivers, thus holding a monopoly position in the market.

### 10.5 Limitations and Validity

Like all empirical research, this study has several limitations. Most significantly, as a single case study in a particular organizational context, the findings are necessarily bound to that context and may be difficult to generalize to other contexts, such as private sector organizations or organizations of different sizes and structures. Generalization based on case studies is based on analytical, not statistical logic. We have attempted to increase generalizability by presenting principles of backsourcing that we believe are likely to be valid in other cases as well, as they are closely linked to the activities identified in the literature, and are at a rather high level of abstraction. This can be seen as our attempt at analytical generalization (e.g., Yin 2009). The detailed reporting of both the context and results of the study helps facilitate naturalistic generalization (Lincoln and Guba 1985). Further studies are needed to validate the principles and findings in other contexts, and to help assess whether the abstraction level presented here is appropriate for other cases.

Qualitative studies relying on semi-structured interviews introduce the risks of respondent bias and inaccuracies in recall (Patton 2014). Although we used data-source triangulation (Jick 1979) to help validate our findings, it is possible that the roles of the people interviewed in the actual backsourcing effort, as well as post-fact rationalization of the process and experiences could have affected their responses.

Furthermore, we primarily interviewed individuals involved in the backsourcing initiative, which may have led to the underrepresentation of critical perspectives from people affected by, but not directly involved in, the initiative. Thus, while we attempted to document the challenges and issues objectively, and we think our interviews were very open in their stories, the picture reported might still be somewhat idealized.

Studying a phenomenon that occurs over a significant time span in an organization also presents challenges when it comes to isolating the effects of the study's object from other concurrent changes and situations within the organization. Thus, some of our outcomes might be colored by factors that are not directly related to backsourcing.

Next, we discuss threats to validity from the four viewpoints presented by Yin (2009): internal, construct, external validity, and reliability.

Internal validity concerns the correctness of causal relationships in reporting the results. In this paper, we report causalities as claimed by our subjects. The threat is that this might not perfectly represent reality. We mitigated this threat by reporting the case as a descriptive case study without claiming causal relationships.

Construct validity concerns how well the case description represents reality. We interviewed subjects involved in the transformation, giving us first-hand accounts of the changes. As in all interview studies, there are risks related to respondent bias and recollection. We mitigated this risk mainly through data source triangulation (Jick 1979) by interviewing several subjects in different subcases and roles and by complementing the interviews with information from corporate documents and databases.

External validity concerns the potential for generalization of the results. As this case study does not provide any theory, we do not claim universal generalizability. Instead, we provide a rich context description to support theoretical and naturalistic generalization.

Reliability refers to whether other researchers conducting the same study would obtain the same results. The primary questions here concern variability in data collection and analysis. We mitigated this threat by using the same data collection instrument and analysis procedures over the whole material. We conducted the interviews in a similar fashion. Having an author representing the case company, as in our case, can introduce researcher bias. We managed this through joint analysis and documentation. Furthermore, she was not involved in the actual backsourcing effort in any way, thereby reducing the risk of bias. We think that her participation, through insider knowledge and access, far outweighed the risks of bias. Further, coding and analysis were performed collaboratively to ensure that interpretations were not unduly influenced by prior knowledge or biases.

Finally, as discussed in this paper, we requested feedback from the organization on our results. The paper was read by several employees of the organization with in-depth knowledge about the backsourcing effort. We received no change requests, as the results were considered to reflect the organization's situation and backscourcing journey accurately.

## 11 Conclusions

In this paper, we presented an in-depth case study of backsourcing in a large public sector organization, providing an analysis of the reasons, approach, challenges, and outcomes related to this significant decision. We have exposed how the organization approached the transition from outsourced to insourced development, providing insights into this previously understudied phenomenon in software engineering at a level of detail not seen in the existing literature.

