Leiden University

ICT in Business and the Public Sector

Organizational design patterns for managed service providers that implemented automatic IaaS provisioning

Name:	Yorick Julian Molema
Student ID:	s3679071
Date:	06-06-2025
1st supervisor:	Christoph Johann Stettina
2nd supervisor:	Joost Visser
Company supervisor:	Pieter-Jan Nefkens

MASTER'S THESIS

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Einsteinweg 55
2333 CC Leiden
The Netherlands

Abstract

Background: Infrastructure as a service (IaaS) providers play a critical role in today's cloud computing landscape. Driven by rapid market growth there is an increasing importance of standardization and automation. The rising importance of standardization and automation requires significant changes across multiple organizational dimensions in IaaS providers. However, despite its importance, there is a notable lack of academic research specifically addressing the organizational design of IaaS providers, particularly from a business and operational perspective.
Aim: This thesis aims to fill this gap by developing practical guidance for IaaS providers undertaking organizational redesigns to support automation and standardization. Specifically, it aims to identify and analyze organizational design patterns that enable efficient and scalable IaaS provisioning, providing recommendations to guide providers through their transformation journey.
Method: A multiple-case study was conducted involving 14 IaaS providers and 14 interviews, resulting in a total of over 1,200 minutes of recordings. The data were analyzed to identify recurring organizational design patterns across cases, the team topologies within these patterns, and critical factors of the patterns.
Results: Four organizational design patterns were identified: (1) platform-driven self-managed platform organizations, (2) platform-driven managed service provider organizations, (3) solution-driven managed service provider organizations, and (4) solution-driven Dev & Ops provider organizations. The study extended the team topologies model by introducing two new sub-configurations of stream-aligned teams: technology-based and process-based teams. Furthermore, six critical factors —service delivery, operational governance, maturity, organization size, infrastructure capabilities within application teams, and size of on-premises cloud — were found to characterize these patterns and inform organizational design choices.
Conclusion: As the IaaS sector continues to expand rapidly and adopt standardization and automation, these organizational design patterns provide essential guidance for managed service providers to structure their teams and processes effectively. The proposed framework supports IaaS providers in scaling operations and automating provisioning, thereby enabling sustainable growth and improved service delivery in an increasingly complex cloud infrastructure landscape.

Acknowledgements

First of all, I would like to sincerely thank Christoph Stettina, my primary university supervisor. I am grateful for his continuous guidance and the many one-on-one meetings we have had over the past 1.5 years. Due to personal circumstances, my thesis took longer to complete than expected, and throughout this process, Christoph has been a constant source of support. I especially appreciate the way he provided help not only on the content but also on a personal level during our meetings as well. His dedication to aim for the best has inspired and motivated me throughout the writing of this thesis. Additionally, I am thankful for the valuable connections he facilitated and I am glad to continue our collaboration with the host organization.

Secondly, I would like to thank Joost Visser, my secondary university supervisor, for his valuable input during our check-ins and for his helpful tips on case organizations and interesting insights related to government topics.

Third, I would like to thank the host organization for hosting my thesis and for the freedom and trust I was given throughout the process. I am especially grateful to Annemiek Beijer for offering me this opportunity and for her understanding and flexibility despite the delays. I would like to thank one person in particular: Pieter-Jan Nefkens. He has supported me first as a working student and later as my supervisor for this thesis. His personal support has meant a lot to me, and our many conversations have been a true source of inspiration. I deeply appreciate the time and effort he dedicated to my personal and professional development, and I look forward to continuing our collaboration. I hope to keep learning from him in the future. I also want to thank Hans van der Linden for giving me the space to continue this project as I transitioned into a working role. Lastly, I extend my gratitude to all my colleagues at the host organization for their openness, valuable conversations, and ongoing support throughout this journey.

Contents

1 Introduction

Infrastructure as a service (IaaS) providers are challenged to redesign their organizations as they shift from customization to standardized and automated services [1] in response to growing demand. Despite the critical nature of this transformation, IaaS is the least studied cloud service model from a business and organizational perspective [2], indicating a significant gap in the current body of research. Organizational design is a critical factor for IaaS providers seeking to operate efficiently and provision scalable resources [3]. Historically, the IT infrastructure landscape was highly diverse [4], a characteristic that was reflected in the service models of infrastructure providers. However, the rise of cloud computing has driven a pronounced shift towards standardization, positioning standardization as a core capability for IaaS providers [5]. Fueled by rapid market growth [6], IaaS providers have increasingly prioritized standardization and automation. This shift impacts various organizational dimensions, as core competencies must shift from customization to standardized and automated provisioning processes. Consequently, IaaS providers must adapt new organizational designs to accommodate these changing demands.

Currently, there is a lack of clear guidelines on how IaaS providers should approach this organizational redesign. However, these guidelines are necessary, because of increasing demand. The cloud has played a crucial role in making IT services more accessible over the past few years and continues to do so. Cloud services continue to grow significantly: in 2025, the end-user spending on public cloud services is expected to exceed $700 billion, a growth of more than 20% compared to 2024 [7]. IaaS forms the foundational layer of the cloud computing stack, providing customers with access to essential computing resources such as processing power, storage, networking, and other fundamental infrastructure components [8]. The process of making these resources available to end users is known as provisioning. IaaS provisioning typically involves the on-demand deployment—including the allocation, configuration, and activation—of physical and virtual infrastructure, leveraging advanced networking and virtualization technologies. Automated IaaS provisioning refers to the delivery of resources without manual intervention, using predefined workflows and orchestration tools. Automation is key to rapid and scalable IaaS. Notably, both IaaS and sovereign IT infrastructure cloud services are projected to grow at an annual rate of 36%, significantly outpacing the average growth rate of the broader cloud services market [6]. To keep up with increasing demand, IaaS providers must implement automated IaaS provisioning and embrace standardization. This transformation affects multiple organizational dimensions, requiring a redesign of the organization. This highlights the need for clear guidance to support organizational redesigns.

This thesis contributes to the development of this guidance by studying existing organizational designs and creating a model to guide IaaS providers in their organizational redesign. This study is conducted in collaboration with a department within a large IT service provider within Dutch government which functions as an IaaS provider. The host organization is a large IT service provider of the Dutch government, their IaaS department is currently undergoing an organizational redesign, it has a strong interest in practical guidance to support this transformation. Organizational designs from current IaaS providers can offer valuable guidelines and practical examples for structuring their organizations to deliver more standardized and automated services. To this end, we conduct an exploratory multiple case study involving 14 organizations. Based on this research, we propose four organizational design patterns intended to guide IaaS providers in their transformation efforts. These patterns address various organizational dimensions and provide actionable recommendations to help organizations enhance scalability, improve operational efficiency, and successfully implement automated IaaS provisioning.

1.1 Focus and scope

This thesis focuses on a specific type of IaaS provider, the managed service provider. An MSP is a cloud service provider that delivers and proactively manages cloud services for their customers [9, 10]. More and more organizations are outsourcing their complex IT infrastructure landscape to managed service providers (MSPs) to reduce complexity [10]. In this thesis, we focus on Infrastructure as a Service (IaaS) offered by a managed service provider (MSP).

On a high level, an organizational design can be defined as “the deliberate process of configuring structures, processes, reward systems, and people practices to create an effective organization capable of achieving the business strategy”[11, p.1]. The six categories of the STAR model: strategy, capabilities, structure, processes, rewards and people [11] serve as a basis for organization design. Furthermore, security is added as a separate part of the organizational design. We see the organizational structure is seen as the most important part of the organizational design. For the other components of the organizational design more high-level answers are provided.

As structure is seen as the main component of the organizational design, it is important to be able to precisely describe it. Team topologies are used for this. The team topologies model describes four team configurations: stream-aligned, enabling, complex subsystem, and platform.

The following questions will be answered in this thesis:

1. What are organizational design patterns for a managed service provider that implemented automatic IaaS provisioning?
2. What are the configurations of teams in organizational design patterns for managed service providers that implemented automatic IaaS provisioning?
3. Which critical factors should an organization consider when choosing a pattern?

1.2 Structure of document

To answer these questions, the thesis is structured as follows. First, in chapter 2, a literature review is provided in which existing literature about IaaS and organizational design is discussed. Chapter 3 present an organizational design framework which is created based on literature presented in the literature review. Chapter 4 describes the methodology for the research, in which the set-up for the multiple case study is provided. The results of the research are presented in chapter 5. In this chapter, a report per case is provided. In chapter 6, the discussion, the research questions are answered based on the results which are elaborated on in chapter 5. The answers to the three research questions form the model. The created model is then applied to a managed service provider and the team configurations are critically discussed. The thesis finishes with a conclusion in which a short summary of the main findings is given, the threats to validity are discussed and future work is recommended.

2 Literature review

Although IaaS plays a central role in today's IT landscape, academic literature offers limited research specifically addressing the organizational design of IaaS providers, and more broadly, there is a general lack of business-oriented studies on this service model [2]. Due to this gap, this literature review draws on broader sources related to organizational design. The aim is to summarize the current body of knowledge that may provide indirect insights or frameworks applicable to IaaS providers.

The literature review is structured into three main sections. The first section, digital transformations and infrastructure as a service, provides background on the evolution of digital transformation, the shift to cloud computing, and the emergence of IaaS as a dominant service model. The second section, organizational design of IT infrastructure, examines historical developments in infrastructure management, the role of managed service providers, and existing models for scalable and standardized IT delivery. The final section, organizational design criteria, outlines key concepts and frameworks related to organizational design, including strategies to prevent silos, the role of teams, and models such as team topologies.

This literature review forms the foundational layer for the rest of the thesis. The knowledge that is presented here is used to create an organizational design framework and is used in the set-up and analysis phase of the multiple case study.

2.1 Digital transformations and infrastructure as a service

2.1.1 A historical perspective on digital transformations

Digital transformations (DT) are a significant research field in information technology (IT) [12, 13]. A DT can be defined as: "acquiring the digital tool, techniques, approaches, mechanism, etc. for the transformation of the business, applications, services, and upgrading the manual process into the automation." [14, p.615] Even though this definition might be highly applicable to this thesis, a more conceptual definition is needed. In a review of 282 papers [12], a more conceptual definition is proposed: "a process that aims to improve an entity by triggering significant changes to its properties through combinations of information, computing, communication, and connectivity technologies"[12, p.118]. This process involves changing an organization's structure, processes, culture, and strategy [12]. The transformational framework of [15] provides another view on the DT process; they provide four key dimensions: structural changes, changes in value creation, the use of technologies, and how to finance DT. As structure and culture are covered in all definitions, they have an essential impact on DT, which is confirmed by [16].

From a historical point of view, several large evolutionary steps towards our current era of DTs can be identified. The creation of mainframe computers in the mid-20th century marks the root of digitalization. After the mainframe computers were further developed, personal computers were introduced during the 70s and 80s. The fact that individual users and businesses could access a computer was a monumental shift. During the 1990s, the internet era, the IT sector was revolutionized. Paving the way for e-commerce and online business models, the rise of the internet facilitated global connectivity and data exchange. Quickly after this, the first business processes became digitalized using enterprise resource planning (ERP) systems, managing business processes like accounting and inventory. IT services became even more accessible with the introduction of cloud computing at the beginning of the 2000s. With the Internet of Things (IoT), cloud computing has vastly expanded the scope of digitalization. In our current era, AI, blockchain, and big data characterize digitalization [17]. Figure 1 provides an overview of the history of DTs.

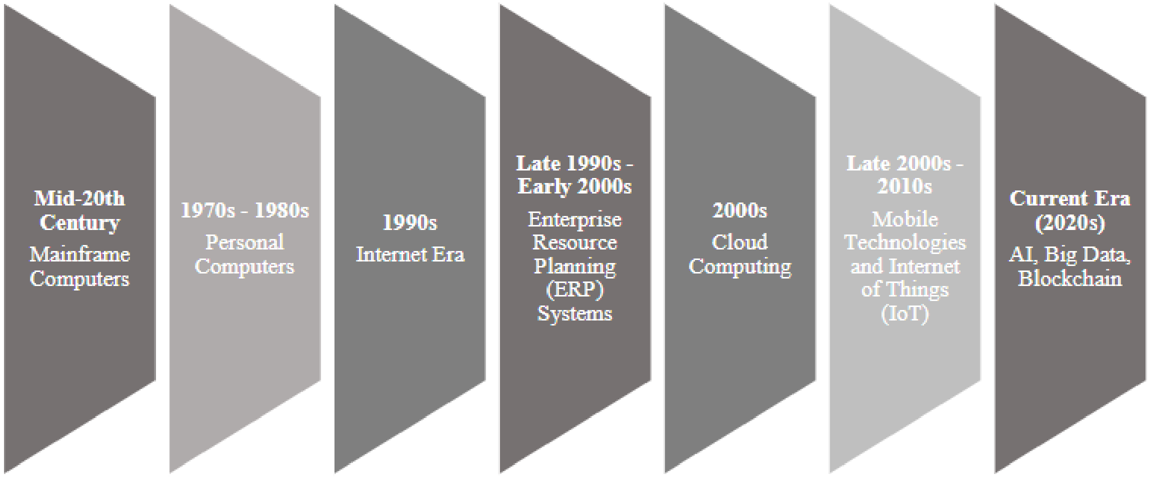

Figure 1: The dynamics evolutionary of digital transformation in the IT sector [17, p.3]

2.1.2 Moving to the cloud and automation

The foundation for cloud services was laid a long time ago. In 1967, resource sharing, an essential aspect of cloud computing, started. IBM built an operating system using virtualization techniques in which multiple users could share the same resources simultaneously. During the '70s and '80s, IT was further researched and developed, and virtualization techniques were further developed and supported by network, storage, and operating systems. In the '90s, these techniques reached a higher maturity level, and in the late '90s, the term cloud emerged. In the '90s, cloud computing was broadly known as grid computing, where resources were shared across continents; however, grid computing did not meet all cloud requirements yet. Amazon Web Services (AWS) continued developing grid computing and launched the first public cloud in 2002. As with any product, the first release had a low maturity level. In 2006, AWS released a more mature version, which is now seen as the first version of the cloud. During the 2010s, competition grew, and with that came many services and new technologies in the cloud [18].

Even though the cloud was introduced in the early 2000s, the cloud and how we use it are still transforming. Nevertheless, the concept itself did not change over the years. Cloud computing offers widespread, convenient, and on-demand access to a shared pool of configurable computing resources through a network, allowing for rapid provisioning and release with minimal need for management effort or interaction with service providers. It has five essential characteristics: on-demand self-service, broad network access, resource pooling, rapid elasticity, and measured service [8]. Cloud services can be delivered using three service models: Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS). SaaS offers the whole application stack to the customer, PaaS provides the capability to land customer applications on the cloud, and IaaS offers computing resources of the internet [8].

Cloud computing has and still is revolutionizing companies and entire industries by offering dynamically scalable and virtualized resources as a service over the internet. Cloud computing can be broadly used to automate processes. Especially with the combination of IoT, cloud computing is a powerful automation tool for industrial automation [19]. For example, when looking at traditional manufacturing processes, cloud computing can transform the complete business model, improve the alignment of product innovation with strategy, and create smarter business processes [20]. A recent example is the fully automated warehouse of Picnic, an online supermarket in the Netherlands. Cloud services play an essential role in the automation of their warehouse. Even though they tried the same with an on-premises solution, cloud services are the final solution for Picnic [21]. The cloud is a significant enabler in digital transformations, as its services significantly impact the transformation of business processes.

2.1.3 Infrastructure (as a service)

Around the beginning of the 2000s, IT was looked at slightly differently than today. IT infrastructure was defined as "a set of IT resources and organizational capabilities that are shared across the organization and that provide the foundation on which IT applications are developed and business processes are supported."[22, p.3] IT infrastructure serves as the foundational basis for shared IT capabilities, which in turn provide the groundwork for other business capabilities. To provide reliable infrastructure services, technical and managerial expertise are required. As these are difficult to acquire and develop, IT infrastructure is a determinant of the competitive advantage of companies [23]. The technical experts focus on hardware and base software platforms, middleware, communications technology, and other shared services that serve as a basis for applications [23].

Over the years, IT infrastructure has become more complicated. In the beginning, the advances in the hardware were keeping up with the developments at the application level [24]. Nowadays, IT infrastructure has become more decentralized and is moving more and more to the cloud and on-premises. The IaaS market has grown by 30% in 2022 alone and has surpassed $100 billion in revenue[25]. The previous IT infrastructure definition stated that infrastructure was shared at the organizational level, the cloud has transcended this border. Currently, organizations are able to share the same IT infrastructure.

A clear definition of IaaS is needed to study the subject in more detail. The United States National Institute of Standards and Technology (NIST) has created a broadly accepted description for the cloud and its components. Their definition of IaaS will be used to describe IaaS throughout this thesis. They define IaaS as: "The capability provided to the consumer is to provision processing, storage, networks, and other fundamental computing resources where the consumer is able to deploy and run arbitrary software, which can include operating systems and applications. The consumer does not manage or control the underlying cloud infrastructure but has control over operating systems, storage, and deployed applications; and possibly limited control of select networking components (e.g., host firewalls)."[8, p.3] This definition does not specify the components of IaaS; IaaS consists of "the facility, communication networks, physical compute nodes, and the pool of virtualized computing resources managed by a service provider."[26, p.198]. In conclusion, IaaS takes care of the infrastructure service layer in the cloud, i.e., allocation of hardware and essential services required for operating a cloud environment [27].

In more detail, IaaS "(1) Provides access to shared resources on need basis, without revealing details like location and hardware to customers, (2) provides details like server images on demand, storage, queuing, and information about other resources, among others, and (3) offers full control of server infrastructure, not limited specifically to applications, instances and containers." [27, p.425] The resources are configured to the user's needs. Broadly looking, several resource instance types can be identified: "general-purpose, compute-optimized, memory-optimized, accelerated computing (GPU), storage-optimized" [28, p.148]. GPU stand for: graphics processing unit.

When a customer starts using the cloud infrastructure, the resources have to be provided to the customer, i.e., IaaS provisioning. There is a large number of definitions in use for IaaS provisioning; as IaaS provisioning is an important term in this thesis, there is a need for a clear, well-specified definition. A list of existing definitions is given below:

1. "It is the allocation of a service provider's resources to a customer." [27, p.428]
2. "The IaaS provisioning has to deal with the allocation, configuration and activation of heterogeneous resources, which involve different virtualization technologies, as well as physical hosts, storage, software and network equipment in the cloud." [29, p.1]
3. "In the context of infrastructure as a service (Iaas), the deployment of services means selecting servers from clouds resource pool and then assigning the requested VMs to them." [30, p.24]
4. "An IaaS cloud enables on-demand provisioning of computational resources in the form of VMs deployed in a cloud provider's data center." [31, p.14]

Combining these definitions with the definitions given for IaaS results in the following definition for IaaS provisioning: *IaaS provisioning deals with the on-demand deployment (allocation, configuration,* *and activation) of physical storage and servers, network, and virtualization techniques to the customer.* This definition will be used in this thesis.

The process of IaaS provisioning is highly suitable for automation. To automate IaaS provisioning, the deployment steps, allocation, configuration, and activation need to be automated. The automation of IaaS provisioning streamlines the deployment and management of virtual infrastructure and software, empowering services to optimize the utilization of flexible cloud resources [32]. This thesis will see IaaS automation as a digital transformation (DT). When business (services) are transformed using new technologies, this can be seen as a digital transformation [14]. The transition to IaaS automation is powered by new technologies such as Ansible [33]. Therefore, IaaS automation can be seen as a digital transformation. In a digital transformation, changing an organization's structure and processes is a critical aspect [12, 15, 16].

2.1.4 Service model

NIST identifies three types of system layers that a cloud service provider (CSP) uses to deliver its service (see also Figure 2):

1. Service layer: "this is where Cloud Providers define interfaces for Cloud Consumers to access the computing services"[34, p.13]. In the service layer, an interface/service portal is used where clients can request their infrastructure services.

2. Resource abstraction & control layer: this is the middle layer in the model, which contains the "system components that Cloud Providers use to provide and manage access to the physical computing resources through software abstraction"[34, p.13]. This layer's resource abstraction component includes different resource abstraction methods, such as virtual machines or containers. The goal of the resource abstraction component is to "ensure efficient, secure, and reliable usage of the underlying physical resources"[34, p.13]. The control component of this layer includes: "resource allocation, access control, and usage monitoring". This component in the layer should connect the resource abstraction platforms below to enable efficient resource allocation. When there is a clear split in the structure of a pattern, these two components will be visualized as separate layers.

3. The physical resource layer: this is the bottom layer of the model, which includes: "hardware resources, such as compute (CPU and memory), networks (routers, firewalls, switches, network links and interfaces), storage components (hard disks) and other physical computing infrastructure elements"[34, p.13].

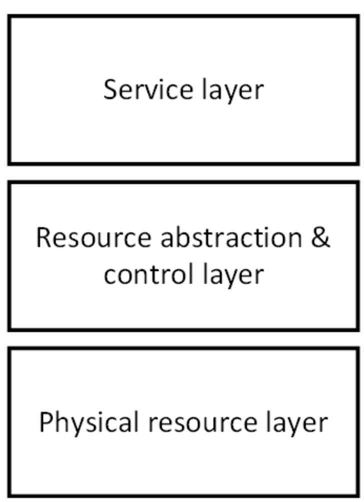

Figure 2: A visualization of the three layered service model

2.2 Organizational design of IT infrastructure

The previous section provided a foundational understanding of IaaS and its historical development. Building on that knowledge, this section explores how IaaS has been and is currently organized from an organizational perspective. However, it is important to note that existing literature on this topic remains limited.

On a high level, organizational designs can be defined as "the deliberate process of configuring structures, processes, reward systems, and people practices to create an effective organization capable of achieving the business strategy."[11, p.1] Organizational designs are viewed as a collection of coherent decisions influenced by contextual factors, such as the strategy and environment of an organization [35]. In digital transformations, changing the organizational design is a critical aspect of the transformation [12, 15, 16]. In the transformation, the organizational design should build on the technologies that empower the transformation. In addition, agile structures are often implemented to swiftly adapt to environmental opportunities and threats during the digital transformation [36].

2.2.1 History of organizational design of IT infrastructure

In the past, most IT infrastructure landscapes at companies were formed by application projects requiring specialized hardware. The specialized hardware combined with mergers and acquisitions of companies made it challenging to manage the IT infrastructure landscape. Most of the time, the same infrastructure services were being used for a set of applications. However, combining, centralizing, and integrating these services into a bigger infrastructure landscape was difficult. This resulted in a rigid, non-scalable infrastructure that firms could not benefit from. Because of this, there was a need for more flexibility in IT infrastructure [24]. In the last decades, infrastructure has moved to the cloud, resulting in higher flexibility. Furthermore, the scope of IT infrastructure has been moving past organizations' borders. In the definition provided by Xia and King, organizations are said to share their resources [22]. However, current IT organizations can also share their resources through an external provider, the cloud. The cloud is not only a more flexible solution but also more scalable. Users can scale their resources either up or down. This shift throughout history has changed how organizations manage and structure their IT infrastructure.

2.2.2 Managed Service Provider

Cloud services are offered to a customer by a cloud service provider (CSP). The managed service provider (MSP) is a specific type of cloud service provider and will be the main point of interest in this thesis. An MSP "delivers services, such as network, application, infrastructure and security, via ongoing and regular support and active administration on customers' premises, in their MSP's data center (hosting), or in a third-party data center." [9] MSPs are proactively managing the IT for their customers, this includes:"remote monitoring and management, service desk operations, incident and problems management, network operation centers, and other workflow management processes to help fix IT problems before they impact operations."[10, p.336] Furthermore, MSPs rely heavily on the central role of help desk services. These services can be split into hardware and software support, procurement, asset management, network monitoring and management, security and service management, and administration [10]. To conclude, when compared to a normal CSP, an MSP manages the services after delivering them, whereas a CSP does not. As an example, patches and updates are managed by the MSP [10]. This thesis is scoped at an organizational design for a MSP that offers IaaS through automatic provisioning.