The primary motivation for backsourcing in the case organization was the need to reduce costs, which was primarily achieved throught the elimination of expensive external consultants, and the reduced administration related to insourced vs. outsourced development. However, interestingly, our study revealed that cost reduction was not a major benefit of the effort, and the organization was not able to provide data that would have clearly evidenced to what extent that goal was achieved. Other major benefits reported by the organization included better control of the whole lifecycle of the IT systems, improved ability to deliver software quickly, and improved quality — both internal and external - through more focus on technical debt.

An important finding of the study is the importance of creating and communicating a clear strategy driving the backsourcing effort. The development and dissemination of the sourcing strategy provided the decision rationale and goal state, helping to align the practical efforts of the different application areas. Furthermore, it enabled the application areas to exercise wide autonomy in implementing the strategy, allowing them to tailor the approach to their specific circumstances.

We also emphasize the importance of establishing a competent internal organization that can assume the responsibilities of external vendors. This included a strong push to recruiting new personnel, retaining existing experts, and to ensure effective knowledge transfer from the vendors to the new internal teams. Our study showed the need for strong support from the organizational level to the application areas for key processes related to recruitment, contract management, and the institution of agile practices.

While the outcome of the backsourcing effort was perceived as mostly positive, we also identified several challenges during and after the process. During the process of backsourcing, recruiting and training people required extensive effort from the whole organization. Another challenge is motivating and involving the organization, even people not directly involved in IT development. One challenge both during and after backsourcing, is finding the right balance between internal and external people, so that core knowledge is kept inside the organization while vendors can fill the knowledge gaps faster and cheaper. Another challenge is to manage expectations regarding the outcomes, specially related to cost.

Through this study, we contribute to the currently limited body of knowledge on backsourcing in software engineering, providing empirical evidence of the viability of backsourcing in large and complex contexts. However, as a single case study, the generalizability of the findings is limited, but we think that the principles used for backsourcing are likely to be valid also in other cases. The rich case description is provided to support such natural generalization. Future research with further in-depth studies in both the private and public sectors is needed to deepen and further validate the insights presented in this paper. More data can contribute to further analysis of factors contributing to the success or failure of backsouring efforts, as well as how to conduct such initiatives successfully.

In conclusion, this study demonstrates that backsourcing can be a powerful strategic tool for organizations seeking to regain control over their IT systems and enhance time-to-market and software quality. We also believe that the approach is likely to lead to significant cost savings, although we were unable to demonstrate this in this case, primarily due to the lack of output data. The success of backsourcing efforts depends on a clear strategy, an approach that adapts to the specific needs of different areas in a large organization, combined with organizational support for key processes. As more and more organizations come to realize that IT is a core function now and in the future, backsourcing may become an increasingly attractive option for organizations that currently rely on outsourcing.

This paper opens up several venues for future research. Studies could evaluate and improve on the principles outlined here, and assess their applicability in various contexts. Studies could also look into ganing a more in-depth understanding between the principles and activities of backsourcing and the outcomes, paving ground towards building a theory of backsourcing. Frameworks for analyzing the cost-benefit of backsourcing decisions would be valuable for organizations considering such decisions. A deeper comparison between the fields of IT and software engineering, and backsourcing efforts in other fields might point out areas for cross-pollination between fields. Finally, identifying easily measurable quantitative metrics related to outsourcing costs and benefits would provide a sound basis for comparing cases.

## Appendix A: Interview Guide

- Introduction Who we are / Introductions
- Goals of the study
- Definition of backsourcing: bringing back previously outsourced activities in-house
- Confidentiality, data collection, analysis, usage and storage. GDPR, consent form.
- Agreement to record interview

- Demographics / Personal Background Name, education
- Professional background before NAV
- Professional experience in NAV
- Current role and responsibilities

- Background for Backsourcing (pre-backsourcing, decision making) When was the first time you heard about backsourcing in NAV?
- Why did backsourcing become of interest to NAV? (Documentation? Analysis of problems/challenges/improvement possibilities related to sourcing that backsourcing could help solve?)
- Who were involved in the discussion at that time?

- What activities were conducted to prepare for backsourcing?
- What kind of analyses were done? By whom?
- Did you conduct a risk analysis? What risks were you aware of?
- Were alternatives to backsourcing evaluated/considered (e.g., to change contracts, stronger partnerships, change or vendors, etc.)?
- What was your role at this stage?
- What were the goals of backsourcing?
- What problems were backsourcing supposed to solve?
- When were the decisions related to backsourcing made?
- Who made the final decision to backsource? At what level?
- Were the ministry or any politicians involved?
- How widely was the decision discussed? (Management, personnel representatives, product owners, human resources, ...)?
- Did you notice any reaction, and if so, what kind by the vendors?

- Backsourcing "process" (implementation, transition phase) Were you involved in the implementation of backsourcing? What was your role?
- How was the decision communicated to the organization? To vendors? Externally?
- How was the backsourcing initiative organized within NAV?

    - Responsibilities, processes, activities
    - Thematic areas

        Contract management (changes to, or termination of, the contracts)
        Knowledge transfer and competence development

        Recruiting and organizational build-up
        Transfer of ownership of the applications

- Was the backsourcing process planned in detail beforehand?
- Did you conduct a pilot case?
- What do you think NAV did well in the backsourcing process?
- What were the major challenges you faced in backsourcing? How did you solve them?
- How different has the process been in different areas? Differences between contracts / application areas?
- Was the backsourcing effort conducted systematically? Or was it more ad-hoc? Why?

- Outcomes and evaluation (post-backsourcing) Have you evaluated the success of backsourcing post-hoc? Did you do any anlyses? What kind? Can you show us?
- How successful do you think NAV was in backsourcing?
- Are there areas in which NAV has not been successul, or still have major challenges?
- Do you think backsourcing was the right decision for NAV? Why or why not?
- Did you observe any risks that were not accounted for. How did you handle them?
- Drawbacks?
- How satisfied are people with backsourcing - managers, middle managers, other personnel. How do you know that?

- How well does collaboration with the vendors work post-backsourcingv
- Has there been any effects/consequences for end-users? For operations? What?
- What effects has the backsourcing had for the organization? (Structure, strategy, culture, ). Contrast pre- and post-backsourcing.
- What kind of applications are most suitable for backsourcing? Why?

- Close-out What advice would you give other organizations that are considering backsourcing? (Explain and motivate each piece of advice)
- In retrospect: What should you have done differently? Why? Did you make any big mistakes? What?
- What do you think NAV will look like in 5 years?
- Is there something else you would like to mention that you think is important that we did not ask you about?
- Thank you very much for your time!

- Remember Ask for material shown and/or identified during the meeting.
- Take pictures of drawings etc.

**Author Contributions** Casper Lassenius: research design, data collection, data analysis, writing. Parastoo Mohagheghi: research design, data collection, data analysis, writing. Jefferson Molleri: data collection, data analysis, writing.

**Funding** Open Access funding provided by Aalto University. This study was funded by the EDOS project at the Simula Metropolitan Center for Digital Engineering. Parastoo Mohagheghi receives a salary from the case organization, where she is a senior advisor.

**Data availability statement** The data used in this paper is not publicly available for confidentiality and privacy reasons. Reasonable requests for sanitized subsets of the data will be honored and can be sent to the corresponding author. The codebook is available at https://doi.org/10.6084/m9.figshare.29264501.v1.

## Declarations

**Ethical Approval** Not applicable.

**Informed consent** Participants gave written consent to participate in the study.

**Conflicts of Interests** Parastoo Mohagheghi was an employee of the case organization while conducting the study. The other authors have no conflicts of interest.