2.2.3 Existing organizational designs for factory-like delivery in IT

Automatic IaaS provisioning can be compared to a factory-like delivery process. In a real-life factory assembly line, the product is sequentially (automatically as much as possible) put together until the product is completed. The high-level process of IaaS provisioning follows this analogy: the customer requests resources in a self-service portal, after which the MSP automatically creates the desired product and delivers it to the customer. When this process is fully automated, the process of IaaS product creation by the MSP can be compared to the factory assembly line process. As in a factory, only several products can be made. Otherwise, the factory would become too large; this is the case for automatic IaaS provisioning as well. Customers have to select their product, or parts of their product, from a shopping list. As we could not find literature on organizational design for automatic IaaS provisioning, the following section will focus on focus on factory-like delivery of IT products.

Terms linked to factory-like software delivery have been used since approximately the '70s, even before the cloud computing paradigm. Around the '90s, the term software factories arose. It describes how product lines can increase productivity and decrease the cycle time of application assembly. Software factories use a type of product line that is automated and supported by metadata, captured by models using domain-specific modeling languages [37, 38]. Currently, consultancy firms often use the term digital factory to describe how a virtual assembly line can drive the DT of an organization. A digital factory uses an automation platform that can facilitate digital supply chain management and unblock transformation potential [39, 40, 41, 42]. A term currently in use in scientific publications is the digital innovation unit (DIU). DIUs are organizational entities designed to break away from conventional structures; they should bridge the gap between business and IT strategies by creating unique interdisciplinary frameworks. These units strive for new approaches that emphasize flexibility, streamlined decision-making, customers and users [43, 44].

Table 1 provides more details of the previous terms and shows their relevance to an organizational design for automatic IaaS provisioning. As table 1 shows, the organizational design for the automatic provisioning of IaaS for MSPs is not fully covered by one of the previous terms or other terms used in existing research.

Concepts	Definition	Goal	How	Relevance	Sources
Software factory	"A product line automated by metadata captured by models using domain specific modelling languages" [37, p.25]	Make application creation more cost-effective by reusing components. This will enable the creation of software engineering supply chains which will enable mass customisation. A software factory should increase productivity and product quality and reduce the cycle time, time to market and costs of a software engineering team.	A Software factory requires knowledge of the creation process of members of a specific product family. This knowledge has to be captured by the software factory after which it will be made available in reusable models, frameworks and smaller patterns implemented in the proper tools. These items can then be used to create a similar item (in the same product family for example) faster.	A software factory describes a modular way of creating software. However, there is a lack of an organisational design or parts of this. Furthermore, there is a lack of automatic software delivery and a self-service portal.	[37, 38]
Digital factory	A virtual assembly line driving the digital transformation of an organisation. A digital factory uses an automation platform that can facilitate digital supply chain management and unblock transformation potential.	The digital factory is focused on the efficient and quick development of digital products. It describes some details on the maintenance of the product. The main purpose of a digital factory is to drive the digital transformation of an organisation through an automation platform.	Methods: Standardization and automation, scaled agile framework, design thinking, DevOps, lean business, continuous testing, and a user-centric approach. Furthermore, the digital factory should work best using cross-functional teams and a center of excellence.	The term digital factory is a confusing term. In the scientific world, the term is used to describe the digitalisation of factories using technologies such as IoT. This means that in the right context, the scientific context of the term is low. The transformation it describes and the method it uses can be relevant for IaaS automation.	[39, 40, 41, 42]
Digital innovation unit (DIU)	"DIUs are organizational units that are used to depart from existing structures. They consolidate business and IT efforts in the form of novel interdisciplinary organizational structures." [43, p.2]	The main goal of DIUs is to create a new work approach in which flexibility, streamlined decision-making, customers and users are central. This new way of working should contribute to the creation of digital products powering the digital transformation.	Innovative product creation, powered by the new way of working in which flexibility, streamlined decision-making, customers and users are central.	There is a lack of the use of automated software delivery. Furthermore, the main focus is on innovation, which can be used in the process of moving to automatic IaaS provisioning. However, this is not the main purpose of this paper.	[43, 44]

Table 1: Existing concepts

2.3 Organizational design criteria

Since there is a lack of literature specifically addressing the organizational design of IaaS providers, it is necessary to draw on broader literature related to organizational design and the processes involved in developing such designs. This section will cover important criteria to consider when creating an organizational design for an MSP using automatic IaaS provisioning and serve as a basis for creating an organizational design framework. First, the STAR framework [11], which can be used to create an organizational design, will be discussed. When creating an organizational design, the STAR model considers strategy, capabilities, structure, processes, rewards, and people. Second, there will be discussed how to prevent silos from forming (in IT organizations). Third, the role of teams in organizations will be addressed. Teams form the building blocks of the structure of an organization. Furthermore, team configurations and interaction modes can help create a well-functioning structure. Lastly, this section covers other factors influencing the organizational design linked to the IaaS business model.

2.3.1 Organizational design

When taking a more detailed look at organizational design, it can be defined as a process which "involves decisions about the configuration of the formal organizational arrangement, including the formal structures, processes, and systems that make up the organization."[45, p.48] In this definition, organizational arrangements are key; they encompass the explicit and relatively consistent elements of the organization. Organizational arrangements can be divided into structures, processes, and systems. The structure of an organization describes formal relationships and associations between groups and individuals. Secondly, processes describe the sequence in which organizational activities are executed. Thirdly, systems that describe the implementations of either social or physical technologies that facilitate the execution of tasks [45]. The goal of the organizational design is "to develop and implement a set of formal organizational arrangements that will, over time, lead to congruence, or a good fit, among all the components of the organization: strategy, work, people, the informal organization, and the formal organizational arrangements."[45, p.48]

The STAR model gives another view on organizational design, it is a framework for creating organizational designs. The STAR model considers strategy, capabilities, structure, processes, rewards, and people when creating an organizational design. The strategy should be aligned with all the other factors in the star model and could be pictured in the middle of the star (the other five factors) [11]. The STAR model overlaps with the description of [45]. Structure and systems are used in both; the other four components of the STAR model can be found in the system, and goals of the organizational design of [45]. The STAR model is not the only model that corresponds with Nadler et al's view on the scope of organizational design [45]. Merron states that an organizational design comprises five components: strategic objectives and orientation, vision, structure, and culture [46]. All components of Merron can be found in the description of Nadler et al.

The STAR model has been refined over thirty years and consists of five clear categories, all linked to strategy [11]. Furthermore, the STAR model's categories overlap with those used in [45]. However, the STAR framework offers a precise categorization of the components, whereas the framework of [45] does not. In addition, the STAR model is among the most popular frameworks for creating an organizational design [47]. Therefore, the STAR framework will be used as the main framework in the organizational design process.

2.3.2 Preventing silos in IT organizations

Conway's law states that there is a "a very close relationship between the structure of a system and the structure of the organization which designed it" [48, p.30]. Conway suggests that the structure of an organization should mirror the IT architecture. However, many organizations form their teams around functional expertise. This creates teams such as testing, architecture, or a dedicated operations team forming silos in an organization [49]. Silos are hindrances in reaching optimal organizational performance and can be described as physical or psychological obstacles that divide individuals, business units, or locations, hindering collaboration among them.[50].

Organizations with silos or silo behavior can never reach their optimal level of performance. Causes of silos linked to organization design are specialized units, no connection between leadership and workforce, organizational culture, a focus on developing skills resulting in a lack of individual relationship building, and a lack of employees easily building relationships with similar people. Organizational structure and culture have a significant role to play. The structure of an organization determines the allocation, management, and alignment of authority and responsibility, as well as the flow of information among different divisions and levels within an organization [51]. Structuring your organization around functional expertise does not allow for a quick and safe flow of change in the organization [49]. An organization's culture encompasses the collective beliefs and values that influence the behavior and functioning of employees within an organization [51].

The formation of silos is not solely a consequence of organizational design; behavioral factors also play a fundamental role in reinforcing silos. It can be defined as the cause of unconscious behavior and mental state, dynamics between teams, and managerial interactions. Silo behavior can result in a lower team identity and can make it difficult for individuals within teams to form relationships [52]. Furthermore, silos impede goal achievement by undermining internal collaboration [53]. In addition to this, silos hinder critical knowledge flows through an organization, with a sub-optimal performance as a result [54].

In IT organizations, linear hierarchical design is a large contributor to the forming of silos. Organizations stick to this linear hierarchical design because technical proficiency arises from specialization. Hence, organizations tend to organize employees into business units that handle similar processes. Expertise and know-how for a particular set of challenges are centralized in one area, allowing the relevant specialists to address them effectively. Dividing employees according to their functions contributes to a steady organization, which is often important to high-level management. Even though these three reasons sound like advantages, they do not stimulate a fast product flow [51]. A very easy example of the above is the split between development and operations in IT organizations.

The effect of silos in IT organizations is often that each silo takes ownership of one part of the software development cycle. The software development cycle consists of "requirements gathering, designing, constructing, testing, maintenance, and support."[51, p.342] When software development teams operate in isolation and only take responsibility for their part of the work, the progress of the work is divided into "chunks" as it moves through the software development cycle. Furthermore, information between teams is not optimal when silos are in place. Information is being contained in silos, hindering teams outside the silos from accessing this information [51]. When teams focus on a single step in the development cycle the end goal, creating value for the customer, is often not the most important to silos. The stream-aligned team, which will be introduced in 2.3.4, has creating value for the customer as its primary goal. They enable a fast product flow and want to add value for the customer [49]. Team topologies model can pose a solution to silo forming.

Management interactions might offer a short-term solution to silos [53]; however, creating a new organizational design is the right long-term solution [11]. The team configurations can be used in this design, as they help in breaking down silos and create a quick and safe flow of change. The team configurations have as a principle that teams directly involved in the flow of change should be able to complete most of their work themselves. This requires them to be cross-functional, containing team members with more than one expertise. If the teams in the flow change need specific expertise and help or want to automate part of their work, they can ask an enabling, complicated subsystem or platform team for this [49].

This shows the importance of an organizational design in breaking down silos. When an organizational design break-downs silos, or even better, does not create them, information flows are more effective, the flow of the software product is faster and more efficient, and the end goal, creating value for the customers, is not forgotten. The team configurations of the team topologies model can be used to prevent silo-like behavior.

The next section discusses the role of teams in the organizations. Furthermore, the team configurations and interaction modes are further explained.

2.3.3 Teams

Team is a word which is used in many different contexts. In the organizational context, a team can be defined as "a small number of people with complementary skills who are committed to a common purpose, set of performance goals, and approach for which they hold themselves accountable."[55, p.41] A group of people does not necessarily have complementary skills, whereas a team does. Therefore, a team's performance is more than the sum of its individuals.

During the 80s and 90s, business environments became more complex, which caused a need for more agility and speed. Teams were used increasingly more to achieve this [56]. Nowadays, teams are widely used in organizations and form the building blocks that create the structure of team-based organizations. Team-based organizations can be organized around customers, products, or processes, with support systems to ensure the teams can perform [57]. In IT organizations teams should be organized around product flow [49]. A team-based organization can be achieved by organizing teams around product flow; e.g. the IaaS product for this thesis. This principle is further explained in the next section, Team topologies.

A team needs to be formed, i.e., going from a group of people to a coherent team. Tuckman's stages of group development describe the phases a group goes through to create a performing team. Tuckman's model contains four phases: forming, storming, norming, and performing. (1) In the forming phase, a group gets together and tries to find the boundaries of tasks and interpersonal behaviors. First relationships between teammates and leaders are forming in this phase. (2) After this, a group enters the storming phase. This phase is characterized by polarization and conflict between teammates, which affects the team's task performance. (3) During the norming phase, any resistance that arose during the storming phase is successfully addressed and resolved. Individuals feel safe enough to discuss personal opinions when working on tasks. (4) In the last phase, performing, the interpersonal structure of the team is used as the activity tool. All issues from previous phases have been fully resolved [58]. This thesis will create an organizational design for teams in the performing phase.

2.3.4 Team topologies

The team topologies model defines four distinct types of team configurations: stream-aligned, enabling, platform, and complicated-subsystem teams [49]. While the original term topology is used to describe the structural and interaction patterns between teams, it can be ambiguous in the context of Infrastructure as a Service (IaaS), where topology typically refers to network or system layout. To avoid confusion, we refer to these team types as team configurations throughout this thesis.

A team configuration describes how a team is organized and works with other teams. Skelton and Pais propose four team configurations and three core interaction models in their book "Team Topologies, organizing business and technology teams for fast flow"[49]. They define a team as "a stable grouping of five to nine people who work toward a shared goal as a unit"[49] A team is proposed to deliver value and is seen as the smallest point of value delivery in an organization. They state a team can perform optimally with a size of 7-9 people. Furthermore, teams should not appear out of nowhere but should be properly designed. The team configurations and interaction models help during the design process [49].

Stream-aligned teams

A workflow can be called a stream when aligned to an organizational capability or business domain. A team can be characterized as a stream-aligned team if it"is aligned to a single, valuable stream of work; this might be a single product or service, a single set of features, a single user journey, or a single user persona"[49]. The team should be able to build and deliver value for both the user and customer without requiring other teams to perform part of their work. The stream-aligned team is the main configuration in an organization, as the other team configurations aim to reduce the workload of stream-aligned teams. A stream-aligned team works on the whole value chain and is closest to the customer, enabling them to react quickly to customer feedback. A stream-aligned team will not consist of specialists, but instead, the team members should be able to cover different areas of expertise to ensure stream-aligned teams have all the capabilities necessary to complete their tasks. Furthermore, stream-aligned teams should be multidisciplinary, DevOps and not hand over the product to other teams [49].

Enabling teams

Stream-aligned teams do not contain specialists and focus on product delivery, so they have no time and knowledge for research, making huge improvements. Enabling teams contain specialists and help the stream-aligned teams overcome problems and make significant improvements. By doing this the enabling teams allow the stream-aligned teams to attain and enhance capabilities without the need to put in the corresponding effort. The main goal of enabling teams is to understand the issues and shortcomings of stream-aligned teams and provide efficient guidance for these problems. An enabling team can have a temporary or long-term relationship with one or more stream-aligned teams [49].

Complicated subsystem teams

A complicated subsystem is defined as a team that "is responsible for building and maintaining a part of the system that depends heavily on specialist knowledge, to the extent that most team members must be specialists in that area of knowledge in order to understand and make changes to the subsystem." [49] The degree of specialist knowledge should be Ph.D. level [59], and the primary purpose of complicates-subsystem teams is to reduce the cognitive load of stream-aligned teams. Complex subsystems reduce a load of stream-aligned teams by "building and maintaining a part of the system that depends heavily on specialist knowledge"[49]. The capabilities necessary to do this are often hard to find and grow. Therefore, these types of specialists are better placed in a separate team. Placing these specialists in a stream-aligned team would not be in line with the goals of a stream-aligned team: feasible and cost-effective. A complex subsystem team is only created when it requires a team of specialists to keep it running or develop it further [49].

Platform teams

The main goal of platform teams is to deliver services that enable the stream-aligned team to be fully autonomous. By providing these services, the cognitive load of the stream-aligned team is reduced. Because of this, the stream-aligned team can maintain full ownership of their application in all phases. Platform teams deliver their services through a platform. A platform is defined as "a foundation of self-service APIs, tools, services, knowledge and support which are arranged as a compelling internal product. Autonomous delivery teams can make use of the platform to deliver product features at a higher pace, with reduced coordination."[49] For ease of adaptation, the platform should be easy to use, and the services should be usable and reliable [49].

Multiple teams can form a configuration together. For example, numerous stream-aligned teams are able to create a platform configuration together. The configurations within the platform are referred to as the inner configurations [49].

In conclusion, the team topologies model proposes four team configurations [49]:

1. **Stream-aligned teams:** Stream-aligned teams serve as the primary team configuration within an organization. They are directly responsible for delivering value and are closely involved in the continuous flow of change. The other team types exist primarily to support stream-aligned teams in their activities.

2. **Enabling teams:** Enabling teams are designed to assist stream-aligned teams in achieving rapid progress by facilitating significant improvements and helping them overcome obstacles that require specialized expertise. To fulfill this role, enabling teams consist of specialists with the necessary knowledge and skills to address domain-specific or technical challenges.

3. **Platform teams:** Platform teams deliver services that enable stream-aligned teams to operate with complete autonomy.

4. **Complicated-subsystem teams:** Complicated-subsystem teams consist of Ph.D.-level specialists responsible for parts of the system that depend on deep, specialized knowledge.

2.3.5 Team interaction modes

Besides four team configurations, the authors of the team topologies model propose three team interaction modes: collaboration, x-as-a-service, and facilitating [49]. Formalizing interaction modes between teams helps assess the effectiveness of software delivery. When these interaction modes have become habits, teams will experience an improved understanding of their objectives, minimized frustration with other teams, and maximum team engagement [49].

Collaboration

The term collaboration is widely used. In this case, the definition does not differ from how the word is usually used: "working closely with another team"[49]. Collaboration is highly suitable when teams need discoveries or maximum flexibility. This interaction mode does not require frequent hand-offs between teams, enabling teams to discover new technologies rapidly. Furthermore, collaboration occurs between teams with different skill sets, resulting in the combined experience and knowledge required to solve a problem. Collaboration has a high cost, and as the cognitive load is high for the teams involved, the result should be valuable. Lastly, a team should only be in collaboration mode with at most one different team [49].

X-as-a-Service

Offering technologies as a service is becoming more and more popular. Especially in the cloud domain, anything is implemented as a service [60]. The team interaction mode x-as-a-service is used when several teams require the same platform, API, or code without putting much effort into it. X-as-a-service is defined as: "consuming or providing something with minimal collaboration"[49]. Using this interaction results in well-defined responsibilities ensuring consistent delivery; however, it relies on effective product management. When a product is fully developed, i.e., focused on predictable delivery instead of further product development, x-as-a-service can be implemented. Running the product as a service requires outstanding work from the team responsible for the service. When this is done right, the team using the service can focus on its value delivery. When implemented correctly, this interaction mode requires minimal collaboration and results in maximum value added for the teams using the service.

Facilitating

The facilitating interaction mode is defined as "helping (or being helped by) another team to clear impediments"[49]. This interaction mode is most effective when the facilitating team can simultaneously help a small number of teams. For enabling teams, the facilitating interaction mode is the main way of working together with other teams. The main goal of a facilitating team is to empower other teams to enhance their effectiveness, accelerate learning, gain a deeper understanding of new technologies, and identify and eliminate shared impediments across the teams. The facilitating team does not directly contribute to the software being built.

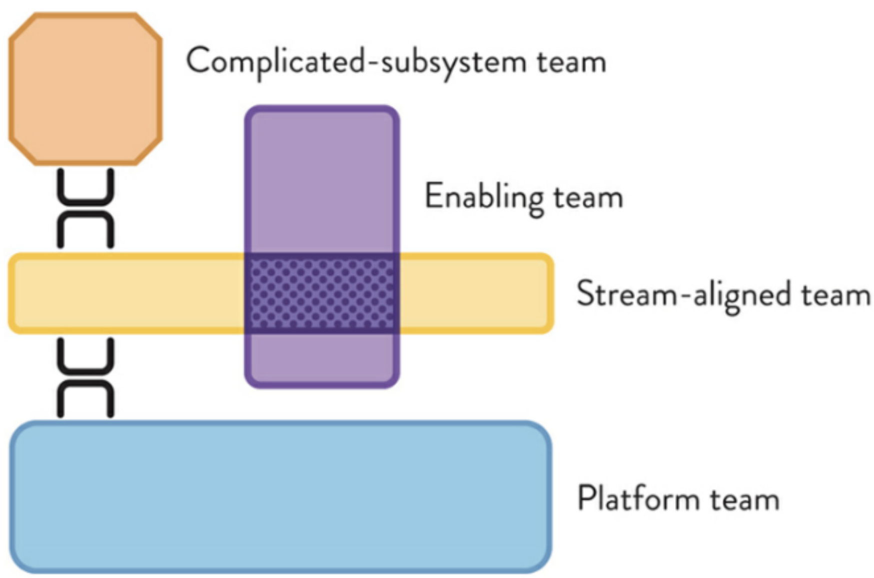

Figure 3: Team configurations and interaction modes

2.3.6 Other factors influencing organizational design

An organizational design is based on how an organization creates, delivers, and captures its value, i.e., a business model. The star model confirms that the business model is at the basis of the organizational design [11]. A business model consists of four basic components: "the value proposition, customers, a financial model and capabilities." [61]

Floerecke and Lehner researched the business model for IaaS extensively [62, 63, 64]. In their recent work [2], they qualitatively evaluate the business model canvas's key determinants, or success factors, for IaaS by scoring the success factors using experts from the field. The most critical success factors will be described below to provide a general business model for the IaaS case. When an IaaS provider successfully implements the key success factors, the provider will be able to differentiate itself from others. The qualitative analysis identifies different sizes of IaaS providers: small, medium, and large. Success factors are scored for the various sizes and all sizes together. The relevant success factors for this thesis have a high score for either medium, large, or both. Below, the success factors are linked to the four components of the business model [62]:

1. Value proposition
   1.1. Integration services (cloud services into existing IT).
   1.2. Hybrid clouds to connect local infrastructure with IaaS.
   1.3. Transition services from on-premises IT to IaaS.
   1.4. High user experience.
   1.5. Extensive customer support.
2. Customer segments
   2.1. Large companies as customers.
3. Customer relationships
   3.1. Referral marketing.
   3.2. Cross-selling in general.
   3.3. Cross-selling with a personal customer approach.
4. Channels
   4.1. Personal customer approach at the sales process.
5. Cost structure
   5.1. Labour costs.
6. Key resources
   6.1. IaaS-specific certificates.
   6.2. Highly qualified employees.

For the components revenue streams, key activities and key partnerships no significant success factors are identified. Therefore, they are not included in the overview.

Value proposition
A successful value proposition for an IaaS provider requires implementing several key factors [2]. First of all, integration services, integrating the IaaS product with the existing (business) process and IT landscape—and second, the use of hybrid cloud services to connect local infrastructure to the IaaS product. Thirdly, supporting services, helping the customer make the switch to a whole cloud-based landscape. Lastly, a high user experience and extensive customer support are success factors when the customer is using the IaaS product [2]. These success factors are relevant for the organizational design as they show the responsibilities of the help desk and other customer-supporting teams.

Customers
In the cloud sector, there is no successful business without customers; this implies that CSPs should focus on one or more customer segment(s). For CSPs offering IaaS large customers are the customer segment leading to the most success. Size is not the only variable when choosing a customer segment, location is influencing success as well. Medium-sized providers should focus on local, regional, and national customers, whereas large providers should focus on national, multinational, and global customers [2]. Before acquiring a new customer, the CSP should reach the customer.

For large-sized CSPs offering IaaS, referral marketing, i.e., word of mouth, is key in gaining new customers [2]. Having satisfied customers is key for referral marketing, as they will serve as a solid referral base, offering access to new customers [65]. To have satisfied customers, properly implementing all success factors discussed in this section is required. When having reached the customers through referral marketing, cross-selling is a success factor in the acquiring process. More specifically, cross-selling using a personal approach is important for IaaS providers [2]. Lastly, when the customer is acquired, a personal customer approach to stimulate customer relationships is a key success factor. For middle-sized providers, providing the customer with a personal approach during the sales process even gets the highest success score of all variables researched [2]. For large-sized providers, this success factor is significant as well [2].

Most of these success factors are relevant to a provider's sales department. To implement the personal approach during and after the sales process, a sales department might use several account managers or customer relationship managers. For an MSP, keeping in touch with the client and offering a personal approach might require less effort, as they are responsible for managing the client's resources even after they are sold.

Financial model

Controlling costs is intrinsically linked to maximizing profits. The most significant costs medium-sized providers must regulate are their labor costs [2]. This is the only substantial success factor linked to the financial model of medium-sized providers. Large-sized providers have no key success factors for their financial model. The research measured costs of hardware, data center rent, software licensing, and energy consumption [2]. Controlling a provider's labor costs means that the organization should not be larger than necessary.

Capabilities

Capabilities at the team level are included in the STAR model. Still, capabilities are essential for the business model as well, as they contribute to the successful and efficient creation of the product. The most critical capability for IaaS providers is having qualified employees, i.e., employees with a high degree of expertise. Consequently, hiring more skilled employees or training the current employee base is key to obtaining success. Furthermore, for medium-sized providers, having IaaS-specific certificates is a critical success factor. Without these, a medium-sized provider will not have a complete set of key resources to be successful [2]. Even though these factors cannot be directly linked to the organizational design, they are essential enablers of the business process.

3 Organizational design framework

Our research focuses on developing an organizational design for managed service providers (MSPs) offering IaaS. As identified in the literature review, there is a lack of existing organizational designs specific to IaaS providers. This highlights the need for a dedicated framework to support the creation of such designs. This chapter presents the phases of organizational design and the key components involved, based on the literature discussed in the previous chapter.

When creating an organizational design, two high-level approaches can be taken: a bottom-up or top-down approach. In the bottom-up approach workflows, processes, measures, and other operational processes are carefully designed, forming the basis of the strategy for the organizational design. The top-down creates the design around a strategy from higher-level management [45]. It is essential that an organization's strategy always influences the organizational design [11], making the bottom-up approach less effective.

A more detailed approach is needed than these high-level approaches. When a deeper analysis is conducted, it can be concluded that organizational design is a complex process where choices made early in the process can have a considerable influence later. Therefore, well-founded decision-making is crucial in the process. A framework can guide this. As stated in the before, the STAR model will be used as the main framework during the organizational design process. However, the STAR model offers five decision categories with no sequence of steps. Therefore, the input of other frameworks will be used to create a complete framework.

The basic principle of the STAR model is that executing different strategies calls for different organizational approaches. A strategy contains crucial capabilities for an organization to reach its strategic goals. An organization's top management is responsible for designing the organization so that the capabilities can be built [11]. At the heart of the STAR model lies alignment with the strategy, i.e., each part of the model should support the organization's strategy. Another fundamental factor is that the strategy should dictate the organizational design, i.e., the STAR model uses a top-down approach. Lastly, the complexity of a business model should be taken into consideration. Simple organizations can not execute complex business models [11]. Figure 4 shows a visualization of the star model. The 6 points in the STAR model are strategy, capabilities, structure, processes, rewards, and people.

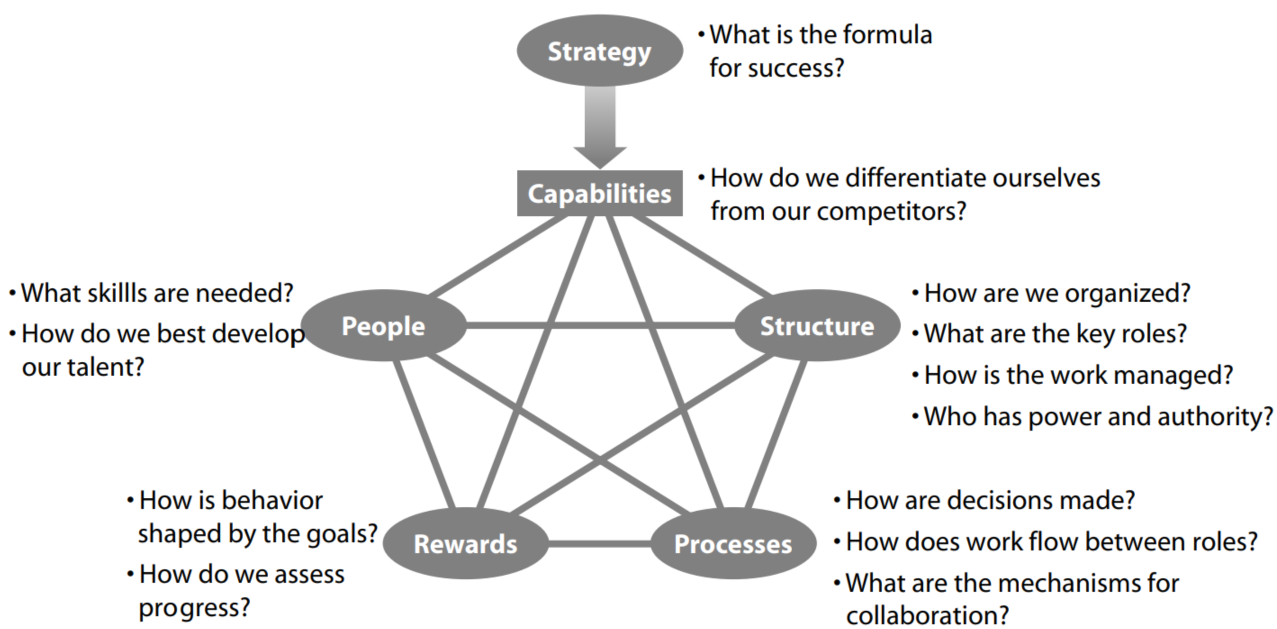

Figure 4: Star model[11, p.3]

3.1 Framework for organizational design

Below, the steps in creating an organizational design of Nadler [45] are linked to the five categories of the STAR model [11]. This will provide a process for understanding the content of the STAR model, which can be used to create an organizational design.

3.1.1 Phase 1 preliminary analysis

The first step in organizational design is assessing the current organizational design, focusing on what could be improved. The outcome of this problem analysis does not necessarily have to be a redesign of the current organization. Other outcomes could be an HR-related problem or inadequate implementation of the strategy from top-level management [45].

3.1.2 Phase 2 Strategic design

Strategy

When the conclusion in Phase 1 is that redesigning the organization is the best improvement option, the strategic design phase can be entered. Creating a strategy for the organization is the first step in both frameworks, as all other components should implement this strategy [11, 45].

A clear definition of a strategy is essential. Consequently, multiple definitions will be discussed. Firstly, in corporate management, strategy is seen as "a process for generating viable directions that lead to satisfactory performance in the marketplace, given a variety of legal constraints and the existence of competitors."[66, p.70] Secondly, the star model defines strategy as "a company's formula for success. It sets the organization's direction and encompasses the company's vision and mission, as well as its short and long-term goals."[11][p.5] Furthermore, the STAR model states that the main purpose of creating and implementing a strategy is to create a competitive advantage [11]. Lastly, Nadler et al. [45] scoped the definition of strategy down for this specific case and focuses on general patterns for the structure of an organization and how to create teams at the top of the organization. Attention should be paid to corporate governance frameworks and processes for top-level decision-making. In other words, it should give the strategic team of the organization clear guidelines for their way of working. Furthermore, the strategy should guide operational design decisions and consider the informal and formal structures and processes [45]. When comparing the view of the STAR model and Nadler et al. [45], the view described by Nadlet et al. can be seen as a more specific strategy for creating an organizational design. Therefore, the definition of Nadler et al. will be used in this thesis.

Structure

The structure needs to be partly accounted for in phase 2, focusing on the top four layers at most [45]. The star model states that the structure of an organization is a main factor in determining how formal power and authority are distributed. It sets out the power distribution, reporting relationships, and communication channels. The teams in an organization form a hierarchical structure for decision-making and management, which can be visualized in an organization chart [11]. This corresponds with the view of Nadler et al. on why the structure should be set in the strategic design phase [45]. Aligning structure and strategy is an important factor for organizational performance, as when they are not aligned, organizations have to work around an obstacle [11]. For IT service providers, in particular, "allocation of roles and responsibilities concerning demand, relationship, performance, and knowledge management is a basic requirement for effective IT service delivery management."[67, p.4573]

There are four structural dimensions: function, product, geography, and customer. Each dimension requires different organizational capabilities. A functional structure is organized around the activities of an organization. Examples of functional silos that are often implemented are human resources, manufacturing, and finance. A functional structure is most efficient for small organizations and some large companies. A product structure is structured around the product lines of an organization. This structure most suits short product life cycles, emphasizing quick product development. A geographic structure is often employed when organizations grow globally. Customer requirements per region/country differ significantly, making a new local branch more efficient. Government regulations can play an essential role in this as well. A customer structure is almost similar to a product structure; however, the organization is structured around customer segments. It is often found in investment banks and professional service organizations. This structure results in a deep understanding of the customer, however, it can only be implemented in large organizations [11].

Teams play a key role in the structure of an organization. More information on teams can be found in 2.3.3

Capabilities
In the STAR model, organizational capabilities are defined as "the unique combination of skills, processes, technologies, and human abilities that differentiate a company." [11, p.7] Capabilities are created within an organization and should be difficult for other organizations to replicate. One of the goals of organizational design is to create superior organizational capabilities, resulting in a competitive advantage. The capabilities of an organization form the basis of a design and can be seen as the design criteria. As a guideline, no more than five design criteria or organizational capabilities should be used in organizational design [11]. Capabilities should be set by top-level management and, therefore, should be accounted for in phase 2.

Capabilities should lead to a better-performing organization. The performance of the delivery service can be specified to two variables, cost efficiency and service quality—the structure and capabilities of an organisation influence service performance. Two capabilities are crucial in leading to better performance for internal IT service departments: performance management and knowledge integration. Performance management ensures that the organization's objectives are tracked efficiently and effectively. Objectives should be reflected in the service level agreements (SLA) and prices. Knowledge integration focuses on utilizing and capitalizing on knowledge critical for delivering the service [67]. Although these capabilities are scoped within internal IT service departments, they are expected to be equally relevant to the delivery of IT services for external customers. This is due to their high-level and broadly applicable nature, which makes them generally transferable across different IT service delivery contexts.

3.1.3 Phase 3 Operational design

When the strategic design of Phase 2 has been finished, the lower-level operational design can be created for Phase 3. When the changes proposed in Phase 2 are small, phase 3 should be relatively straightforward; however, when significant changes in strategy are made, phase 3 becomes complex quickly. Operational design includes reporting relationships, business processes, human resource practices, workflows, and resource allocation. Furthermore, the structure of the lower-level organization should be created in phase 3 [45]. In the STAR model, rewards, people, and processes must be accounted for in phase 3.

Rewards
The main purpose of rewards in an organization is to align employees' behavior with the organization's goals. Furthermore, the company's reward system communicates its values more clearly to employees than any written statement. Rewards that stimulate collaborative behavior are the most difficult to create. Several variables must be considered when designing a reward: the level, the locus of measure, the wanted behavior, and the evaluation process [11].

People
In the star model, the people component focuses on human resource policies, including those for selection, staffing, training, and development. These policies should be established to shape the capabilities and mindsets essential for executing the organization's strategy. Furthermore, the model states that not only management should have the social skills necessary to overarch business units and work together in a team, but employees throughout the whole organisation should have this [11].

Processes
Processes are defined as "a series of connected activities that move information up and down and across the organization."[11, p.17] Processes are implemented at every level of an organization and are used, for example, for order fulfillment, closing a deal, or managing a portfolio. The way processes are designed has a significant impact on the way organizational units work together. As the structure of an organization often results in silos, processes can help in breaking down these silos. Processes are not the only way to break down silos; lateral connections in an organization can help in breaking down silos as well. There are several levels in lateral connections: networks, teams, integrative roles, and matrix, with each level having a stronger connection [11].

The first level, networks, is the basis for other lateral connections. A network contains the informal connection of an employee throughout the organization. At the second level, teams are described as " cross-business structures that bring people together to work interdependently and share collective responsibility for outcomes"[11, p.18]. Where teams are often located within one business unit, integrative roles are implemented in overarching business units. For example, a manager tasked with coordinating work across several business units. This requires more advanced coordination than teams. Lastly, a matrix is defined as "a set of dual reporting relationships used to balance two or more dimensions in an organization."[11, p.19] The other lateral levels serve as a basis for a matrix. In section 2.3.2 silo prevention will be further discussed.

3.1.4 Phase 4 Implementation

Often, organizations make the mistake of stopping at the end of phase 3. Everything has been designed. However, the hardest step still has to be taken—implementing the new design. The organization should enter a state of transition to achieve the desired future state. Obstacles to this state can be political power plays, concerns of the workforce, and the perception that management is losing control due to changes in traditional work methods. Successful implementation involves precise planning, continuous monitoring, and ongoing management participation.

4 Methodology

In order to answer the main research question of this thesis: “What are organizational design patterns for a managed service provider that implemented automatic IaaS provisioning?” an exploratory multiple case study is conducted. As presented in the literature review, the current knowledge on organizational design patterns for IaaS is very limited. This highlights the need to study existing organizational designs in practice. To address this, the thesis conducts a multiple case study, involving a total of 14 case organizations. In total, 14 interviews were conducted, totaling to 1,205 minutes of recorded conversation and generating 1,123 pages of transcripts. The interviews were carried out between July 24, 2024, and October 22, 2024, and the findings will be presented in the results chapter.

This chapter further explains why this approach is suitable to answer the research question. Furthermore, this chapter elaborates on the case selection strategy, the data collection and the analysis.

4.1 Research method: multiple case study

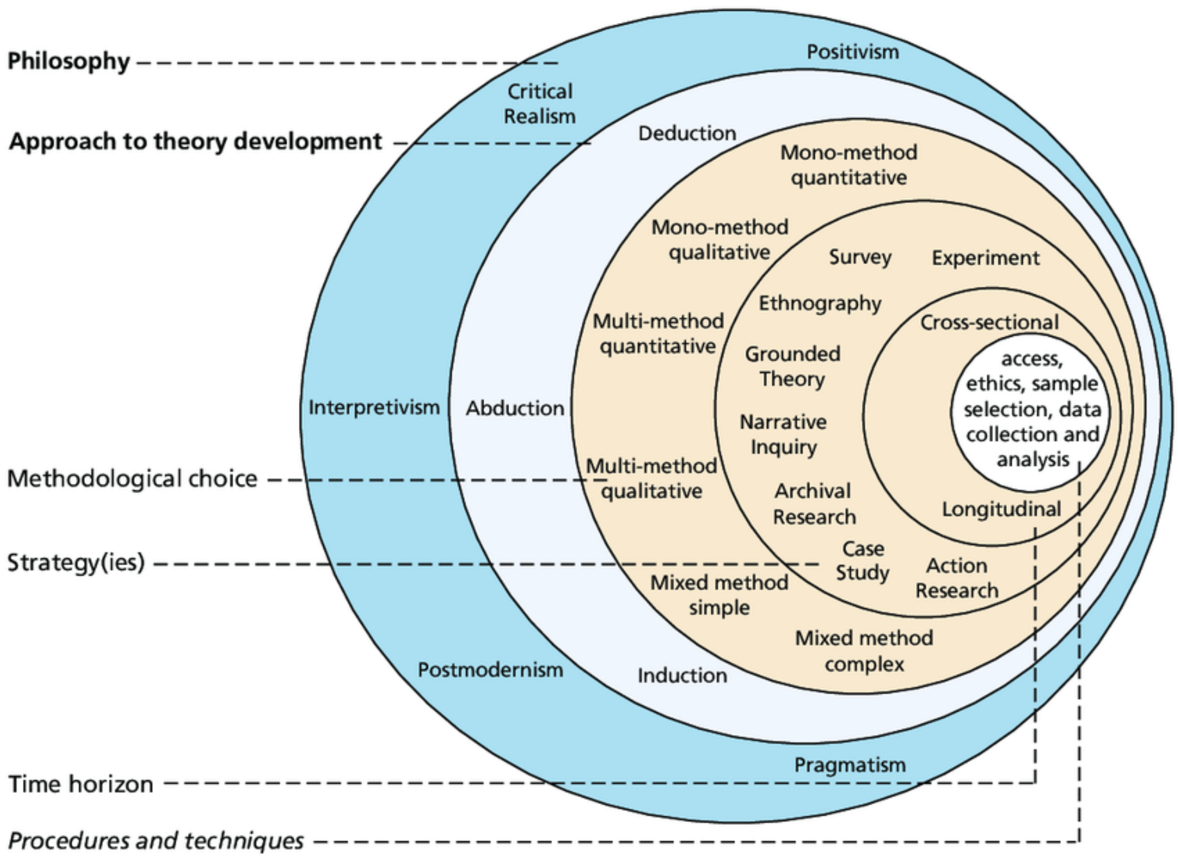

Figure 5: Research onion [68, p.131]

The research method is chosen through using the research onion. The research onion is a layered model that can be used to formulate a methodology for business studies [68]. This section will go through each layer to conclude the most suitable method.

1. Philosophy

The research philosophy describes the basic research technique. From a historical point of view, two streams can be identified: positivism and interpretivism. Positivism emphasizes the objective, measurable observations and it supports the use of quantitative methods, while interpretivism focuses on understanding subjective meanings and social contexts through qualitative methods [68]. In addition to these two historical streams, three more recent streams can be added: pragmatism, critical realism and postmodernism. Pragmatism focuses on the assumption that a researcher can flexibly adopt either a positivist or interprevist position depending on which is most suitable for the specific research question [69]. Critical realism states that an objective reality exists independently of human perception and that understanding this reality is inherently influenced by social, cultural, and historical factors [68]. The last philosphy, postmodernism, challenges a single objective reality emphasizing that knowledge is socially constructed, shaped by power relations and language and context-dependent [68].

For this research, the interpretivism philosophy is most suitable. This stream focuses on new understandings and states that the researcher and his interpretations are part of what is being researched. Focusing on new understandings is essential when researching new patterns for IaaS providers.

2. Approach to theory development

There are three approaches to theory development: deduction, abduction, and induction. Firstly, deduction focuses on using existing theories to form a hypothesis and perform data collection to test the hypothesis. Secondly, induction focuses on starting with observations from data collection, after which the data will be analyzed to form a theory. Lastly, abduction focuses on empirical data followed by research concluding a conclusion based on the evidence [69].

Due to the lack of existing theories and literature deduction is not suitable. Furthermore, studying large IT infrastructure providers in an empirical way is not possible in the scope of this thesis. Therefore, an inductive approach is most suitable.

3. Methodological choice

The methodological choice layer in the onion proposes multiple research approaches. There are two major research approaches: quantitative and qualitative research. Quantitative research explains phenomena using numerical data, which are analyzed using mathematical methods; quantitative research deducts conclusions from data that can not be statistically analyzed [70].

This research focuses on IT organizations; when studying organizations, the qualitative method is frequently used [71]. For this research, a qualitative method is most suitable as measuring organizational patterns quantitative is difficult due to the nature of organizational designs. Because of this, a mono-method qualitative approach will be used in this thesis.

4. Strategies

This layer in the onion proposes specific research strategies that can be used to answer the research question. The main goal of this research is to study how managed service providers (MSPs) offering IaaS are organized by understanding what is happening in the field of MSPs. This implicates an exploratory research method [72]. Qualitative exploratory research is possible using several methods. It is important to pick the proper method for your research and avoid a mismatch. Yin states that five primary research methods can be used: case studies, history, archival analysis, surveys, and experiments [73]. The research onion adds ethnography, grounded theory, narrative inquiry and action research to the proposed methods by Yin. To keep the scope manageable, only Yin's five primary research methods will be discussed.

Given that IT infrastructure [24] and IT organizations [19] have been changing over time, a research method focusing on history is not suitable. Furthermore, there is not much literature on the organizational design of MSPs offering IaaS, so an archival analysis is impossible. In addition, control of behavioral events is impossible due to the size of MSPs, making experiments impossible. This leaves a survey or case study. A case study is appropriate when there is a lack of existing literature [74] and matches the goal of this research, as case studies are used to develop theories [75]. When choosing a case study as a research method, there are two options: a single case study focusing on a single case or a multiple case study focusing on the similarities and differences between cases [75]. The main research questions focuses on patterns across multiple MSPs, which implcates a multiple case study [1].

5. Time horizon

The research onion states that there are two times frames to perform research in. A cross-sectional study in which the research focuses on a specific point of time (short term) or a longitudinal study examining data over an extended period to perform a comparative analysis [69]. This thesis focuses on finding patterns on organizational design at a specific point in time, making this a cross-sectional

[1] Due to data & time limitations a single interview was conducted per case

study.

The procedures and techniques used will be further explained in 4.4 and 4.3.

Validity

For case studies, as for all social science research methods, there are four validity tests: construct validity, internal validity, external validity, and reliability [73]. Internal validity only applies to explanatory or causal studies [73]. As this study is exploratory, internal validity is not applicable.

Construct validity: can be defined as "identifying correct operational measures for the concepts being studied"[73, p. 40]. Three often-used case study tactics are used to raise construct validity, two of which are implemented in this research setup. First, multiple sources of evidence were used (interviews + documents). Secondly, the report per case is reviewed by the interviewee. There is no formal chain of evidence, as an external observer will not observe the chain due to the project's scope and the supervisors' time management.

External validity: can be defined as "defining the domain to which a study's findings can be generalized"[73, p.40]. As this case study uses clear case selection criteria, replication logic can be used as a tactic [73] to increase external validity. The domain to which the findings can be generalized is the domain to which the case selection criteria apply.

Reliability: can be defined as: "demonstrating that the operations of a study – such as the data collection procedures – can be repeated, with the same results"[73, p.40]. Tactics that can be used to increase reliability are using a case study protocol and a case study database. The case study protocol is further described in the methodology. Furthermore, a separate database was used for the raw data (interview recordings in the Microsoft Teams environment of Leiden University) and the case reports (Overleaf).

4.2 Case selection strategy

The case selection strategy for our multiple case study follows the replication logic proposed by Yin [73]. When selecting cases for our multiple case study, the cases should "either (a) predict similar results (a literal replication) or (b) predict contrasting results but for anticipated reasons (a theoretical replication)." [73, p.54]. The replication logic enhances the reliability and generalization of multiple case study findings by systematically comparing cases for consistent or contrasting results [73].

For our research, this means that to validate the patterns of a managed service provider (MSP) that actively manages infrastructure services, contrasting cases are needed, which produce an anticipated different result (theoretical replications). The contrasting cases in our research do not actively manage the infrastructure service they offer. The findings in these should support the conclusions of the MSP cases. Furthermore, the theoretical replications should be used to confirm the new theoretical insights found in this exploratory study, enhancing the quality of the new theory [73]. The theoretical replications should not differ in other aspects

This implicates MoSCoW selection criteria given below:

- Must have:
  - Scale of operations: The organization must host 600 or more applications. This threshold is chosen to be able to compare organizations to the host organization (SCC-ICT);
  - Agile and DevOps adoption: The organization must have implemented an Agile or DevOps way of working;
- Should have:
  - Managed services capability: Organizations should deliver managed infrastructure services. This is the differentiation factor leading to literal and theoretical replications and therefore a should have.
- Could have:

– Sector similarity: organizations operating in the public sector, similar to the host organization, are given preference;
– Business to Business (B2B) orientation: organization that operate B2B are given preference;
– Scaled Agile Framework (SAFe) or Large Scale Scrum (LeSS): organizations using SAFe or LeSS are considered valuable sources of insight;
– Hybrid application landscape: Organizations with a hybric application landscape are of particular interest. Nevertheless, companies with purely modern application environments may also provide innovative practices in organizational design.

In total, approximately 30 companies were contacted, all within the Netherlands. Before contacting the cases, as much online research as possible was done on whether the selection criteria were met. Furthermore, before an interview was held, there was a 15 to 30-minute introduction meeting with most interviewees to explain the goal of the interview and see whether an organization met the case selection criteria. In total, 16 interviews were held, of which 14 companies met all selection criteria. The cases are named using the alphabet.

4.3 Data collection: interviews and artifacts

The interviews should lead to a dataset from which exploratory insights can be concluded. Since the research is about organizational design, few exact variables can be measured. Furthermore, the researcher has deepened his knowledge of organizational design, but at the start of the interviews, he could still be considered a beginner on this subject. Therefore, a semi-structured interview in which themes can be covered in a conversational style where there is room for the interviewee to provide information not anticipated by the researcher seemed most suitable. Furthermore, a semi-structured interview is best when wanting to know the reason behind choices [76], which is essential to understanding why organizations are organized the way they are.