**Clinical trial number** Not applicable.

**AI statement** Grammarly was used for proofreading and language checking.

## References

Barney HT, Aurum A, Low GC, Wang K (2013) Investigating post-outsourcing decisions: using the intellectual capital view. In: Building sustainable information systems. Springer, pp 63–75
Bergkvist L, Fredriksson O (2008) Outsourcing terms: a literature review from an ISD perspective. In: Proceedings of the European Conference on Information Systems (ECIS). AISNET, p 13
Butler N, Slack F, Walton J (2011) IS/IT backsourcing-a case of outsourcing in reverse? In: 2011 44th Hawaii international conference on system sciences. IEEE, pp 1–10
Chakrabarty S, Wakefield R (2010) The strategic choice to continue outsourcing, switch vendors, or backsource: do switching costs matter? Inf Manag 47(3):167–175
Corbin J, Strauss A (2014) Basics of qualitative research: techniques and procedures for developing grounded theory. Sage Publications
Dibbern J, Goles T, Hirschheim RA, Jayatilaka B (2004) Information systems outsourcing: a survey and analysis of the literature. Data-Base Adv Inf Syst 35(4):6–102
Dikert K, Paasivaara M, Lassenius C (2016) Challenges and success factors for large-scale agile transformations: a systematic literature review. J Syst Softw 119:87–108
Edison H, Wang X, Conboy K (2021) Comparing methods for large-scale agile software development: a systematic literature review. IEEE Trans Softw Eng, pp 1–1
Ejodame K, Oshri I (2018) Understanding knowledge re-integration in backsourcing. J Inf Technol 33(2):136–150
Falasco S, Caputo P, Garrone P (2024) Can geographical indications promote environmental sustainability in food supply chains? insights from a systematic literature review. J Clean Prod 444:141100
Hirschheim R, Lacity M (2000) The myths and realities of information technology insourcing. Commun ACM 43(2):99–107
Hirschheim R, Lacity MC (2006) Four stories of information systems insourcing. In: Information systems outsourcing. Springer, pp 303–346
Jansson M, Carlström E, Karlsson D, Berlin J (2021) Drivers of outsourcing and backsourcing in the public sector—from idealism to pragmatism. Financ Account Manag 37(3):262–278
Jick TD (1979) Mixing qualitative and quantitative methods: triangulation in action. Adm Sci Q 24(4):602–611
Kotlarsky J, Bognar L (2012) Understanding the process of backsourcing: two cases of process and product backsourcing in Europe. J Inf Technol Teach Cases 2(2):79–86
Lacity M, Hirschheim R, Willcocks L (2007) Realizing outsourcing expectations incredible expectations, credible outcomes. Inf Syst Manag
Legard R, Keegan J, Ward K (2003) In-depth interviews. Qualitative Research Practice A Guide for Social Science Students and Researchers 6(1):138–169
Leyh C, Schäffer T, Nguyen TD (2018) Information system backsourcing: a systematic literature analysis. In: 2018 Federated Conference on Computer Science and Information Systems (FedCSIS). IEEE, pp 1–10
Lincoln YS, Guba EG (1985) Naturalistic inquiry, vol 75. Sage
Misztal P (2024) Backsourcing as a new trend among companies in the european union. J Appl Econ Sci (JAES) 19(3(85)):291–303
Moe NB, Šmite D, Hanssen GK, Barney H (2014) From offshore outsourcing to insourcing and partnerships: four failed outsourcing attempts. Empir Softw Eng 19(5):1225–1258
Mohagheghi P, Lassenius C (2021a) Organizational implications of agile adoption: a case study from the public sector. In: Proceedings of the 29th ACM joint meeting on european software engineering conference and symposium on the foundations of software engineering, ESEC FSE 2021, pp 1444–1454
Mohagheghi P, Lassenius C (2021b) Organizational implications of agile adoption: a case study from the public sector. In: Proceedings of the 29th ACM joint meeting on european software engineering conference and symposium on the foundations of software engineering. ACM, Athens Greece, pp 1444–1454
Mohagheghi P, Lassenius C, Bakken IO (2020) Enabling team autonomy in a large public organization. In: Paasivaara M, Kruchten P (eds) Agile processes in software engineering and extreme programming - workshops - XP 2020 workshops, Copenhagen, Denmark, June 8-12, 2020, Revised Selected Papers', vol 396 of Lecture Notes in Business Information Processing. Springer, pp 245–252
Molléri JS, Lassenius C, Jørgensen M (2023) Backsourcing of it with focus on software development—a systematic literature review. J Syst Softw 204:111771
Niazi M, Mahmood S, Alshayeb M, Riaz MR, Faisal K, Cerpa N, Khan SU, Richardson I (2016) Challenges of project management in global software development: a client-vendor analysis. Inf Softw Technol 80:1–19
Nicholas-Donald A, Osei-Bryson K-MA (2017) The economic value of back-sourcing: an event study
Nujen BB, Halse LL, Damm R, Gammelsæter H (2018) Managing reversed (global) outsourcing-the role of knowledge, technology and time. J Manuf Technol Manag 29(4):676–698