The first step in creating the interview guide was determining which categories would be used. For this, the literature presented in 2.3 was critical. First, the categories of the STAR framework were used: strategy, capabilities, structure, processes, rewards, and people. Questions were created for each category. After careful consideration, it was concluded that obtaining precise and reliable information regarding rewards would be challenging. It is uncertain whether interviewees would be willing to disclose their salaries or additional benefits, let alone those of their colleagues. This makes it difficult to form an accurate understanding of the reward structure in the case of organizations. Therefore, rewards have been excluded from the interview guide and the remainder of the thesis.

Furthermore, the People dimension of the Star framework includes recruitment and selection, training, performance management, and organizational culture [11]. To ensure the research remains focused and manageable, the elements related to recruitment and selection, training, and performance management are excluded from this thesis. These components primarily pertain to the execution of human resource strategy, which falls outside the scope of this study. Moreover, the capability dimension already captures the essential competencies required within the organization, which sufficiently addresses the relevant aspects of this research. In contrast, the culture component is retained as a minor part of the interview guide, as it offers valuable insights into the broader organizational context. Additionally, security is essential in IaaS [77] and has an influential impact on organizational design. Therefore, security is added as a category in the interview guide. The following categories were used in the interview:

1. Structure
2. Strategy
3. Capabilities
4. Processes
5. Culture
6. Security

The literature review also presents an organizational design framework with 4 phases: preliminary analysis, strategic design, operational design, and implementation. As this thesis focuses on measuring organizational design at a specific point in time at the case companies, the phases are not applicable during the interview phase. However, in the interview, strategy, structure, and capabilities (strategic design, phase 2) are handled before processes, culture, and security (operational design, phase 3).

The final version of the interview guide is included in the Appendix . During the interview, the interviewee is asked to create or deliver a visualization of the organizational structure. This visualization is added as an artifact to the data. The questions in the interview guide are used to determine the role of the teams in the structure so that a proper analysis can be performed later. Other categories consist of solely questions.

The interview guide was shared after the introduction meeting and before the interview, the interviewees often requested it. After a few interviews, sending the questions beforehand became standard, as it was asked in every introduction meeting. During the interviews, it became apparent that people could speak freely, and interviewees could be considered informants or experts [73]. To further avoid socially desirable answers, interviews were scheduled with a single interviewee. However, in three interviews, the interviewee had invited a colleague. In two of these interviews, the interviewees worked closely together for multiple years, and the researcher did not get the impression socially desirable answers were given. In the third interview, a second interviewee joined for part of the interview (approximately 30 min), which gave the researcher multiple views. The fact that the second interviewee was not present the whole time ensured a minimization of socially desirable answers.

In total, the 14 interviews took 1205 minutes, and 1123 pages of transcripts were obtained. The interviews were held from 24 July 2024 to 22 October 2024. Interviews took 60 to 120 minutes and were video and audio recorded. The duration varies due to the interviewee's availability. Furthermore, a summary of the interview was sent to the interviewee for feedback.

4.4 Data analysis: visual mapping

After the interviews, a dataset consisting of transcription and organization visualization was available. It was decided that the transcriptions would not be coded, as the high-level answers could be deduced from the transcripts. Coding the transcripts would only offer a small extra amount of insight as the interviewees answered the questions clearly. In addition, labeling does not match the exploratory nature of this research, where the interest is in high-level findings and not in small details. Instead of labeling the transcriptions, the interviews were all replayed, and transcriptions were read. While doing this, the information was used to create an overview per case in which information on the six categories of the interviews is presented. Afterward, another round was conducted to determine whether findings in one case were present in another case. Creating this overview was sufficient to find matching patterns across cases.

The visualization created by the interviewee and the information on the role per team were used to create a new structure based on the team configurations. This was done using visual mapping. Visual mapping utilizes graphical forms enabling the presentation of large amounts of information in a compact space and serves as an effective tool for developing and validating ideas in theory development. Visual mapping can also be used to compare and synthesize multiple visuals to create a broader theory [78]. This makes it ideal for creating patterns in which the structure visualization will be essential.

The researcher was already familiar with the team configurations used in the visualizations before the interviews, so he knew when he had enough information per team to determine the configuration. When the new structure based on the configurations was created, the interviewee checked whether the right configurations were chosen.

The overviews per case are analyzed and used to create a model presented in the discussion to answer the three research questions. The model consists of patterns, team configurations within these patterns, and an overview of critical factors per pattern. A pattern is a high-level description of a group of cases with similar characteristics. The team configurations use the team topologies model but focus on the high-level configurations, and the critical factors can be used to determine which pattern is suitable for an organization. When the overview per case was finished, it became apparent that groups could be formed based on the organizational structure. Cases with similar structures form the basis for a pattern. Furthermore, the frameworks of cases were compared to see in which pattern they would fit. Other high-level similarities and differences are clearly explained in the Appendix .

5 Results

The results of the multiple case study described in the previous chapter are presented in this chapter, with a detailed case report provided for each organization. A total of 16 interviews were conducted at 16 organizations. In one of the interviews, it became apparent that the organization did not meet the case selection criteria. In another interview, not enough information was obtained to include the case in the research. Therefore, 14 cases will be presented in the results section. In total, the 14 interviews took 1205 minutes, and 1123 pages of transcripts were obtained. The interviews were held from 24 July 2024 to 22 October 2024.

5.1 Overview of collected cases

Table 2 shows an overview of the collected cases:

Table 2: Description of collected cases

Case	Sector	Organization size	Number of applications	Customer	Type of offering	Public or private services	Role of interviewee + (experience in role)
A	Utility \ Energy	Small	500-600	Internal	Standard	Public	Lead Architect platform infrastructure (7y)
B	IT services	Small	1000	Internal	Standard	Both	Services director (7y)
C	Utility \ Energy	Medium	1100	Internal	Standard	Both	Release train engineer & Product manager (3.5y)
D	Government	Medium	-	Commercial	Standard	Private	Account manager (3y)
E	Government	Medium	-	Internal	Standard	Private	Solution Architect (3y)
F	Government	Large	-	Internal	Standard	Private	Advisor MT DataCenter Services (5y)
G	Financial services	Large	10.000+	Commercial	Standard	Both	Chapter lead (4y)
H	IT services	Large	-	Commercial	Standard	Both	Enterprise Architect (14y)
I	Financial services	Large	-	Internal	Custom	Both	Solution Architect (13y)
J	Government	Medium	1100	Internal	Custom	Private	Release train engineer & Technical Architect (15y)
K	Manufacturing	Large	2500	Internal	Custom	Private	Domain Architect (1y)
L	IT services	Medium	600+ (50 large clients)	Commercial	Custom	Private	Lead Architect (8y)
M	IT services	Large	-	Commercial	Custom	Both	VP Services (2.5y)
N	IT services	Large	-	Commercial	Custom	Both	Relation manager & Sales manager (7y)

In Table 2 cases are presented using the following variables:

- Sector: the sector in which the organization of the infrastructure department is active;

- Organization size: the size of the infrastructure department. It can be either small (<50 employees or <5 teams), medium (50-150 employees or 5-15 teams), or large (>150 employees or >15 teams). Size is solely based on the number of employees as the difference between workload for public cloud and on-premises is too big. This is because an on-premises environment needs people to care for the housing (the data center), which is not the case for a public cloud platform.
- Number of applications: the number of applications the infrastructure department hosts;
- Customer: the customer consuming IaaS. It can be either internal (within the same company) or commercial (IaaS is provided as a commercial activity to an external client);
- Type of offering: The level of uniformity or customization in IaaS. Companies that offer standard IaaS focus on a uniform, one-size-fits-all service with limited customization designed to suit general application needs. In contrast, companies that provide custom IaaS offer flexible configurations and tailored solutions to meet specific application requirements, reflecting low standardization.
- Public or private services: whether the infrastructure services are provisioned using public services (from external vendors), private (on-premises data center) services, or both.
- The role of the interviewee: the role of the interviewee at the current organization and the number of years of experience in this role. The entries with two interviewees had a single interview with two interviewees simultaneously.

5.2 Per case analysis

The rest of this chapter is organized into two distinct sections. First, a short description per category, including an overview of that category, is given, and subsequently, the results per case are provided.

5.2.1 Introduction

The following categories are used to analyze the cases: structure, strategy, processes, capabilities, culture, and security. This is further explained in the literature review and methodology. This subsection presents a description per category, including an overview of that category.

Organizational & team structure:
A visualization using the team configurations is given per case. Following the team topologies model, there are four team configurations [49]:

1. **Stream-aligned teams:** Stream-aligned teams serve as the primary team configuration within an organization. They are directly responsible for delivering value and are closely involved in the continuous flow of change. The other team types exist primarily to support stream-aligned teams in their activities.

2. **Enabling teams:** Enabling teams are designed to assist stream-aligned teams in achieving rapid progress by facilitating significant improvements and helping them overcome obstacles that require specialized expertise. To fulfill this role, enabling teams consist of specialists with the necessary knowledge and skills to address domain-specific or technical challenges.

3. **Platform teams:** Platform teams deliver services that enable stream-aligned teams to operate with complete autonomy.

4. **Complicated-subsystem teams:** Complicated-subsystem teams consist of Ph.D.-level specialists responsible for parts of the system that depend on deep, specialized knowledge.

More information on the configurations can be found in 2.3.4. It is important to note that new configurations were identified in the interviews: the technology-component configuration and the process configuration. These new configurations are sub-configurations of the stream-aligned configuration. When a team is not one of these sub-configurations, a team will be visualized as a regular stream-aligned team. The definition of these new configurations is further explained in the discussion section 6.5. The new configurations are already presented in the results section to present a complete overview.

The visualization of the structure per case is layered: teams offer services to the teams directly above, below, behind, or in front. Furthermore, the actual names of the teams are replaced with descriptive names to guarantee anonymity. In addition, the team size in the visualization is not linked to the real team size. Lastly, the visualization per case uses the following color codes:

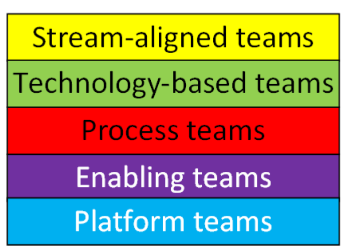

Figure 6: Team configurations color codes

Table 4 presents the team configurations identified in each case. Cells highlighted in green indicate that the organization applies the corresponding configuration.

Table 4: Team configurations per case

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
Stream-aligned	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓			
Stream-aligned Technology-component			✓	✓		✓	✓	✓	✓	✓	✓		✓	✓
Stream-aligned Process											✓	✓	✓	✓
Enablement	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Platform	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓		
Complicated subsystem														

Frameworks:
A short description of the applied frameworks is given per case. Cases used either SAFe, Scrum, ITIL, DevOps, worked project-based, or used a combination of frameworks. Furthermore, some organizations work with an operations lead of the day: an operations engineer responsible for handling the operational tasks of the day. This role can rotate among team members. In addition to this, interviewees were asked to rank their organization on the COBIT maturity framework, which provides the following maturity levels [79]:

- (1) Ad hoc: Everything happens ad hoc, there is no standardization at all.
- (2) Repetitive but intuitive: Repetitive tasks happen the right way, but there is no documentation at all.
- (3) Defined: Processes and responsibilities are defined and documented.
- (4) Managed and measurable: Processes are not only defined, but include version control and have specific checkpoints to verify that execution is correct. Risks to the business are identified, documented and potentially mitigated. Vendor strategy and lifecycle management strategies are used.
- (5) Optimized: Alerts are in place if the process is bypassed and management programs are in place to continuously improve risks, risk mitigations and procedures.

Table 5 shows an overview of the frameworks and COBIT maturity per case. Case G uses the Spotify model instead of the SAFe model.

Table 5: Applied frameworks per case

Frameworks	A	B	C	D	E	F	G	H	I	J	K	L	M	N
Maturity rating by interviewee	3	5	4		3	3	4	5	4	2	4	3		
SAFe	✓		✓		✓	✓		✓			✓			
Spotify							✓							
Scrum	✓		✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ITIL	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
DevOps	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓				
Ops lead of the day	✓					✓	✓		✓	✓				
Project-based		✓						✓			✓	✓	✓	✓

Culture:
A short description of the culture is given per case. If the infrastructure department's culture differs from that of the rest of the organization, it will be included. Furthermore, interviewees were asked to give a classification from 1-5 for agile attitudes, where 1: attitude is not visible at all, and 5: attitude is visible across the whole infrastructure department. The following attitudes were used:

- Attitude toward learning spirit - "is the degree to which an actor evaluates openness and searches positively for new things regarding their work in a digitally driven volatile, uncertain, complex and ambiguous (VUCA) environment"[80, p.10].

- Attitude toward collaborative exchange - "the extent to which an actor positively values transparent work and sharing knowledge to solve problems by swapping information and ideas with colleagues"[80, p.10].

- Attitude toward empowered self-guidance - "the extent to which actors positively value reflection on themselves and their work processes, organize themselves, and take responsibility for their work"[80, p.10].

- Attitude toward customer co-creation - "the extent to which an actor in a digitally driven VUCA environment positively values being continuously oriented towards value for the customer and staying in direct contact with them"[80, p.11].

Table 6 presents an overview of the classifications for the agile attitudes per case. Case F has two classifications, the first for the agile part of the organization and the second for the non-agile part. The agile attitudes of case N could not be included as there was no time to address them in the interview.

Table 6: Agile attitudes per case

Agile attitudes	A	B	C	D	E	F	G	H	I	J	K	L	M	N
Learning-spirit	4	5	3.5	4	4	2/4	5	5	4	3.5	5	3	4	
Collaborative exchange	3	5	5	5	4	4/4	4	5	2	3	5	5	5	
Empowered self-guidance	4.5	5	3.5	3.5	3	3/4	4	5	3	3	4	4	3.5	
Customer co-creation	3	5	1	3	5	4/2	2	5	4	2	4	4	4	

Security:
A short description of how security is organized is given per case. Interviewees were asked what could be shared about the security organization, which led to high-level answers. Two patterns could be identified. Firstly, the security team(s) offer consulting services to the rest of the organization, focusing on creating a more secure product or implementing the latest regulations and guidelines. Secondly, the development teams were ultimately responsible for creating a security product in all cases. Lastly, in one of the cases, there is a separate security team focusing on audits. Note that no example security patterns were presented in the interview.

Table 7 shows an overview of the security organization per case.

Table 7: Security processes per case

Security	A	B	C	D	E	F	G	H	I	J	K	L	M	N
Consulting	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Responsible at team-level	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	
Audits		✓												

Capabilities:
A short description of the capabilities used is given per case. No example capabilities were presented in the interview, which is why the interviewees' answers varied largely. Furthermore, if a cell is not highlighted in green, this does not necessarily indicate the absence of the corresponding capability in the case organization; rather, it reflects that the interviewee did not explicitly mention it. Table 8 shows an overview of the capabilities per case. The capabilities of case M could not be included due to time constraints in the interview.

Table 8: Capabilities per case

Capabilities	A	B	C	D	E	F	G	H	I	J	K	L	M	N
Advanced technical skills	X	X		X		X	X	X			X	X		
Soft skills					X	X		X			X	X		
Autonomous	X	X	X											
DevOps	X													
Passion for infrastructure			X											
Distinction between a development and operations profile				X	X									X
Mutli-domain expertise					X				X					
SAFe/Scrum										X				
Working in a changing environment						X								X
Automation skills						X								
Diamond-shaped capability profile per team							X							

5.2.2 Case A

Introduction: Case A is a large organization with approximately 800 IT employees working on the application side. However, the infrastructure for these teams is provisioned through a single platform team. This platform team offers two types of public infrastructure platforms. Case A focuses on a strong standardized approach, where each application team is responsible for its infrastructure.
Role of interviewee: Lead architect public infrastructure cloud platform
Organization size: Small
Customers: Internal
Sector: Utilities
Strategy: (1) Modern technology landscape, (2) Build for change, (3) Innovate, (4) Great place to work and (5) collaborate within the sector.
Maturity: Level 3
Structure:

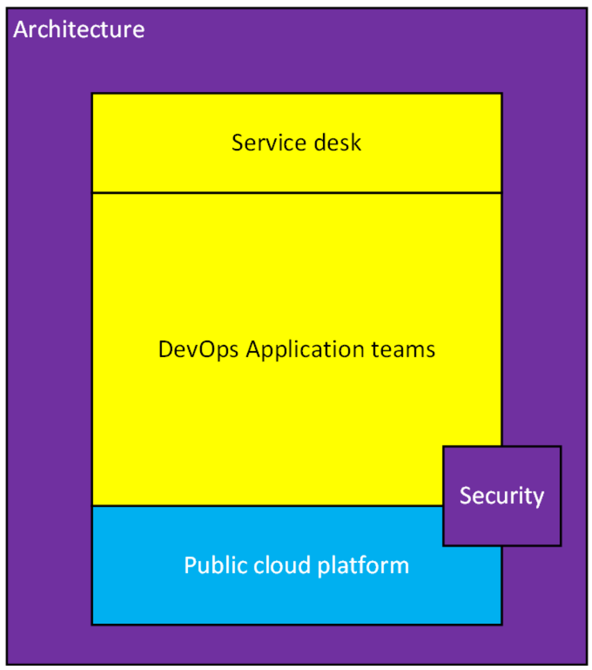

Figure 7: Structure case A

Within the IT department of case A, there are two types of teams: application teams and an infrastructure platform team. The application teams deliver applications to the business, while the infrastructure platform team offers the infrastructure services that enable these applications. The application teams are stream-aligned as they work on one clear stream and are the main type of team adding value. The infrastructure team is responsible for offering the application teams two types of infrastructure platforms.

There is a service desk where business users can reach the IT department. This is a skilled service desk where approximately 80% of issues are solved by the service desk. The other 20% is directed to the team responsible for the part of the system causing the problem.

Furthermore, architecture is the enabling team overarching the IT department. Lastly, a security team enables both the infrastructure and the application teams.

Processes: The IT department of case A operates using the SAFe framework. Both the infrastructure and application teams work using Scrum and DevOps. The infrastructure platform team is responsible for offering two public cloud platforms and has the capability to add more platforms. The main responsibilities of the infrastructure platform team include:

- offering infrastructure services to the application teams

- maintaining and updating the platforms to keep the platform secure and up-to-date with the latest trends in the market and the requests of the application teams.
- providing a pipeline that the applications can use to request infrastructure services. This pipeline should add security measures to the request to make the provided service secure.
- help the application teams with requesting services from the right public cloud platform.

The platform team has a dedicated person who is responsible for the operation of that day (the go-to guy); this role is being rotated among team members. This means that every team member should be capable of executing development and operational tasks. The go-to guy is monitoring issues actively and can be reached via Slack by application teams to solve issues. Operational tickets are handled using ITIL.

The application teams that consume the infrastructure services work using a DevOps approach, i.e., the team is responsible for an application's development and operational aspects. The operational aspects include the infrastructure on which the application runs. Within an application team, one or more infrastructure specialists are responsible for requesting and running the infrastructure. The platform team is solely responsible for the operation on the platform level, i.e., keeping the platform running, but not for the infrastructure services provided. In other words, the application teams and platform team share the responsibility of running the infrastructure and are each responsible for different aspects of the infrastructure.

The platform team has created a golden path to help the application teams with their infrastructure operation. When the application teams follow the golden path while requesting the infrastructure, most operational tasks are automated. For example, when the golden path is used, patching is automated. When an application team chooses to deviate from the golden path, the added operational tasks are their own responsibility. Furthermore, an application team can outsource its infrastructure tasks to an external partner. Another option is to establish an infrastructure team within an agile release train (ART). ARTs are structured groupings of agile teams that collaborate toward a shared goal or solution, enabling coordinated and continuous value delivery. The infrastructure team within an ART at case A can become responsible for requesting and running all infrastructure within the ART. Application teams (or ARTs) should determine whether they would like to use one of the above options.

Using the platform-based way of working where application teams are responsible for requesting and running their own infrastructure minimizes the number of handoffs. This is because of the fact that requests are submitted and then executed automatically.

Capabilities: Teams within the IT department should operate as autonomously as possible. This has as a consequence that within a team engineers should have both development and operational experience. Furthermore, as an application team is responsible for the infrastructure of their application they should have an infrastructure specialist within their team. Everyone in the IT department gets an obligatory training which includes the basis aspects of infrastructure.

To let the platform team function autonomously highly experienced (senior) engineers are needed that are able to work on both development and operational tasks on two public platforms. The fact that the platform team wants to be able to implement platforms from more vendors contributes to the fact that a high degree of experience is needed.

All teams in the IT department follow Scrum and SAFe methodologies. These frameworks assign distinct responsibilities to different managerial levels across the areas of people, processes, and technology. As a result, no single manager is responsible for all three domains. Case A has dedicated managers for people, process and technology on all levels in the IT department. Even though this transformation was made several years ago, the traditional managers ,handling people process and technology, are sometimes still struggling in their new role.

Culture: Case A has an open culture where employees experience a lot of freedom. This has the consequence that employees get the opportunity to try new solutions without being punished when something goes wrong. Furthermore, employees get the opportunity to participate in training to learn new things.

Agile culture
Attitude towards learning spirit: 4

Attitude toward collaborative exchange: 3
Attitude towards empowered self-guidance: 4.5
Attitude towards customer co-creation: 3

Security: Each team is responsible for securing their product (infrastructure/application). This means that engineers should not only be DevOps specialists, but they should be able to integrate security into their products as well. When needed, they can use of consultancy services from the security team. Furthermore, there is a SOC monitoring the security across the IT department.

5.2.3 Case B

Introduction: Case B is a small organization operating as an IT service provider offering both on-premises and public cloud. Furthermore, case B does not use a large-scale operating model, as all infrastructure teams are clustered in a single platform. Each team is free to choose its way of working, which is possible because of mature, self-managing teams.
Role of interviewee: Service director
Organization size: Small
Customers: Internal, final customer consuming application services is external.
Sector: IT services
Strategy: (1) Create best of breed products which are easy to integrate. (2) When a component is used across multiple teams, services will integrate the components in their portfolio and offer it to the whole organization. (3) When necessary create own tooling.
Maturity: Level 5
Structure:

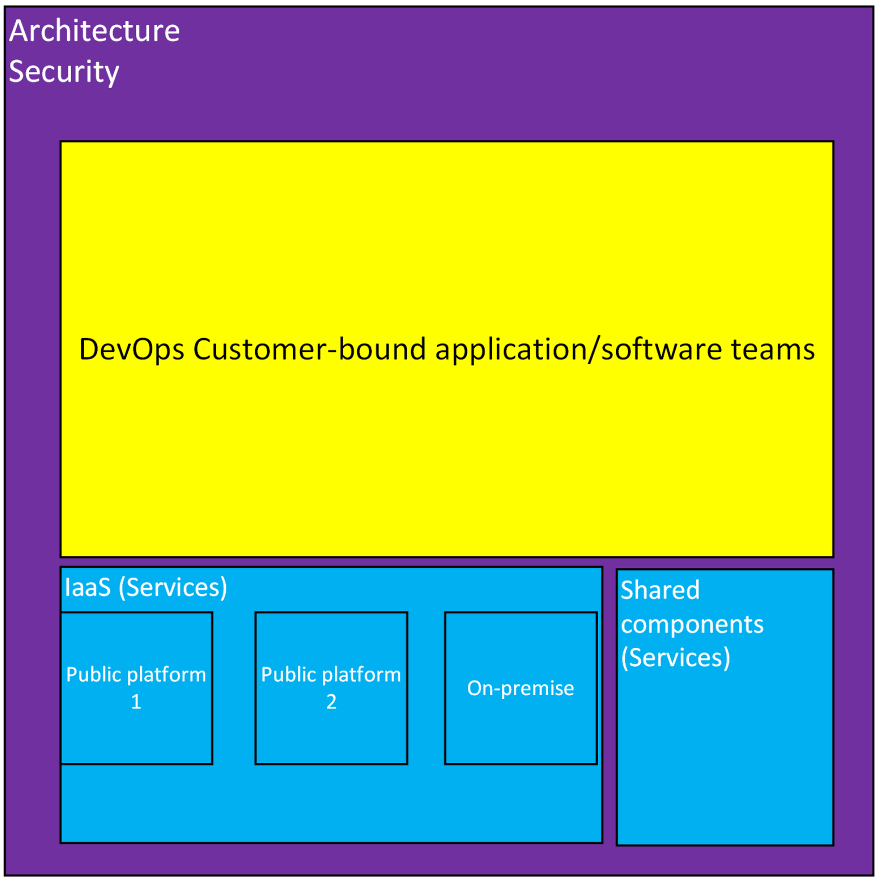

Figure 8: Structure case B

Within the IT department of case B, there are two types of teams: customer-bound application/software teams and shared services (platform) teams. The customer-bound application teams offer application services to one customer. These teams have no direct involvement in developing infrastructure services and solely request infrastructure services. Platform teams facilitate the application teams in the infrastructure layer or other components used across application teams. There are three infrastructure platform teams: Azure, AWS, and on-premise.

Case B does not use a service desk, as the application teams are in direct contact with the client. Furthermore, people within the organization have an extensive network. This results in clear team communication, as people know who to approach. Both architects and security specialists enable the IT department to handle case B.

Processes: The infrastructure platform teams are each responsible for offering an infrastructure platform. The main responsibilities of the infrastructure platform team include:

- offering infrastructure services to the application teams;
- maintaining and updating the platforms to keep the platform secure and up-to-date with the latest trends in the market and the requests of the application teams;
- helping application teams in the request process of IaaS.

Each team within the IT department of case B is entirely self-managing and responsible for their part of the system's plan, build, and run phase. This requires a high maturity of all team members. All teams work according to DevOps; within a team, all team members should be able to execute all plan, build, and run tasks. ITIL processes are used for operational tasks. Application teams can reach the services team on a Slack channel to address operational issues. Each team is responsible for automating their (operational) tasks as much as possible. In case B, automating as much as possible is important, and tasks should be automated instead of hiring a new employee.

In case B, no large-scale team coordination method, such as SAFe, is used. All teams within the IT department, including services, work on a project basis. For the infrastructure teams, the projects are the services requested by the application teams. As each team is self-managing, they can choose their daily way of working. Some teams use methods similar to scrum.

Capabilities: To enable teams to operate autonomously and be self-managing, each application team consists of an architect, operations manager, sales manager, and engineers. The sales manager is unnecessary because the platform teams have solely internal clients. The service directors are responsible for the platform teams and act as operations managers across the platform teams. Another important factor for self-managing teams is the maturity of the team members. On average, case B does not hire employees with less than 10 years of experience in their sector. Case B states that non-technological skills are at least as important as technological skills when selecting new employees.

Culture: Case B has a culture in which individual employees are highly responsible for daily operations and problem-solving tasks. This enables self-managing teams. Another important aspect of the culture is that Case B is a flat organization without strong hierarchical management structures.

Agile culture
Attitude toward learning spirit: 5
Attitude toward collaborative exchange: 5
Attitude toward empowered self-guidance: 5
Attitude toward customer co-creation: 5

Security: The self-managing teams are responsible for their security. Each team has an engineer who takes on the role of security lead and is the team's connection to the security team. The security team is responsible for security across the whole IT department. If necessary, this team offers consultancy services to other teams. Furthermore, the security team is responsible for tooling and reports. Lastly, an audit team handles the audits across the IT department.

5.2.4 Case C

Introduction: Case C is a medium-sized company in the utilities sector that offers two types of infrastructure environments to application teams within the same organization: on-premises and public cloud. It underwent an agile transformation several years ago and now uses the SAFe framework with high maturity, which gives it the flexibility needed to adapt to its changing market.
Role of interviewee: Product manager and Release train engineer
Organization size: Medium
Customers: Internal
Sector: Utilities
Strategy:
Maturity: Level 4
Structure:

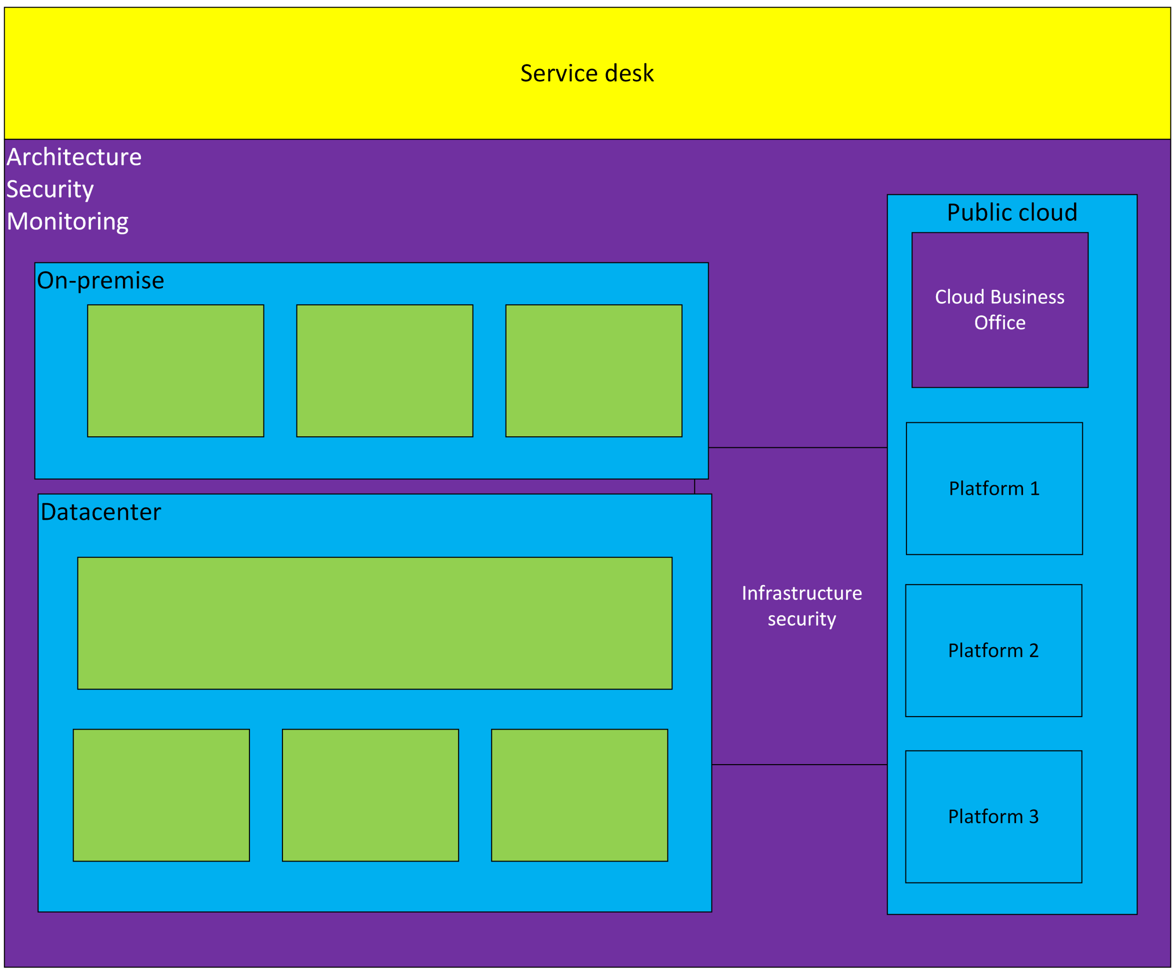

Figure 9: Structure case C

Case C offers two types of infrastructure environments: on-premises and public cloud. The public cloud environment consists of multiple platforms, each offering a vendor enabled by a cloud business office. The cloud business office enables the public cloud platforms by offering onboarding, FinOps, security guidelines, and consultancy. The on-premises environment consists of two platforms: the data center and the hosting layer of the infrastructure. The datacenter layer offers services to the software layer and directly to the client. These two platforms consist of stream-aligned technology-bound DevOps teams.

A monitoring and architecture team enables the platforms. The monitoring team consists of specialists and offers services to the on-premises and public cloud platforms. Furthermore, there are two types of security teams, both enabling the platforms. First, an infrastructure security team offers security services to all the other infrastructure teams. Second, a security team that oversees the whole organization and provides the necessary guidelines and tooling.

There is a service desk. However, this is not a service desk specific to the infrastructure layer.

Processes: The IT department, in case C, operates using the SAFe framework. The teams visualized in Figure 9 are all part of the same agile release train (ART) (except from high-level architecture and the second security team). A scrum master and product owner manage the teams within the ART. On the ART level, a product manager and architect are responsible for the product, a release train engineer is responsible for the ART processes, and a chapter lead is responsible for human resources.

Both the stream-aligned and platform teams work according to DevOps. No clear distinction exists between development and operations within one team, engineers are expected to do both. Case C does not expect engineers to be able to do both. Both types of teams should operate as autonomously as possible.

This ART delivers the services to other ARTs working on applications that need IaaS. The infrastructure services are mostly delivered via one of the platform teams or one of the teams in the on-premises platform. However, when the infrastructure service runs, the application teams need supervision from the infrastructure teams to keep their service running; i.e., the application teams score low on infrastructure maturity. The operational tickets are handled using the ITIL processes.

Capabilities: Case C highlights that IT infrastructure engineers have different capabilities than application engineers. Because of this, scrum masters and product owners for an IT infrastructure team need other capabilities. In general, infrastructure engineers have a strong passion for their discipline, and it is essential that the scrum master and product owner at least understand this passion. In case C, there have been mismatches between application business owners and an infrastructure team.

Culture: In case C, there is an engineering culture where the engineers (and teams) should operate autonomously. Because of this, there is a high level of responsibility at the individual level. The engineers (and teams) take this responsibility very seriously.

Agile culture
Attitude toward learning spirit: 3.5
Attitude toward collaborative exchange: 5
Attitude toward empowered self-guidance: 3.5
Attitude toward customer co-creation: 1

Security: All teams within the ART are responsible for their own infrastructure. When needed, teams can make use of the consultancy services of the infrastructure security team. Furthermore, there is a second (higher-level) security team providing guidelines and tooling across the whole IT organization.

5.2.5 Case D

Introduction: case D is a medium-sized government institution offering on-premises to external customers. Furthermore, case D has clear layers in the structure which are similar to the cloud and does not use a large scale operating model. Case D is able to work effectively without a large scale operating model, because of the maturity of their teams.
Role of interviewee: Relation manager
Organization size: Medium
Customers: Commercial
Sector: Government
Strategy: Modern, sustainable, reliable IT infrastructure by using generic building blocks in a cost-effective way.
Maturity:

Structure:

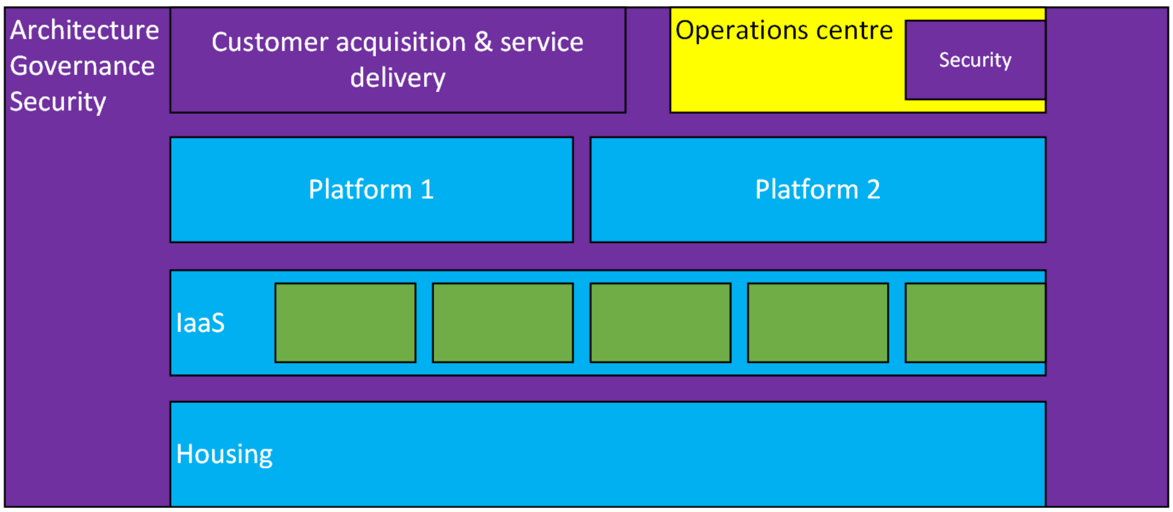

Figure 10: Structure case D

Case D has clear layers in its organizational structure, similar to the layers of the cloud. The bottom layer, housing, offers data center services to the IaaS layer. This layer consists of one team. As the housing team offers its (self) service to the teams in the IaaS layer, it can be seen as a platform team. The IaaS layer consists of 5 technology-bound teams forming the IaaS platform. The next layer is the platform as a service (PaaS) layer. This layer consists of two platform teams, each offering their own platform. Customers of case D can request each layer as a separate service.

Case D has a clear team responsible for the processes around bringing in clients. This is a single team: the customer acquisition & service delivery team. This team is supported by relation management (not in the visual) and is responsible for the client's request up until the provision of the service. The clients have no direct contact with teams below the service delivery team. When the service is provisioned, the operations center becomes responsible for the operational tasks. The operations center also functions as a service desk responsible for operational security and monitoring.

Furthermore, a security team facilitates the PaaS and IaaS teams when they need security consulting. Architecture and Governance are the enabling teams supporting case D.

Processes: Case D does not use a large-scale team operating model such as SAFe. The teams work using Scrum and DevOps. The teams are responsible for the operations of the service they provision. The engineering teams have dedicated scrum masters and product owners per layer. Furthermore, there is a division manager per layer. There are no client-specific projects as case D tries to operate using standard building blocks where possible. The operational processes are implemented according to

ITIL. Case D wants to optimize its processes as much as possible using standardization and automation.

Capabilities: Case D notices that the discipline of IT infrastructure, on average, requires medium-high to high education. However, it mentions that the level of education is not always decisive and runs traineeships on all levels. Furthermore, it is interested in whether an engineer has a developer or operations profile but mentions that the education level does not differ between the two. The developers are the largest group within case D.

Culture:

Agile culture
Attitude toward learning spirit: 4
Attitude toward collaborative exchange: 5
Attitude toward empowered self-guidance: 3.5
Attitude toward customer co-creation: 3

Security: Each team is responsible for securing its product (infrastructure/application). This means that engineers should be DevOps specialists and able to integrate security into their products. The security team within the operations center monitors security across the IT department, and a second security team facilitates the DevOps teams' security issues.

5.2.6 Case E

Introduction: Case E is a medium-sized government institution offering on-premises (outsourced data center) services to internal and external customers. The main focus is on internal customers. Furthermore, case E operates using the SAFe framework, scrum, and DevOps.
Role of interviewee: Systems architect
Organization size: Medium
Customers: Internal
Industry: Government
Strategy: Strategy of case E is strongly dependent on other governmental agencies. Furthermore, case E hosts applications that have a critical social role, making any changes inherently risky.
Maturity: Level 3
Structure:

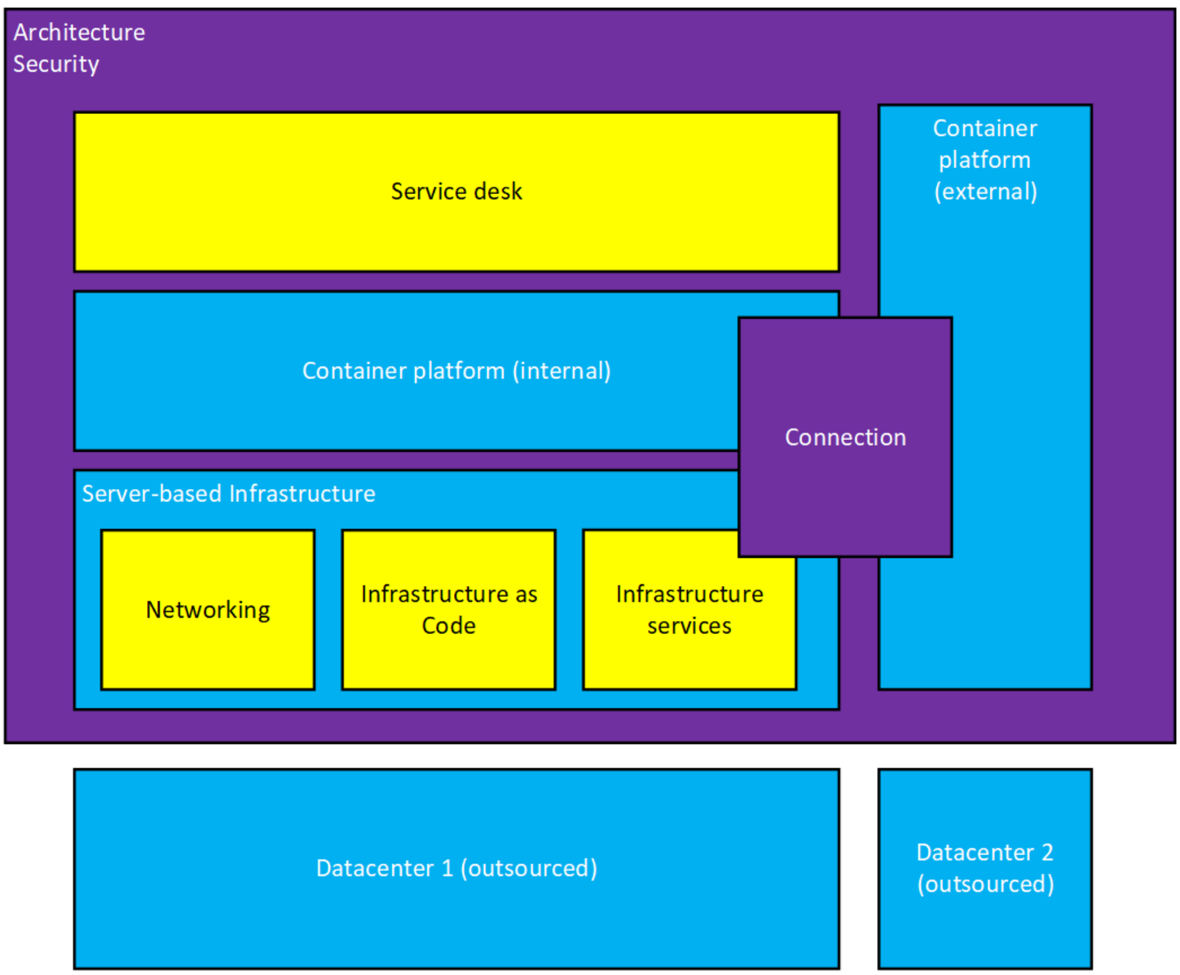

Figure 11: Structure case E

Case E outsources the data center layer to external parties. Furthermore, it offers three platforms: a traditional server-based platform, a container platform for internal use, and a container platform for external clients.

The server-based infrastructure is built on top of the Datacenter 1 platform. Datacenter 1 offers housing and compute, storage, and networking. Case E requests environments from Datacenter 1 that require minimal team customization efforts in the server-based platform. The networking team connects components or (sub)systems if necessary. Furthermore, the infrastructure as code team automates the provisioning of the services. Moreover, the infrastructure services team is responsible for managing the services provisioned by the data center and is the point of contact when the services are provisioned to internal users. The three teams within server-based infrastructure can not all be mapped to the same sub-configuration. Therefore, they are visualized as"regular" stream-aligned teams.

The internal container platform team is built on top of the server-based infrastructure. Case E is trying to use the container platform when possible. Furthermore, there is a service desk for internal users. This skilled service desk tries to solve problems before diverting them to the other internal teams.

The container platform for external users is built upon the services of another data center. The container platform team focuses on offering and maintaining the platform to external users.

The connection team provides the platform with the necessary cross-platform connections. Furthermore, the architecture and security teams enable the internal teams.

Processes: The IT department of case E operates using the SAFe framework. Within the department, there are multiple agile release trains (ARTs). One of these ARTs focuses on IT infrastructure. Approximately 110 people work within this ART. The structure above visualizes the ART. The infrastructure ART is optimistic that the rest of the organization works in the same rhythm. This is because SAFe and sprints are used throughout the whole IT department of case E.

The ART teams operate using Scrum, each with a dedicated Scrum master and product owner. Furthermore, the teams work according to DevOps. Each team is responsible for operating its part of the system, and the external data centers are also responsible for their operation. Case E focuses on easy integration between them and the external data centers. One crucial pillar is collecting and clustering requests before sending them to an external party.

Teams within the art should automate where possible and do as little custom-based work as possible; standardization is key. Furthermore, teams within the art are responsible for their automation. Lastly, at the ART level, the product manager, release train engineer, and systems architect are responsible for the direction/strategy of the ART as well as the people, processes, and technology within the ART.

Capabilities: Within the teams of case E, it is important to have in-depth knowledge of a required technology domain; however, engineers should also be able to work in other domains (T-shape profile). Engineers should be eager to develop their technical and soft skills, which requires a flexible profile. Furthermore, case E believes it is important to identify two types of engineering profiles: development and operational engineers.

The product manager, release train engineer, and systems engineer at the ART level should have clear communication skills to function as a trinity.

Culture: The culture of the whole organization of case E is exceptionally open and transparent. Furthermore, case E is a flat organization. The culture of the infrastructure ART differs a bit from most ARTs within the organization. The infrastructure ART supplies the "track" where the other agile release trains can ride. This makes the role of the infrastructure ART more facilitating than most ARTs.

Agile culture
Attitude toward learning spirit: 4
Attitude toward collaborative exchange: 4
Attitude toward empowered self-guidance: 3
Attitude toward customer co-creation: 5

Security: The DevOps engineering teams are responsible for their security, so they contain the necessary security specialists. Furthermore, security is configured at all levels of the organization. Security teams at the ART level and audit teams offer services to the entire organization. Likewise, an SOC and a security team focus on governance and regulation. Most security teams offer services to the engineering teams within the ARTs.

Security is a serious theme within case E, and multiple teams often have similar tasks. These teams take the same set of tasks seriously, usually resulting in multiple layers of security.