Nujen BB, Halse LL, Solli-Sæther H (2015) Backsourcing and knowledge re-integration: a case study. In: Umeda S, Nakano M, Mizuyama H, Hibino H, Kiritsis D, von Cieminski G (eds) Advances in production management systems: innovative production management towards sustainable growth. IFIP Advances in Information and Communication Technology, Springer International Publishing, Cham, pp 191–198
Patton MQ (2014) Qualitative research & evaluation methods: integrating theory and practice. Sage Publications
Petalidis N (2018) Lessons from attempting to backsource a government IT system. J Inf Technol Teach Cases 8(1):90–96
QSR International (2020) NVivo. QSR International Pty Ltd
Raghuram KS (2016) Mechanics of making or buying—pulling back the information technology engineering in-house. In: 2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT). IEEE, pp 2284–2288
Rajini D, Kaluarachchi KASP (2024) Determinants of outsourcing trajectory: a literature review
Runeson P, Höst M (2008) Guidelines for conducting and reporting case study research in software engineering. Empir Softw Eng 14(2):131
Schwaber K, Sutherland J (2020) The scrum guide: the definitive guide to scrum: the rules of the game
Šmite D, Wohlin C, Galviņa Z, Prikladnicki R (2014) An empirically based terminology and taxonomy for global software engineering. Empir Softw Eng 19(1):105–153
Solli-Sæther H (2016) Modenhet i outsourcing, offshoring og backsourcing: Tilbake til fremtiden?
von Bary B (2018) How to bring IT home: developing a common terminology to compare cases of IS Backsourcing
von Bary B, Westner M (2018) Information systems backsourcing: a literature review. J Inf Technol Manag 29(1):62–78
von Bary B, Westner M, Strahringer S (2018) Adding experts' perceptions to complement existing research on information systems backsourcing. IJISPM-Int J Inf Syst Project Manag 6(4):17–35
Whitten D, Leidner D (2006) Bringing IT back: an analysis of the decision to backsource or switch vendors. Decis Sci 37(4):605–621
Wong SF (2006) Bringing IT back home: developing capacity for change. ICIS 2006 Proceedings, p 40
Wong SF (2008) Understanding IT backsourcing decision. PACIS 2008 Proceedings, p 226
Yin RK (2009) Case study research: design and methods, 4th edn. SAGE Publications, Thousand Oaks, CA, USA
Young S, Macinati MS (2012) Health outsourcing/backsourcing: case studies in the australian and italian health sector. Public Manag Rev 14(6):771–794



## Authors and Affiliations

**Casper Lassenius[1,2] · Parastoo Mohagheghi[3] · Jefferson Seide Molléri[4]**

✉ Casper Lassenius
casper@simula.no; Casper.Lassenius@aalto.fi

Parastoo Mohagheghi
Parastoo.Mohagheghi@nav.no

Jefferson Seide Molléri
Jefferson.Molleri@kristiania.no

1 Simula Metropolitan Center for Digital Engineering, Oslo, Norway

2 Department of Computer Science, Aalto University, Espoo, Finland

3 Norwegian Labour and Welfare Administration, Oslo, Norway

4 Kristiania University, Oslo, Norway