5.2.7 Case F

Introduction: Case F is a large organization (approximately 900 employees within the infrastructure department) with a strong focus on the private cloud. Furthermore, it has a small number of applications running on the public cloud. The private cloud environment consists of multiple platforms.
Role of interviewee: Advisor datacenter services
Organization size: Large
Customers: Internal
Industry: Government
Strategy: Could not be shared
Maturity: Level 3
Structure:

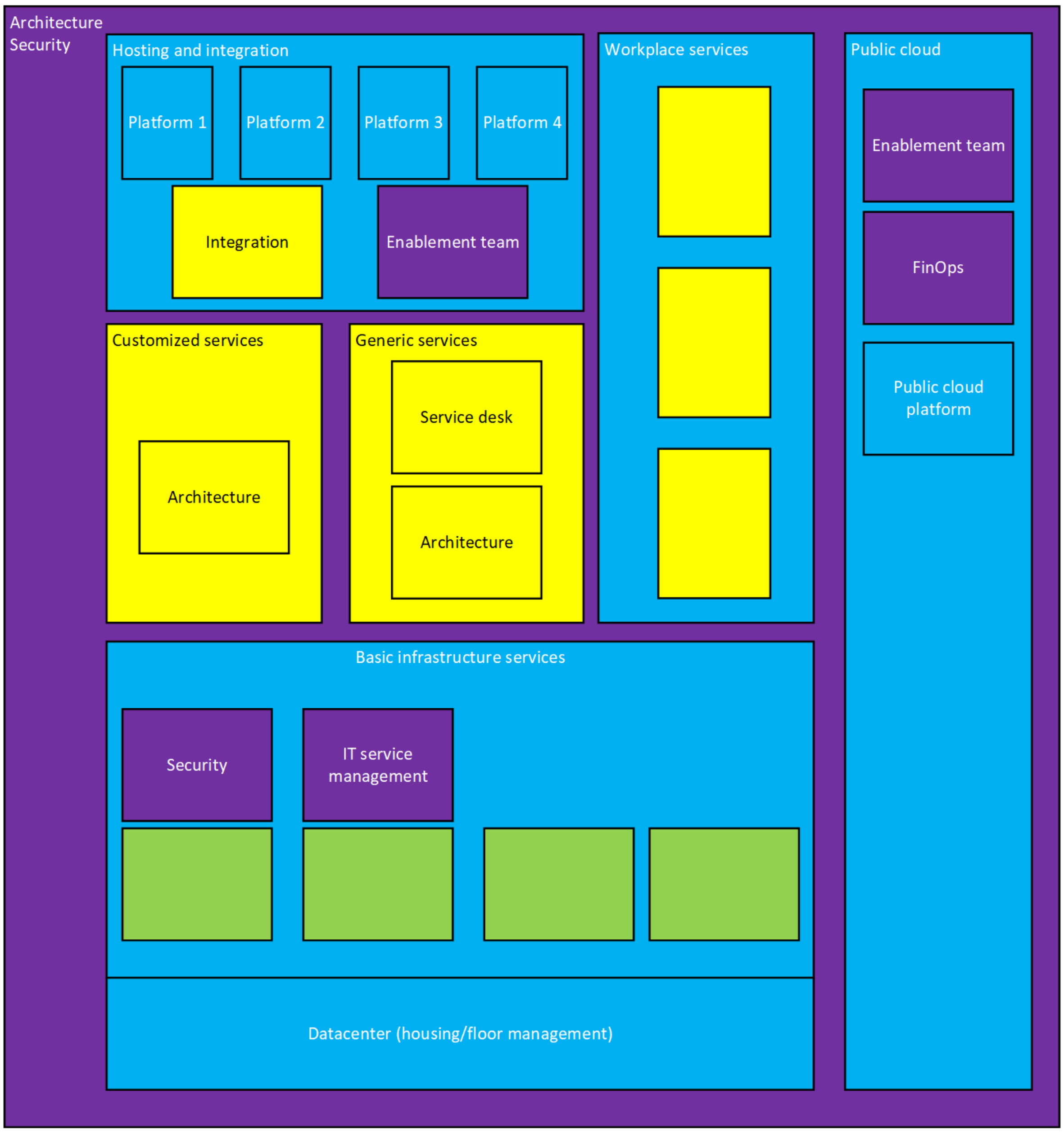

Figure 12: Structure case F

The structure of case F is layered, where all teams are within the infrastructure department of case F.

The bottom layer is the data center of case F, where the hardware components are managed. This layer functions as a platform for the basic infrastructure services platform, which forms the basis for all customer services. Basic infrastructure services consist of technology component teams and are enabled by a dedicated security and service management team.

Case F offers two types of services: generic and customized. Each service has its stream, and both streams use the platform basic infrastructure services to create their service. The generic and customized services form the basis for the hosting and integration platforms. Four types of platforms are offered to the customers, each with its own team. Furthermore, the integration team integrates the system's components. Lastly, one of the platforms has an enablement team that helps clients make use of the platform.

Case F also offers workplace services within its organization, which are part of its infrastructure department. The workplace platform consists of multiple stream-aligned teams. Furthermore, Case F provides a public cloud platform for a few applications. The public cloud platform is enabled by an enablement team that helps clients use it. Furthermore, a multidisciplinary FinOps team takes care of the finance part of the public cloud.

Processes: Because case F is such a large organization, teams have some freedom regarding which processes they think suit them. For example, the workplace services team uses SAFe, as this team has a clear end-user and is closely connected to the end-user. Most teams operate using DevOps practices. Furthermore, some teams use Scrum or Kanban. Within Case F, the processes are mainly determined per department.

Case F uses traditional managers involved in people, processes, and technology throughout the entire infrastructure department. This results in responsibilities being taken away from scrum masters and product owners, as some responsibilities are allocated twice.

Capabilities: Case F emphasizes that you need strong technical skills for both the development and operation sides of infrastructure, as well as soft skills, such as the ability to communicate effectively with your fellow engineers. Furthermore, employees within case F should be able to work in an ever-changing environment and have automation skills.

Culture: The culture within case F can be described as trustworthy. However, people find taking risks difficult due to the large number of internal checks. The large number of internal checks and the high standard can make it difficult to take risks. Another cultural aspect is the can-do mentality, where local heroes are often needed to solve the hardest problems. Within case F, people are dedicated, especially in times of crisis. Lastly, when the culture of the infrastructure department is compared to the culture of the application department, the infrastructure department scores higher on engineering culture than the application department.

Because of the fact that case F has teams which work in an agile way and teams that use a more traditional way of working the agile attitudes were determined for both type of teams.

Agile culture: traditional way of working
Attitude toward learning spirit: 2
Attitude toward collaborative exchange: 4
Attitude toward empowered self-guidance: 3
Attitude toward customer co-creation: 4

Agile culture: Agile way of working
Attitude toward learning spirit: 4
Attitude toward collaborative exchange: 4
Attitude toward empowered self-guidance: 2
Attitude toward customer co-creation: 2

Security: The technology component teams are responsible for their security; however, each department has a dedicated security team. This team can consult the departments' teams on implementing the security requirements. Furthermore, there is an overarching security team focusing on governance and guidelines.

5.2.8 Case G

Introduction: Case G is a large international company in the financial services industry. It uses both public and private clouds. The private cloud consists of a traditional server platform and a container platform. Furthermore, the infrastructure services are offered internally to the application teams within the IT department.
Role of interviewee: Chapter lead
Organization size: Large
Customers: Internal
Sector: Financial services
Strategy: Provide a foundation to build assemble and operate modular propositions
Maturity: Level 4

Structure:

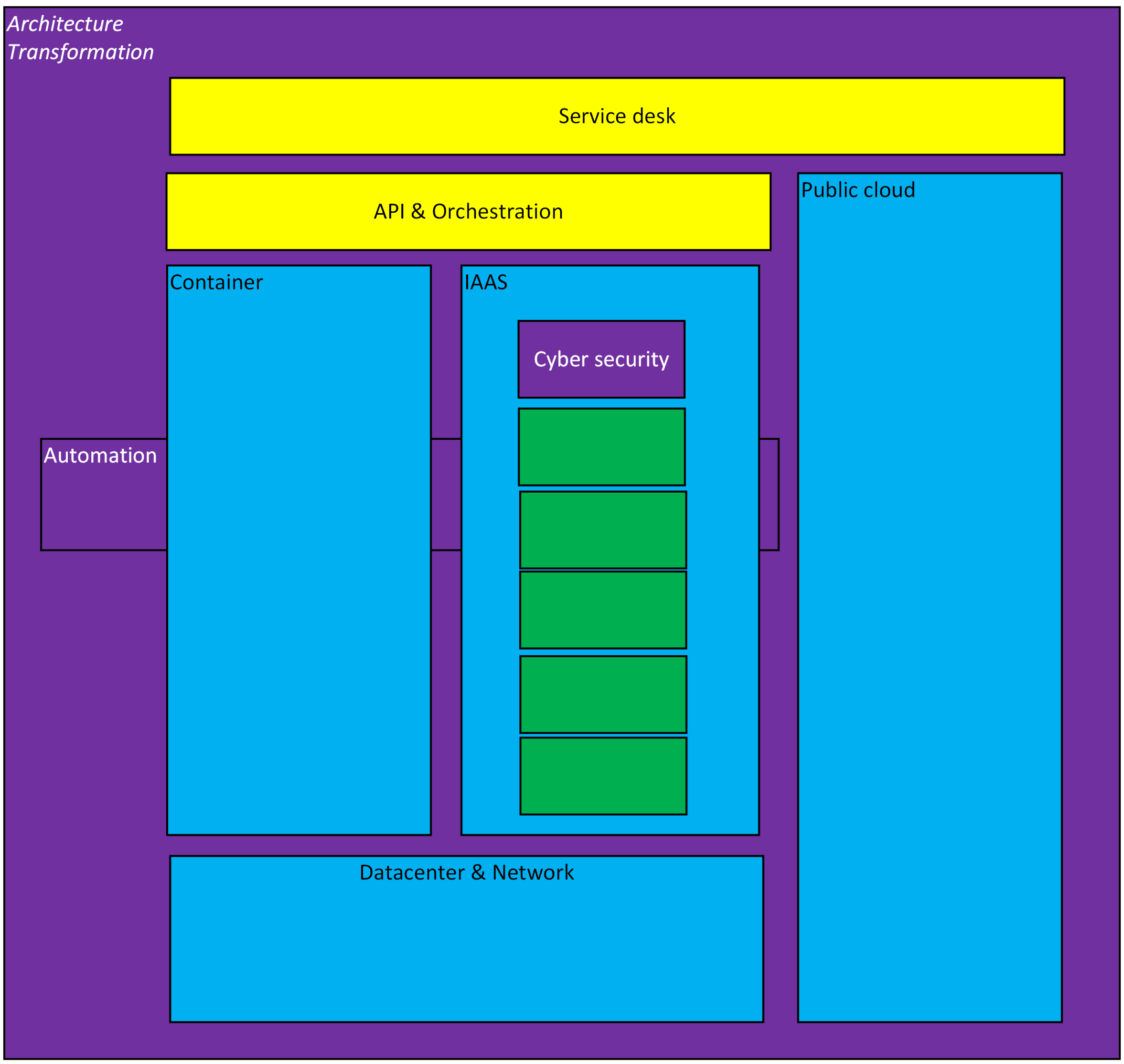

Figure 13: Structure case G

Case G offers both on-premises and public cloud IaaS. The interview focused on the on-premises situation; the public cloud platform was not discussed in depth in the interview. Nevertheless, it is still essential to visualize the structure. The on-premises IaaS can be split into a container and a traditional (server-based) IaaS platform.

The IaaS platform consists of technology-bound teams. The teams within IaaS build their services upon the services of the data center & network layer/platform. The automation team is an enabling team. This team handles automation for the IaaS and container platforms, reducing their cognitive load. The automation does not require Ph.D.-level knowledge and, therefore, is not a complicated subsystem team. Furthermore, there is a cyber-security team within IaaS. This team ensures the end-product of IaaS is secure and facilitates the other teams with security consulting.

There is a separate stream for API & orchestration. This stream is responsible for integrating the product, as well as high-level automation and orchestration. Furthermore, a non-skilled service desk directs the issues to the right team. Lastly, all visualized teams are enabled by architecture and transformation. The transformation team is responsible for non-technical transformations within the organization.

Processes: Case G has implemented the Spotify model. The Spotify model mainly prescribes how to structure an organization, whereas SAFe complements this prescription with more detailed processes. Therefore, case G needs autonomous teams guided by a chapter lead on the people aspect, and a tribe leads on the process and technology aspect. This is correctly implemented within case G.

The technology-bound teams within the IaaS platform work according to DevOps and Scrum processes. Consequently, the teams within IaaS are responsible for operating their part of the provisioned service. Within most teams, a go-to-guy is used, who is responsible for the operational tasks of that day/week and has an exemption for the scrum tasks that day/week. This role rotates across team members. However, not all team members are obliged to be a go-to guy. The operational processes are implemented according to ITIL. Lastly, each team has a dedicated scrum master and product owner who needs to be mature and experienced to function correctly within the Spotify framework. The Spotify framework states that one manager should no longer be responsible for people, processes, and technology. Instead, these three items should be divided among dedicated managers.

Capabilities: Case G amplifies that having a diamond-shaped capability profile per technology-bound team is essential. The diamond shape refers to the number of juniors (low), mediors (medium), and seniors (low) within a team. When selecting new employees, case G searches for employees with the following capabilities: collaboration, market focus, problem analysis, performance expectations & motivation, and organizational sensitivity apart from the technical capabilities.

Culture: Case G has a culture in which everyone is treated equally, and its workforce is diverse. Furthermore, the culture within the infrastructure department is different from the culture of the software department. This is mainly visible in higher cohesion in the infrastructure department.

Agile culture
Attitude toward learning spirit: 5
Attitude toward collaborative exchange: 4
Attitude toward empowered self-guidance: 4
Attitude toward customer co-creation: 2

Security: Each team is responsible for securing its product (infrastructure/application). This means that engineers should not only be DevOps specialists but also be able to integrate security into their products. The security team within IaaS facilitates the other teams with security consulting if necessary. Furthermore, they check whether the provided services are secure.

5.2.9 Case H

Role of interviewee: Enterprise architect
Introduction: Case H is a very large IT service provider offering multi-cloud (public & private cloud) as a commercial activity. Case H does not focus on a single customer segment but provides services across all customer segments. This results in a broad portfolio of services.
Organization size: Large
Customers: Commercial
Industry: IT services
Strategy: Could not be shared
Maturity: Level 5

Structure:

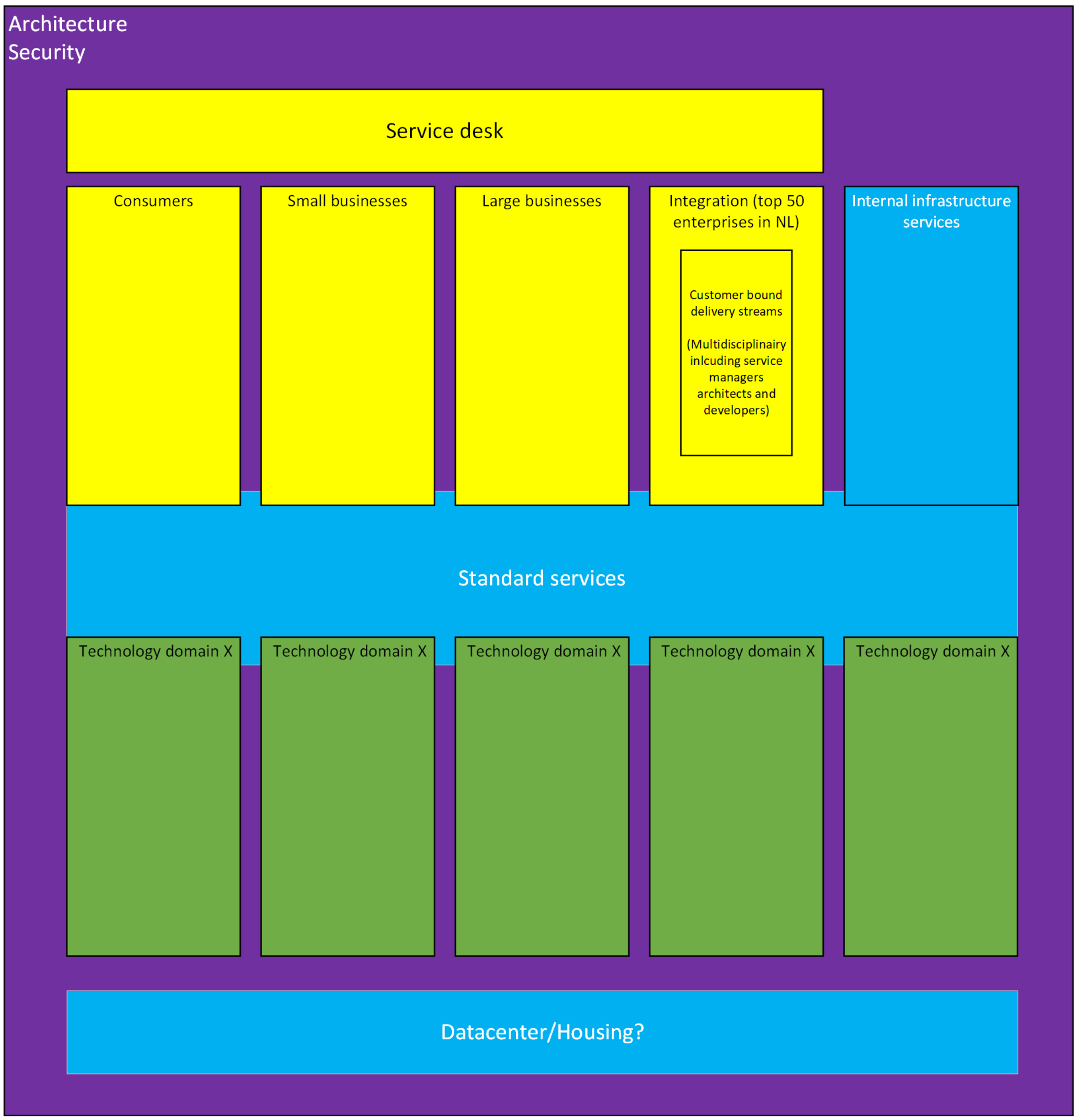

Figure 14: Structure case H

Case H is a very large organization because of this, the structure focuses on a high-level overview. There are clear layers in the structure of Case H. The bottom layer is the data center layer, which functions as a platform for the layers above. On top of the data center layer are multiple technology component streams. There are separate streams for each technology domain, as knowledge across domains varies widely. Within these streams, agile teams work on delivering their part of the technology stack.

Above the technology streams is the platform standard services. This platform offers the technology components as a service to the different streams above. These are the four customer-bound streams and the internal infrastructure platform, which is used to provide infrastructure services to the internal IT organization. The customer-bound streams each focus on a separate customer segment and are responsible for offering the infrastructure to the customer. The work of the streams containing larger clients involves more custom-based work.

Above the customer streams is a service desk consisting of two types of service desks: one focusing on generic/standard services and the other on larger clients with customized services. Lastly, architecture and security enable the infrastructure department of case H.

Processes: Case H operates using SAFe in combination with scrum DevOps teams. However, case H emphasizes that when projects become capital-intensive, scrum and a short cycle way of working are no longer effective. When short cycles are used in the processes, the project becomes too sensitive to lateral steering. Because investments within IT infrastructure become capital-intensive quickly due to high prices for new hardware, not all projects are suitable for scrum. When case H implements capital-intensive projects, the goal is to let their clients consume the service in an agile way, without approaching the project in an agile way.

The operational processes of case H start at the skilled service desk. Only when the skilled service cannot solve the problem are the DevOps component streams are involved in solving the problem. The success rate of solving problems at the service desk is relatively high (>80%). The operational processes of case H are implemented according to ITIL.

Capabilities: As case H works with streams per technology domain, the engineers in these streams need deep knowledge of their technology domain. The agile way of working within these streams requires not only a high level of technological skills but also interpersonal and soft skills. The engineers within the streams had to increase their interpersonal skills.

The managers of a stream are mostly people managers. This is because it is hard to acquire people with both management and technology skills. Furthermore, according to SAFe, managers should not be too involved in technological decisions.

Culture: The culture within case H changed when it started working according to SAFe and Agile. The biggest switch was for engineers who suddenly needed to be involved in customer contact. Case H is currently trying to raise its customer intimacy, however notices that people with both technological skills and people skills are hard to find. Furthermore, case H notices that switching to an agile culture for capital-intensive projects is difficult.

The agile culture variables are given for the part of the organization that works in an agile way.

Agile culture
Attitude toward learning spirit: 5
Attitude toward collaborative exchange: 5
Attitude toward empowered self-guidance: 5
Attitude toward customer co-creation: 5

Security: Within case H, there are teams focusing on helping the clients become more secure and teams focusing on making the internal organization more secure. Both are forms of security consulting.

5.2.10 Case I

Introduction: Case I is a medium-sized organization operating in financial services. Case I offers on-premises and public cloud environments, however, case I is currently fully transitioning to the public cloud. Furthermore, the infrastructure services are offered internally to the application teams within the IT department.
Role of interviewee: Solution Architect
Organization size: Medium
Customers: Internal
Industry: Financial services
Strategy: Cloud first
Maturity: Level 4
Structure:

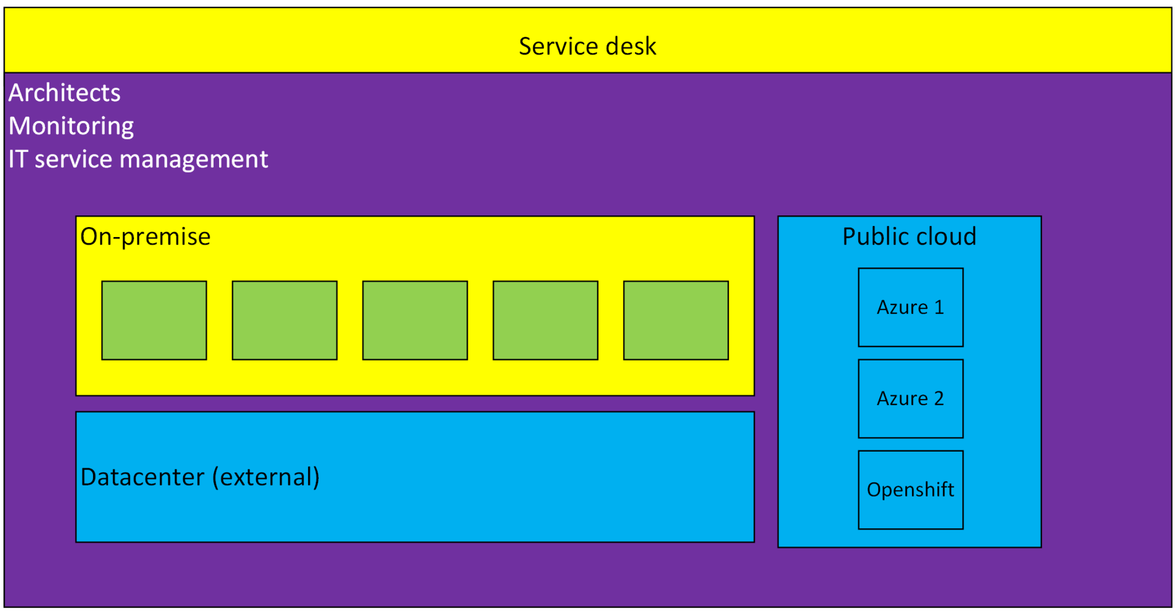

Figure 15: Structure case I

Case I offers on-premises and public cloud environments. The data center layer for the on-premises environment is outsourced to an external service provider and functions as a platform for the on-premises stream. The on-premises hosting layer consists of multiple technology-bound teams. These teams do not have the coherence to function as a platform. Therefore, the on-premises environment is drawn as a stream instead of a platform.

The public cloud environment consists of multiple platform teams. In case I, two types of Azure platforms and an Openshift platform are offered as a service.

Architecture and monitoring enable the on-premises and public cloud teams. Furthermore, the IT service management department offers enabling services in different aspects: governance, service management tooling, requirement management, security, orchestration, and high-level automation. Lastly, a semi-skilled service desk outside the infrastructure department offers services to the infrastructure department. The service desk is able to solve workspace related issues and routs other issues to the right teams.

Processes: Case I has not yet implemented a large-scale operating model. Currently, it is trying to implement SAFe. However, entry-level maturity still has to be reached. Both the on-premises and public cloud teams operate using DevOps and Scrum. Each team has dedicated product owners and scrum masters, and each team has a go-to guy for operations.

Case I is trying to standardize as much as possible. However, sometimes applications require customized infrastructure. When application teams require more access to the infrastructure, this is possible with the necessary training. Furthermore, Case I is trying to move as many of its applications as possible to the public cloud. Consequently, the focus is more on the public cloud platform teams than the on-premises teams.

Capabilities: The public cloud teams, especially the Azure teams, require multidisciplinary skills from the engineers. In other words, engineers should be able to work with more than one technology component. This knowledge does not stretch across all technology components and does not have to be in-depth. The technology component teams (on-premise) require in-depth knowledge within a technology domain.

Furthermore, the infrastructure customers (the application teams within the organization) can get more rights and access to their infrastructure environment. This requires the application teams to perform infrastructure training, after which upgraded rights and access are provided.

Culture: On average, engineers are experts within their technology domain or platform. However, they struggle to transfer this knowledge to other teams when necessary. The employee's safe space is mainly within the team and not so much outside of the team. Furthermore, the fact that there is such a strict line between the on-premises and public cloud environments in terms of organizational structure and required mindset makes it difficult to connect engineers across the two environments.

Agile culture
Attitude toward learning spirit: 4
Attitude toward collaborative exchange: 2
Attitude toward empowered self-guidance: 3
Attitude toward customer co-creation: 4

Security: All teams are responsible for their security. However, when teams need security consultancy, the security team within IT4IT can provide such services. Furthermore, teams need to request exceptions from the security team.

5.2.11 Case J

Introduction: Case J is a medium-sized government institution offering on-premises cloud to internal customers. It is in an ongoing agile transition that has not yet been successful. Currently, Case J is trying to implement the SAFe framework.
Role of interviewee: Technical architect & Release train engineer
Organization size: Medium
Customers: Internal
Industry: Government
Strategy: The strategy is to simplify and shrink the application landscape, outsource generic ICT activities, maintain a top quality level, and ensure conscious compliance.
Maturity: Level 2

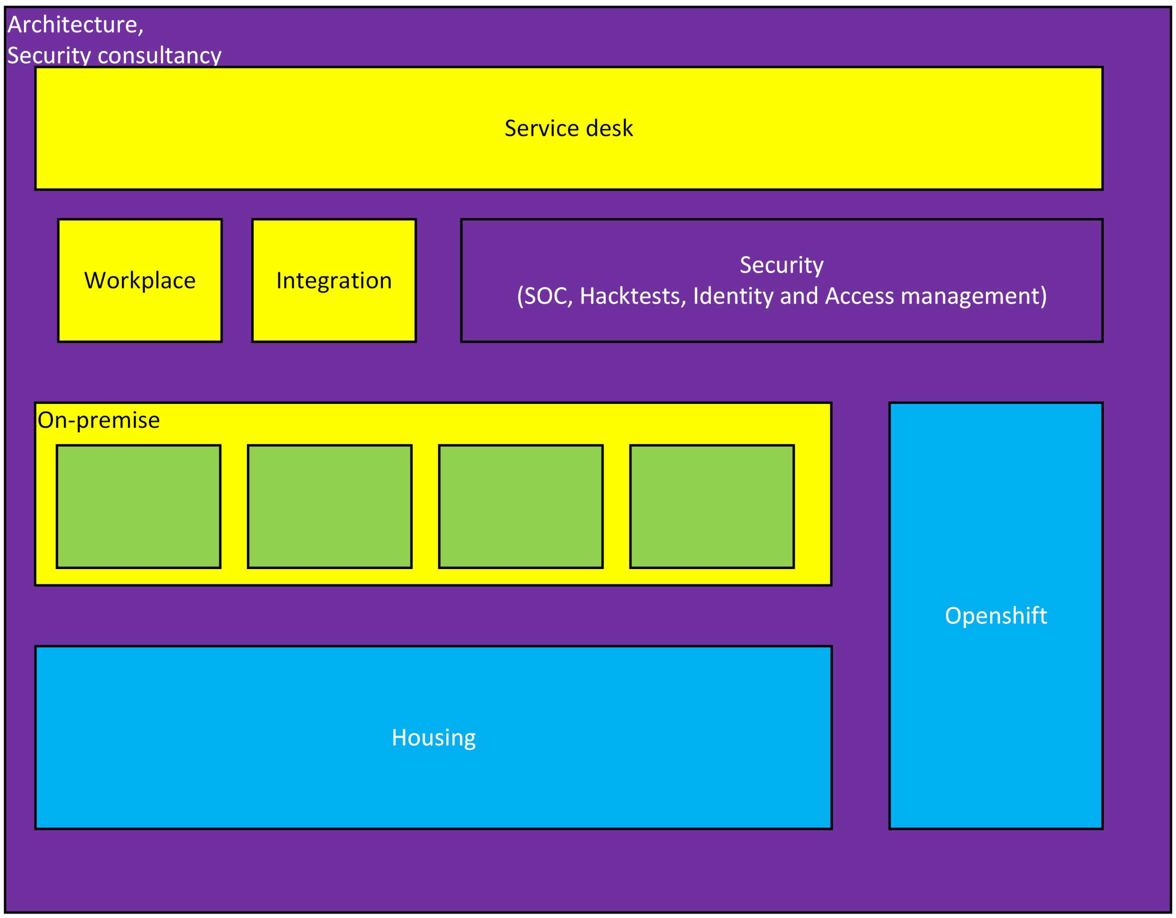

Figure 16: Structure Case J

Case J performs its housing layer in collaboration with another government body. It is responsible for its own rackspace, servers, etc., within the data center. The housing layer functions as a platform for case J's on-premises layer.

The on-premises layer is visualized as a stream, as the technology-bound teams within the stream do not function as a platform. This is explained further in the process section. The Openshift platform is maintained by an external partner and tailored to case J.

On top of the platforms are the workplace, integration team, and security department. The security department has a SOC, a hacktest team, and an identity and access management team. There is another security team focusing on consultancy services across the IT department. The security consultants and architects enable all teams. Lastly, a skilled service desk is used across the whole IT department.

Processes: Case J has not yet (fully) implemented a large-scale operating model. Currently, it is trying to implement SAFe. The agile transformation within case J started long ago and has not yielded success yet. At the beginning of the transformation, traditional technical team leads were given the label product owner, and some teams were given a scrum master. People were expected to start working agile without any training suddenly. Case J is currently trying to start raising the maturity level of the scrum methodology. However, the maturity level varies largely across teams. Case J is aware that to implement SAFe, a higher agile maturity is needed first.

The technology component teams work using DevOps. Within the teams, there is a go-to guy for operational tickets. The operational processes are implemented according to ITIL. The technology component teams do not function as a platform, as the automation grade is insufficient. Furthermore, the application teams often use shortcuts to get their product faster. This results in a high number of customized tasks and ad hoc work, which is not in line with the platform philosophy.

Within case J, the maturity of product owners and scrum masters must be raised to let the scrum framework function properly. Furthermore, case J still uses traditional managers who are involved in people-process technology. This means there is an overlap between the responsibilities of the traditional manager and the product owner or scrum master.

Capabilities: As case J is still implementing Scrum and SAFe properly, it is looking for capabilities to upgrade its maturity level for these frameworks. This means case J is letting its employees participate in Scrum and SAFe training. One of the short-term goals of case J is to give each team a dedicated scrum master and product owner, with proper training, to let teams function properly.

Culture: The culture within case J can be described as free in multiple aspects. Engineers have a lot of freedom; however, this is also due to absent guidelines from (higher) management. This allows engineers to implement their roles and make the technical decisions they want. Furthermore, the willingness to help each other is high. This results in too much ad hoc work across teams and at the team level.

Agile culture
Attitude toward learning spirit: 3.5
Attitude toward collaborative exchange: 3
Attitude toward empowered self-guidance: 3
Attitude toward customer co-creation: 2

Security: Within case J, the DevOps technology components teams are responsible for creating a secure product. There are multiple security teams to help and test this. There is a security department where each product is hack-tested. Furthermore, the department includes the SOC and identity and access management. Lastly, a security team provides security consultancy services through the IT organization of case J.

5.2.12 Case K

Introduction: Case K is a medium-sized company in the manufacturing sector that offers on-premises infrastructure internally. It clearly distinguishes between developing new services and provisioning them. Furthermore, the SAFe framework is used as an operating model in combination with a project-based way of working.
Role of interviewee: Domain architect
Organization size: Medium
Customers: Internal
Sector: Manufacturing
Strategy: Could not be shared
Maturity: Level 4
Structure:

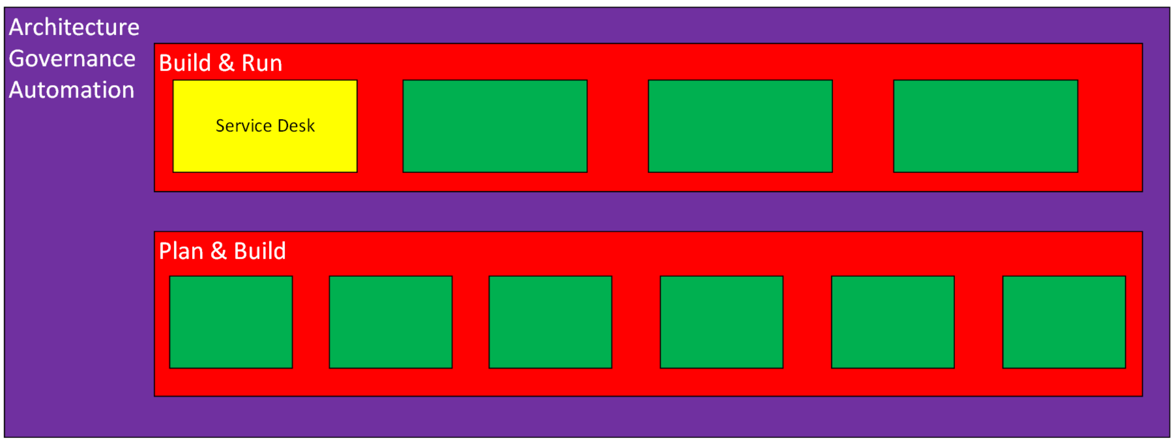

Figure 17: Structure case K

To fully understand the structure of case K, it is essential to understand that case K creates customer solutions based on service blocks. A service block is a piece of technology that can be offered to the client or is part of the product that is offered to the client. Service blocks should be deployed using as much automation as possible.

Case K operates using two streams; the streams take on development or operational tasks. This means the streams are processed-based streams, which are visualized in red. The Plan & Build stream is responsible for creating new service blocks. The Build & Run stream is responsible for offering the service blocks to the client and performing the operational tasks when the service block is running. The Plan & Build stream consists of 6 technology-bound teams, and the Build & Run stream consists of a service desk and roughly mirrors the technology component teams in Plan & Build. Technology-bound teams consist of 8-20 people. The interviewee could not release more details on the Build & Run stream.

The automation agile release train (ART) enables the two stream, however the teams are also responsible for their own automation as well. Lastly, architecture and governance enable the two streams.

Processes: The IT department of case K operates using the SAFe framework. There are multiple ARTs, two of which are strongly linked to IT infrastructure service development and delivery. Within these ARTs are technology component teams with a dedicated scrum master and product owner, leveraging the Scrum methodology. The structure section explains that development and operation tasks are separated between the ARTs. The ART Plan & Build focuses on developing service blocks for ART Build & Run to provision and operate these services in larger quantities. The interviewee notes that it is easier to distinguish between development and operations when working with hardware components.

Plan & Build operates using SAFe and Scrum. However, the projects Plan & Build are working on are long (e.g., 1.5 years or longer). The duration and large number of dependencies can make it harder to use Scrum. For example, the domain architects, each overlooking three technology components teams in Plan & Build, have chosen to divert from Scrum and only use a Kanban board.

Plan & Build tries to use automation by design when possible. However, this is hard as it requires more work (and financial support). Furthermore, it requires a skill set different from most engineers' within case K.

Capabilities: Engineers within the technology component teams should not only be experts within their technology domain but also possess certain soft skills. The lead engineer should mostly be able to communicate clearly and think along with the domain architect.

The domain architect has to overarch multiple technology domains on people, processes, and technology. The interviewee stresses that domain architects work across these three axes, where people might be the most critical dimension. The enterprise architect should be able to apply these skills on a higher technology level.

Where architects focus on people, process, and technology, infrastructure coordinators focus on the processes across technology component teams to let the teams function as an ART and platform. Lastly, the product manager and release train engineer within the ART should be able to have an extensive internal overview of both the ART and the rest of the organization.

Culture: The workforce of case K consists of many nationalities and cultures. This results in a diverse workforce where people respect each other and listen to each other's ideas. All employees want to help improve case K, which makes working together pleasant.

Agile culture
Attitude toward learning spirit: 5
Attitude toward collaborative exchange: 5
Attitude toward empowered self-guidance: 4
Attitude toward customer co-creation: 4

Security: Security must be involved when developing new service building blocks or other capabilities. Furthermore, security is central within the organization across the people and technology axis. Within case K, security teams are working on all types of security, and all teams can offer consultancy services to the teams in need.

5.2.13 Case L

Introduction: Case L is a medium-sized IT service provider offering private and public infrastructure services focusing on networking. It is part of a bigger international group and has expertise in critical IT infrastructure. Lastly, Case L is experiencing ongoing growth.
Role of interviewee: Lead architect
Organization size: Medium
Customers: Commercial
Industry: IT services
Strategy: Could not be shared
Maturity: Level 3
Structure:

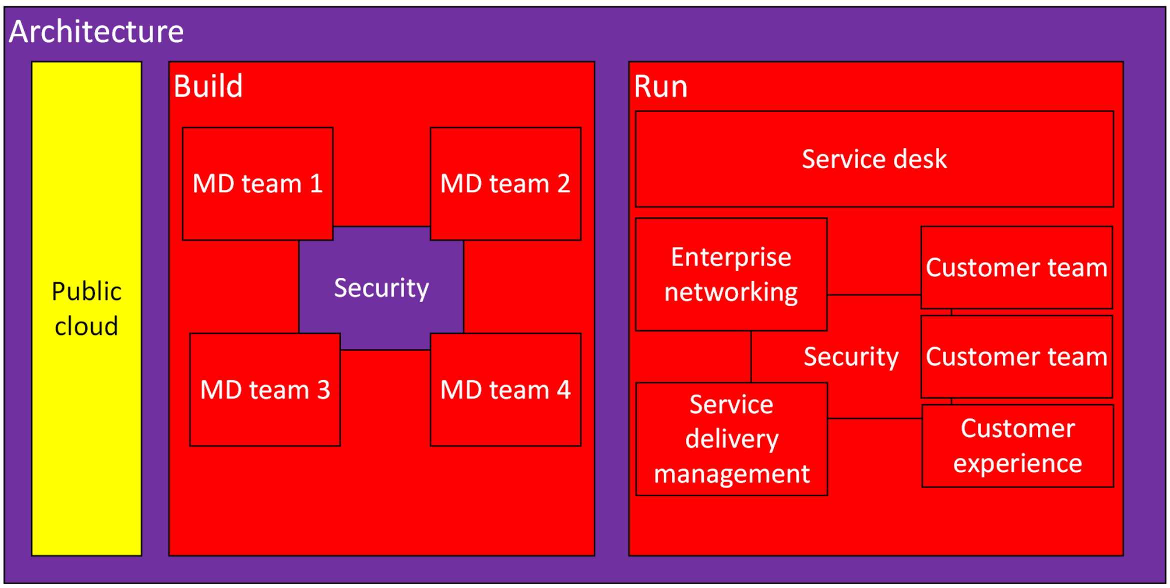

Figure 18: Structure case L

In the structure of case L, there is a strong split between building/developing services and the operational/run side of these services. There are two separate streams, each focusing on one of these processes, thus visualized in red. The build stream focuses on developing and provisioning new services for clients. It consists of four multidisciplinary teams. Each multidisciplinary team has one or multiple clients for whom they develop services. The build stream has a dedicated security enablement team.

The run stream focuses on the operational side of the services provisioned to clients. Clients can buy services from the build stream without using the operational services from the run stream. The run stream consists of two customer teams, focusing on operational tasks for a specific (large) customer. Furthermore, there is a customer experience team helping all clients get the most out of their services. In addition, there is an enterprise networking team focusing on the operational tasks linked to networking and a service delivery management team monitoring the quality of the services. Lastly, there is a security team taking care of security-related operational tasks. Note that this security team is different from a security operations center (SOC), as in this case, the focus is on, for example, maintaining firewalls.

Furthermore, there is a public cloud team. The public cloud team handles development and operational tasks, so it is visualized as a regular stream-aligned team. Lastly, the architecture team enables all streams.

Processes:
Case L does not use a large-scale operating model such as SAFe. Instead, processes are determined per stream. The build stream uses scrum methodologies, even though it does not have dedicated scrum

masters and product owners. The project leaders within the build stream function as product owners, and each has two teams; the team leader functions as a scrum master. Furthermore, scrum rituals are used within the build stream.

As the multidisciplinary teams (6-8 people) each focus on a different set of clients, work is sometimes not distributed evenly. The multidisciplinary teams should be able to handle projects from other clients and distribute work more evenly. Furthermore, the multidisciplinary teams sometimes have to help with operational tasks when the task is too difficult for the run stream or when the run stream is too busy to handle all tasks. When a project is delivered to the client and the client wants to outsource the operational tasks to the run stream, the build stream hands the project over to the run stream.

The run stream mainly works according to ITIL and uses some agile rituals, such as a daily start. A critical moment for the run stream is the handover from the build stream. Furthermore, the engineers within the run stream must stay in contact with the customer's operators. The service desk within the run stream is a non-skilled service desk that redirects operational tickets to the right teams/engineers.

Capabilities:
The skill level of the engineers at run is slightly lower than that of the engineers at build. However, this difference was historically larger and is now decreasing because the engineers at run need a higher skill level to maintain the products. Furthermore, case L has identified the following capabilities for a multidisciplinary team:

- Technical consultants: focusing on the consultancy aspects of the client contact and developing the products.
- Engineers: a strong product development focus, less client contact.
- Project manager: focusing on the progress of the project.

Culture:
In case L, employees want to help each other when possible. Furthermore, health and stress are important items which are monitored by HR. The culture within run differs slightly from the culture within build as the average age within run is lower.

Agile culture
Attitude toward learning spirit: 3
Attitude toward collaborative exchange: 5
Attitude toward empowered self-guidance: 4
Attitude toward customer co-creation: 4

Security:
The engineers within build should have security expertise, which the security team can enable. Furthermore, case L has recently paid more attention to security awareness (internally and at the client), which might lead to a new team. Case L also helps clients with security issues or audits.

5.2.14 Case M

Introduction: Case M is a large IT service provider offering on-premises and public infrastructure solutions. Furthermore, case M has a clear split in the development of products and the operational side of products. In addition, case M does not use a large-scale operating model, and each team has the freedom to choose their own way of working.
Role of interviewee: VP Managed Services
Organization size: Large
Customers: Commercial
Sector: IT services
Strategy: "Clear and futureproof IT solutions with innovative technology, whilst also tackling current IT challenges."[81]
Maturity:
Structure:

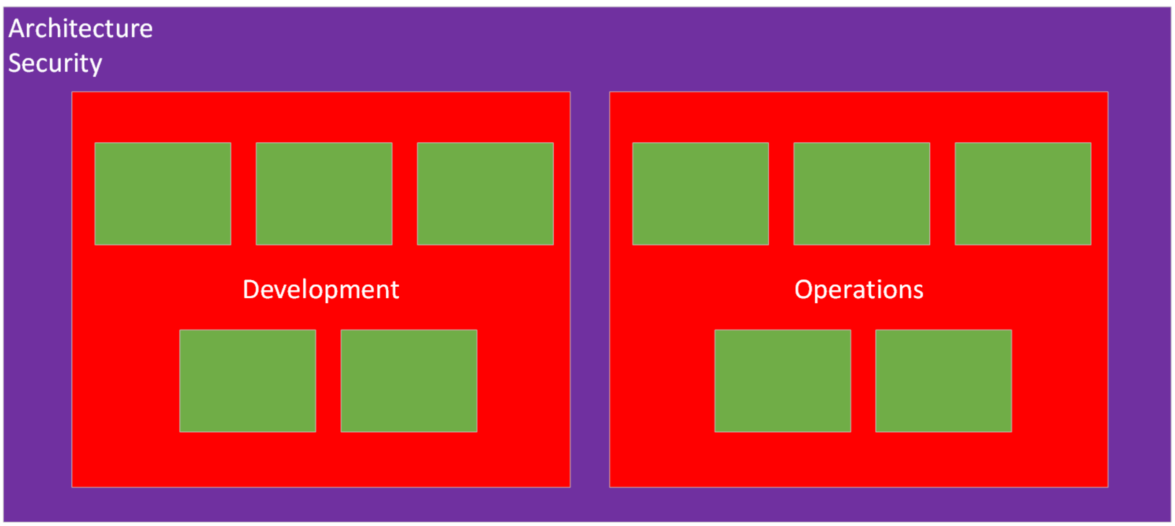

Figure 19: Structure case M

In the organizational structure of case M, there is a clear split between development and operational tasks. There is a stream solely focusing on developing and provisioning products and a stream solely focusing on the operational side of these products. In other words, each stream takes care of a different process, thus process streams visualized in red. Within these streams, there are competence centers, where each competence center focuses on a single technology domain. Each competence center consists of one or more teams and have specialized architects. Furthermore, each competence center has specialized architects. The competence centers work together to form a stream and are visualized in green.

Both competence centers are enabled by an overarching architecture and security team.

Processes:
Case M does not use a large-scale operating model. Instead, process design is mainly determined at the team level. The processes at team level can differ, each team determines whether they use Scrum or another daily approach. Most teams choose to use Scrum. Furthermore, ITIL is mostly used within the operations stream. When looking at the high-level process, there is a clear split between development and operational tasks. This results in a very low adoption rate of DevOps at the team level.

Because of the fact that other teams develop products than the ones maintaining the product, it is important to create easily maintainable products. The development stream should especially keep this in mind when creating new products. Furthermore, integrating the development and operational teams is important because communication between these two streams should be minimal after the product is handed over.

Each competence center and stream has its own manager. Furthermore, most of the scrum teams have scrum masters and product owners. However, not all scrum masters and product owners are dedicated to a team or do not fill the position full-time.

Capabilities:
Capabilities were left out of the interview of case M due to time constraints.

Culture:
The culture of case M can be described as customer-focused and pragmatic. The culture within the infrastructure department does not differ significantly from the rest of the organization.

Agile culture:
Attitude towards learning spirit: 4
Attitude toward collaborative exchange: 5
Attitude towards empowered self-guidance: 3.5
Attitude towards customer co-creation: 4

Security:
The security teams of Case M advise internal teams and clients on security guidelines. Internally, the guidelines are implemented by the development teams. When the development teams struggle, they can consult the security team. For clients, the security team mainly advises them; however, development teams can implement a solution.

5.2.15 Case N

Introduction: Case N is a large IT service provider offering private and public infrastructure services. It has customers in the public and private sectors. Lastly, Case N can be characterized by a strong split between development and operational tasks.
Role of interviewee: Relation manager & Sales manager
Organization size: Large
Customers: Commercial
Sector: IT services
Strategy: Support the execution, optimization, and transformation of customers' businesses by offering a full range of ICT services and solutions
Maturity:
Structure:

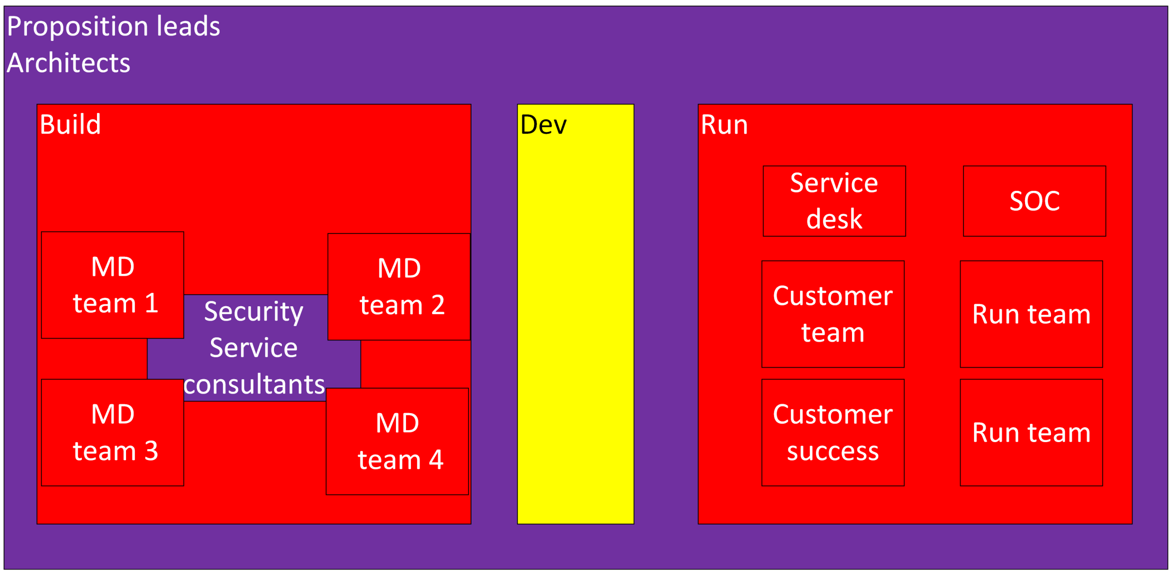

Figure 20: Structure case N

In the structure of case N, there is a clear split between building/developing services and the operational/run side of these services. There are two separate streams, each focusing on one of these processes, thus visualized in red. The build stream focuses on developing and provisioning new services for customers. The build stream consists of several multidisciplinary teams. Furthermore, the build stream has a dedicated security consultancy team. Lastly, the build stream has a service consultancy team. The service consultants focus on customer intake with the proposition lead and help customers request the right services. The development stream focuses on providing supporting IT services to the build and run streams. An example of this is the CMDB.

The run stream focuses on the operational side of the services provisioned to the customers. Customers can purchase services from the build stream without using the operational run stream. The run stream consists of several teams that are focused on the customer. Furthermore, general run teams are providing operational services to multiple teams. In addition, a customer success team helps customers get the most out of their services. The service desk functions as a front office for all operational issues and can be characterized as a skilled service desk as they can solve issues for customers without help from other teams in the run stream. Lastly, a security operations center (SOC) focuses on security monitoring.

All streams are enabled by architecture and the proposition leads. The proposition leads identify market trends, customer needs and are experts in their technology domain.

Processes:

Case N does not follow a single, organization-wide working model. Instead, streams and teams operate according to their own approach. Some teams within case N apply Scrum practices, while the build stream takes a more traditional, project-oriented approach, with project leaders responsible for managing delivery. Scrum methodologies are applied in parts of the organization, but their use is limited in areas where physical infrastructure components are involved, as these can complicate the application of all Scrum principles. The teams in the build stream have strong technical expertise and are capable of delivering highly customized solutions. The proposition leads and service consultants define the objectives for the build teams, who then develop the actual services.

The development stream focuses on managing internal support services using tools such as TopDesk. Scrum is not used in this stream, though some Kanban practices are applied. For the operational tasks in the run stream ITIL is being used. Furthermore, within the run stream, there is an increasing emphasis on automation. Some customer-specific teams exist in the run stream, these teams have a closer connection and relationship with the customer. The architecture team plays a central role in aligning the work between the different streams. In practice, services delivered to clients often represent a mix of standard and customized offerings, typically resulting in a 50/50 balance.

Capabilities:
Due to time constraints, capabilities could only be briefly discussed for case N. In case N, it is important to be able to work in a changing environment and be customer-focused. Furthermore, case N identifies that engineers' maturity levels can be slightly lower in the run stream.

Culture:
The culture of case N revolves around "work hard, play hard". Furthermore, the whole company has a strong customer focus and is driven by an entrepreneurial mindset.

Due to time constraints the agile attitudes could not be included for case N.

Agile culture:
Attitude towards learning spirit:
Attitude toward collaborative exchange:
Attitude towards empowered self-guidance:
Attitude towards customer co-creation:

Security:
A separate consultancy team provides services mainly to the build teams. Furthermore, a security operations center monitors the provisioned services for security breaches or risks.

6 Discussion

In this chapter, we analyze the findings presented in the results section. While the previous section described the results from 14 different cases individually, this section brings those cases together by identifying four key patterns that share similar characteristics in terms of structure and the frameworks used. The discussion chapter consists of five sections, each contributing to the development of a model that can be used by organizations in real life to design their organization. This section begins with a brief overview of the key topics to be discussed, followed by detailed answers to the research questions in the subsequent subsections.

By organizing the discussion around patterns rather than individual organizations, we aim to highlight broader insights that can be applied more generally. Section 6.1 provides an overview of these four patterns that could be identified based on the results and answers the first research question:

1. What are organizational design patterns for a managed service provider that implemented automatic IaaS provisioning?

These patterns are found at an organization level, however, there are also team configurations, which are present at team level. These team configurations are part of the patterns. Information on team level is equally important as the organization level in order to implement a pattern. These team configurations are presented in section 6.2. which addresses research question 2:

2. What are the configurations of teams in organizational design patterns for managed service providers that implemented automatic IaaS provisioning?

In addition to the four configurations proposed in the team topologies model, two new sub-configurations are introduced. These new configurations were introduced because it was not possible to fit all real life case teams into one of the existing team configurations. After introducing these new configurations, the need for them will be further discussed in section 6.2. After the patterns and the team configurations within a pattern are discussed in sections 6.1 and 6.2, the following section looks at how organizations can figure out which pattern is the best fit for them. By doing so, section 6.3. answers research question 3:

3. Which critical factors should an organization consider when choosing a pattern?

To help clarify this, a decision tree is developed to guide organizations in choosing the most suitable pattern. Following the decision tree, organizations can identify which pattern fits their specific situation. After the three research questions are answered, the answers to the research questions are combined to form a complete model. This model consists of the patterns, team configurations and the critical factors.

Section 6.4 applies this model to an existing managed service provider. This example helps to illustrate and validate how the model works in real-life and makes its application more concrete.

Finally, section 6.5. will reflect on the team topologies model. In this section, the limitations of the team topologies model, which is mentioned in the literature review, are described. Due to a lack of definitions for stream-aligned configurations and value streams, it is challenging to align real-life teams with these configurations. This section proposes a new, clear definition of stream-aligned configuration and value streams to solve this issue.

6.1 What are organizational design patterns for a managed service provider that implemented automatic IaaS provisioning?

To answer research question 1, cases with a high level of similarity are grouped in a pattern. It is essential to notice the difference between a case and a pattern. A case represents an individual company in terms of the components of organizational design: structure, strategy, frameworks, capabilities, culture, and security. A pattern, on the other hand, is a collection of cases that share common characteristics across these same components, emphasizing high-level similarities rather than individual specifics. This section will answer the first research question by introducing four patterns, after which interesting takeaways will be presented.

In the analysis of the results, four organizational patterns are identified:

1. Platform-driven self-managed platform (Case A & B)
2. Platform-driven managed service provider (Case C - H)
3. Solution-driven managed service provider (Case I & J)
4. Solution-driven Dev & Ops provider (Case K - N)

The identified patterns can be visualized based on two categories: type of service delivery and type of operational governance. Figure 21 presents a matrix-based overview of the four identified patterns based on these two categories:

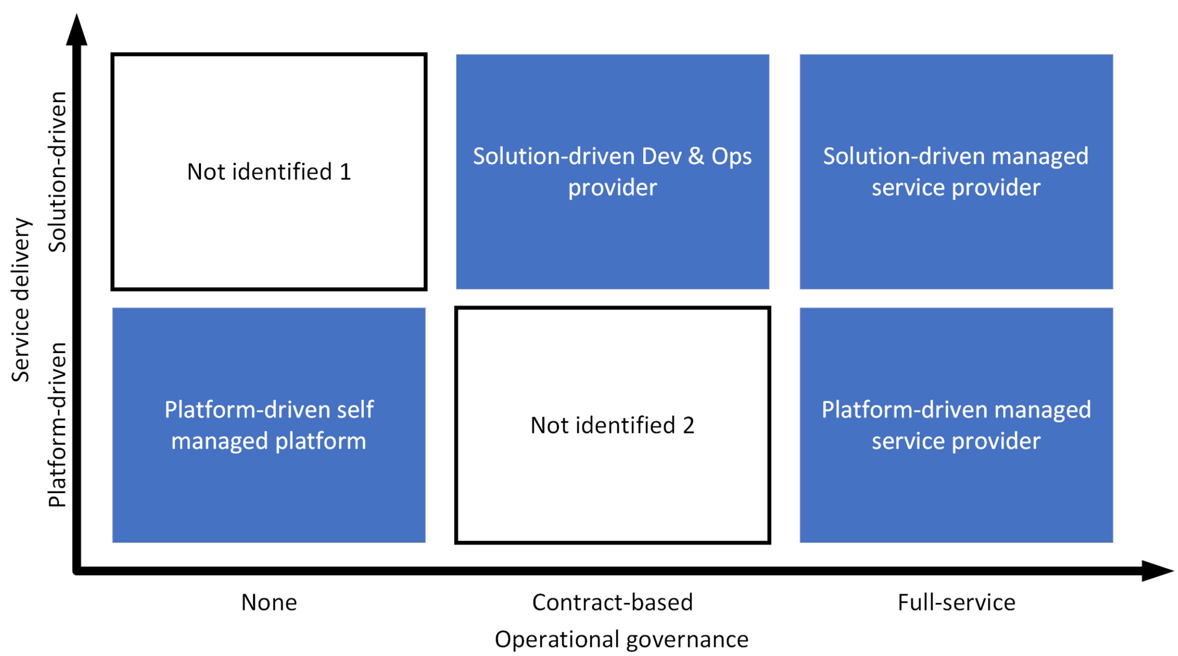

Figure 21: Overview of patterns

The patterns have been named based on these categories. The matrix indicates that, within the defined categories, two patterns remain unidentified. This suggests that no individual cases are classified as solution-driven with no operational governance, nor are any cases categorized as platform-driven with contract-based operational governance. The unidentified patterns will be further discussed in 6.1.1. Below is a definition of each category, after which a short description of each pattern is provided.

Service delivery:

The type of service delivery can either be platform or solution-driven. Platform-driven service delivery focuses on offering standardized services through an automated platform using a service-oriented strategy. A platform/service-oriented strategy is characterized by organizational-wide practices, policies, and procedures designed to "support and reward service-giving behaviors that create and deliver service excellence"[82][p.136].

A solution-driven service delivery focuses on building the right products for the clients using a customer-oriented strategy. A customer-oriented strategy is characterized by a business approach that emphasizes the understanding and satisfaction of customer needs and preferences. It entails aligning products, services, and interactions to improve customer satisfaction and strengthen loyalty [83]. Across cases, it was found that standardization and automation might be lower because of a strong customer focus. This is because a high degree of customization entails higher costs, longer wait times, and greater customer involvement in the service delivery process [84].

Operational governance:
The operational governance category describes the agreements between the service provider and the customers on maintenance and operational services. The type of operational governance can be either none, contract-based, or full-service. When operational governance is nonexistent, the IaaS provider offers no maintenance or operations services for its services. For full-service, the IaaS provider always offers maintenance and operations for its services. When operational governance is contract-based, the IaaS provider makes arrangements per customer regarding whether maintenance and operations are offered for the provided service.

A short description per pattern is given below:
Platform-driven self-managed platform: is based on cases A & B and is characterized by a single large public cloud platform team, a high level of standardization, automation, and no operational governance within the platform team. This pattern is part of a larger IT organization, where infrastructure services are offered to internal DevOps application teams. The DevOps application teams consuming the infrastructure services need infrastructure capabilities within their teams.

Platform-driven managed service provider: is based on case C-H and is characterized by a large on-premises platform consisting of multiple teams, a smaller public cloud platform consisting of various teams, and a high level of standardization, automation, and operational governance. The application teams consuming the infrastructure services have no operational governance regarding the infrastructure services.

Solution-driven managed service provider: is based on cases I & J and is characterized by an on-premises environment and a smaller public cloud platform. Furthermore, there is a low to medium level of automation and standardization, high operational governance, and a strong customer focus. Lastly, the application teams consuming the infrastructure services have no operational responsibility regarding infrastructure services.
The service-driven managed service provider and solution-driven managed service provider have some similarities. However, the solution-driven managed service provider lacks service consultants and a dedicated resource management layer. This results in a pattern where the technology components yield more to the customer's opinion, with a lower degree of standardization and automation as a consequence. Furthermore, the customer-driven managed service provider lacks a team overarching framework. Overall, this results in an on-premises stream instead of a platform.

Solution-driven Dev & Ops provider: is based on cases K-N and is characterized by a strict split between development and operational tasks in the organization, a low level of standardization, and a strong customer focus. The application teams consuming the infrastructure services can execute the operational tasks themselves or outsource them to the operation department. Furthermore, on-premises and public cloud solutions are offered.

In the Appendix a detailed description per pattern is provided.

Table 9: Short description per pattern

6.1.1 Not identified patterns

Two patterns are not identified in the multiple case study: the solution-driven self-managed platform and the platform-driven pattern, where operational governance is contract-based. The fact that these patterns were not found in the 14 cases presented in the results does not mean they do not exist in organizations. We will now evaluate whether an organization could implement these patterns by looking at the characteristics, as the scope of the research does not enable further research into these patterns.

First, the solution-driven self-managed platform. In this pattern, the customer manages the services the IaaS provider provides as operational governance is none. Additionally, the provider focuses on developing the right products for clients using a customer-oriented strategy. We believe that such a provider could exist but argues that a customer-oriented strategy is typically more effective when paired with a higher operational governance level from a business model perspective. On the other hand, the solution-driven Dev & Ops provider, where operational governance is contract-based, demonstrates that it is possible to implement a customer-driven strategy without providing operational governance. However, the contract-based solution-driven Dev & Ops provider does offer the option of operational governance.

Second is the platform-driven pattern, where operational governance is contract-based. In this pattern, the IaaS provider makes arrangements per customer regarding whether maintenance and operations are offered for the provided service. Furthermore, the provider focuses on offering standardized services through an automated platform using a service-oriented strategy. We believe such a provider could exist and considers it even more likely than the previously unidentified pattern. Both the customer and the provider can maintain standardized services, as demonstrated by the identified platform-driven patterns. A hybrid approach would require a matching organizational design. This could result in a separation between development and operations, as seen in the solution-driven Dev & Ops provider. In contrast to the solution-driven Dev & Ops provider, this provider would focus on offering standardized services, making customer-focused teams less suitable.

6.1.2 The importance of a team overarching framework for an on-premises infrastructure platform

To be able to operate as an on-premises platform, as implemented in the platform-driven managed service provider, the on-premises environment should be "a foundation of self-service APIs, tools, services, knowledge, and support which are arranged as a compelling internal product. Autonomous delivery teams can make use of the platform to deliver product features at a higher pace, with reduced coordination"[49]. To offer self-service APIs and other platform functionalities, standardization, and automation are key [85]. To guarantee standardization in the pattern platform-driven managed service provider, SAFe is used as a team-overarching framework. Furthermore, the maturity scores of cases C-H on the team-overarching framework are medium-high, showing the importance of proper implementation. Moreover, no team-overarching framework is used in the pattern solution-driven managed service provider, resulting in lower standardization and automation. As a result, the on-premises environment in this pattern does not function as a platform, but as a stream, showing the importance of such a framework.

It is important to note that activities from multiple teams are needed in the control layer to implement a self-service on-premises platform.

6.1.3 The role of maturity on standardization and automation

The interviewee is asked to rank the organization on the COBIT maturity framework during the interviews. The COBIT maturity framework describes five maturity levels: (1) Ad hoc, (2) Repetitive but intuitive, (3) Defined, (4) Managed and measureable, and (5) Optimized[79]. More information on the COBIT framework can be found in 5.2.1. The key finding on maturity for the patterns is that it plays a critical role in standardization and automation. Standardization and automation are key to the platform-driven patterns. In the platform-driven patterns, the maturity scores are medium-high.

Furthermore, a higher maturity score is related to effectively implementing a team overarching framework, such as SAFe. In contrast, a low maturity score is linked to such a framework's incomplete or improper implementation. This is shown by the platform-driven managed service provider, whose maturity scores are medium-high, and a team-overarching framework is used. In contrast, there is no proper implementation of a team-oversight framework for the solution-driven managed service provider. Moreover, the lower maturity score for this pattern results in a stronger focus on customer-specific needs, often at the expense of standardized processes, standardized products, and scalability.

6.1.4 Infrastructure capabilities within application teams

When operational governance is none, the consuming application teams are responsible for maintenance and operational tasks. Because of this, application teams have to contain infrastructure specialists *(case A & B)*. Literature confirms that IT infrastructure capabilities are crucial for application development projects [86]. When infrastructure capabilities are undesirable in application teams, one of the managed service provider (MSP) patterns can be used. MSPs proactively manage the IT for their customers [9, 10], relieving application teams of the cognitive load of operating and maintaining infrastructure services.

When using an MSP pattern, application teams remain responsible for requesting the right infrastructure services; however, service consultants can be used to reduce the cognitive load. Service consultants help customers order the proper services on the right platform and are used in two patterns: Service-driven managed service provider and solution-driven Dev & Ops provider.

6.1.5 Other interesting findings

The results section contains a culture component per case, describing the culture and a score on agile attitudes. The results of the culture component are inconclusive and will, therefore, not be further presented in the discussion section.

6.2 What are the configurations of teams in organizational design patterns for managed service providers that implemented automatic IaaS provisioning?

In the previous section, high-level information on the organizational patterns was provided. This section will use the team configurations of the team topologies model to provide more details on the configuration of teams within patterns. Furthermore, two new sub-configurations of the stream-aligned configuration are introduced: the technology component and the process configurations. Lastly, the configuration per pattern at the team level will be given.

In the results section the team topologies model is used to indicate the role of a team in the organization. Following the team topologies model, there are four team configurations [49]:

1. **Stream-aligned teams:** Stream-aligned teams serve as the primary team configuration within an organization. They are directly responsible for delivering value and are closely involved in the continuous flow of change. The other team types exist primarily to support stream-aligned teams in their activities.

2. **Enabling teams:** Enabling teams are designed to assist stream-aligned teams in achieving rapid progress by facilitating significant improvements and helping them overcome obstacles that require specialized expertise. To fulfill this role, enabling teams consist of specialists with the necessary knowledge and skills to address domain-specific or technical challenges.

3. **Platform teams:** Platform teams deliver services that enable stream-aligned teams to operate with complete autonomy.

4. **Complicated-subsystem teams:** Complicated-subsystem teams consist of Ph.D.-level specialists responsible for parts of the system that depend on deep, specialized knowledge.

More information on the configurations can be found in 2.3.4. Table 4 shows an overview of the team configurations used per case.

6.2.1 Newly identified team configurations for IaaS providers

Classifying teams into one of the four original team configurations of the team topologies model during the analysis phase proved to be difficult. As a result, two new sub-configurations of the stream-aligned configuration are introduced. The first challenge was the absence of a clear definition of the stream-aligned teams and value streams; see 6.5.1 for more information. This lack of a clear definition for stream-aligned teams blurs the distinction between the stream-aligned configuration and the other three configurations, making it challenging to identify the stream-aligned teams.

The second challenge relates to the set of criteria for stream-aligned teams. According to the team topologies model, stream-aligned teams should be multidisciplinary, utilizing DevOps practices, and should not hand over the product to other teams [49]. In practice, however, it was difficult to find teams that met all these requirements. For example, multidisciplinary teams are only present in case L & N, but these teams did not meet the other requirements.

To address these limitations, two new configurations are introduced in the results section: the technology component configuration and the process configuration. Both configurations are sub-configurations of the stream-aligned configuration as they are involved in the flow of change [49]. Table **??** shows an overview of which requirements are met per configuration, after which a short description of the new configurations is given.

	Stream-aligned	Platform	Enabling	Complex subsystem	Technology component	Process
(1) Directly involved in the flow of change						
(2) Multidisciplinary						
(3) Handles support issues (DevOps)						
(4) No handoffs of the product						
(5) Takes care of one technology component						
(6) Provides internal services (X-as-a-service)						
(7) Helps stream-aligned teams overcome obstacles						
(8) Consists of Ph.D. level experts						

Technology-based: While studying the cases, it became apparent that multiple stream-aligned teams across cases were organized around a single technology component or domain. The workflow of technology-based teams is organized around technology components or domains, such as storage, images, or backups. The technology-component teams are responsible for developing and operating their part of the technology stack, which aligns with the stream-aligned configuration.

Process-based: In other cases, it became apparent that multiple stream-aligned teams were involved in either development or operations. Process-based teams work in an organization characterized by a split between development and operational tasks. They are involved in one of these workflows.

The fact that for every organization, one of the sub-configurations is needed shows that it is challenging to fit real teams into one of the four configurations of the team topologies model. This is confirmed by the literature of Hollenback et al., which states that due to a large number of team taxonomies published in literature, a researcher can't precisely identify a team's correct configuration using these taxonomies [87]. Instead, they propose a three-dimensional framework in which authority differentiation, skill differentiation, and temporal stability are used to categorize teams [87].

6.2.2 Team configurations per organizational pattern

This section will provide a short, high-level description of the team configurations per pattern; Appendix gives a more detailed description per pattern. This description also describes the layers of the service model used in the visualization.

Platform-driven self-managed platform: is based on a single platform team offering mostly infrastructure services from a public cloud vendor. The platform team might consist of sub-teams per vendor. The platform team delivers the services to DevOps application teams.

Platform-driven managed service provider: consists of an on-premises platform with multiple inner configurations: platform teams and technology-based teams. The technology-based teams form a server-based platform. An automation team enables the teams in the on-premises platform. Next to the on-premises cloud platform could be a public cloud platform that functions similarly to the platform-driven self-managed platform. Multiple teams in the control and service layer enable these two platforms.

Service-driven managed service provider: consists of an on-premises stream of technology-based teams. The technology-based teams do not function as a platform as in the platform-driven managed service provider. Next to the on-premises stream could be a public cloud platform that functions similarly to the platform-driven self-managed platform. Lastly, there is a single enablement team in the resource abstraction and control layer.

Solution-driven Dev & Ops provider: consists of two process streams: development and operations. The development streams can consist of either technology- or process-based teams; a combination was absent in the cases. There is a single enabling team in the service layer.

6.3 Which critical factors should an organization consider when choosing a pattern?

Based on the findings in the result section, six critical success factors are identified for the patterns: (1) service delivery, (2) operational governance, (3) maturity, (4) organization size, (5) infrastructure capabilities within application teams, and (6) size of on-premises environment. The critical factors can be used to choose one of the patterns. A detailed description per critical factor is given below after which a visualization of the decision tree follows:

Service delivery: The type of service delivery can either be: platform or solution driven. A platform driven service delivery focuses on offering standardized services through an automated platform. A solution driven service delivery focuses on building the right products for the clients. Because of this strong customer focus standardization and automation might be lower.

Operational governance: The operational governance type can be either none, contract-based, or full-service. When operational governance is none the IaaS provider is offering no maintenance or operations services for the services they provision. For full-service, the IaaS provider always offers maintenance and operations for their services. When operational government is contract-based, the IaaS provider makes arrangements per customer whether maintenance and operations are offered for the provided service.

Maturity: The score on the COBIT maturity scale [79] as used in the results section. The maturity score was given by the interviewee:

- (1) Ad hoc: Everything happens ad hoc, there is no standardization at all.
- (2) Repetitive but intuitive: Repetitive tasks happen the right way, but there is no documentation at all.
- (3) Defined: Processes and responsibilities are defined and documented.
- (4) Managed and measurable: Processes are not only defined, but include version control and have specific checkpoints to verify that execution is correct. Risks to the business are identified, documented and potentially mitigated. Vendor strategy and lifecycle management strategies are used.
- (5) Optimized: Alerts are in place if the process is bypassed and management programs are in place to continuously improve risks, risk mitigations and procedures.

Organization size: the size of the infrastructure department. It can be either small (<50 employees or <5 teams), medium (50-150 employees or 5-15 teams) or large (>150 employees or >15 teams). Size is solely based on the number of employees as the difference between workload for public cloud and on-premises is too big. This is because an on-premises environment needs people to take care of the housing (the data center), which is not the case for a public cloud platform. The same sizing is used in the results section.

Infrastructure capabilities within application teams: This refers to whether the application team has infrastructure specialists or infrastructure capabilities within their team. For the category *not present*, the application teams have no infrastructure specialists or capabilities within their team. For medium, there are infrastructure capabilities within the team. However, there are no dedicated infrastructure specialists within the application team. Lastly, for high, there are dedicated infrastructure specialists within the application team.

Size of on-premises environment: This variable is solely relevant for pattern 1. For the other three patterns, the size of the on-premises cloud is not a crucial factor. The size can be either small, where the on-premises environment can be managed by at most 10-15 people, or large, where more people are needed.

In figure 22 a decision tree is visualized, showing which variables are critical for which pattern. For some variables critical factors have been skipped, this is because these factors are not critical for the pattern. The blue path in the decision tree leads to a specific pattern, while the red variables indicate areas an organization should enhance before adopting that pattern. Organizations can use the decision tree to see which pattern would suit their organization, after which the organizational design framework can be used to implement a pattern.

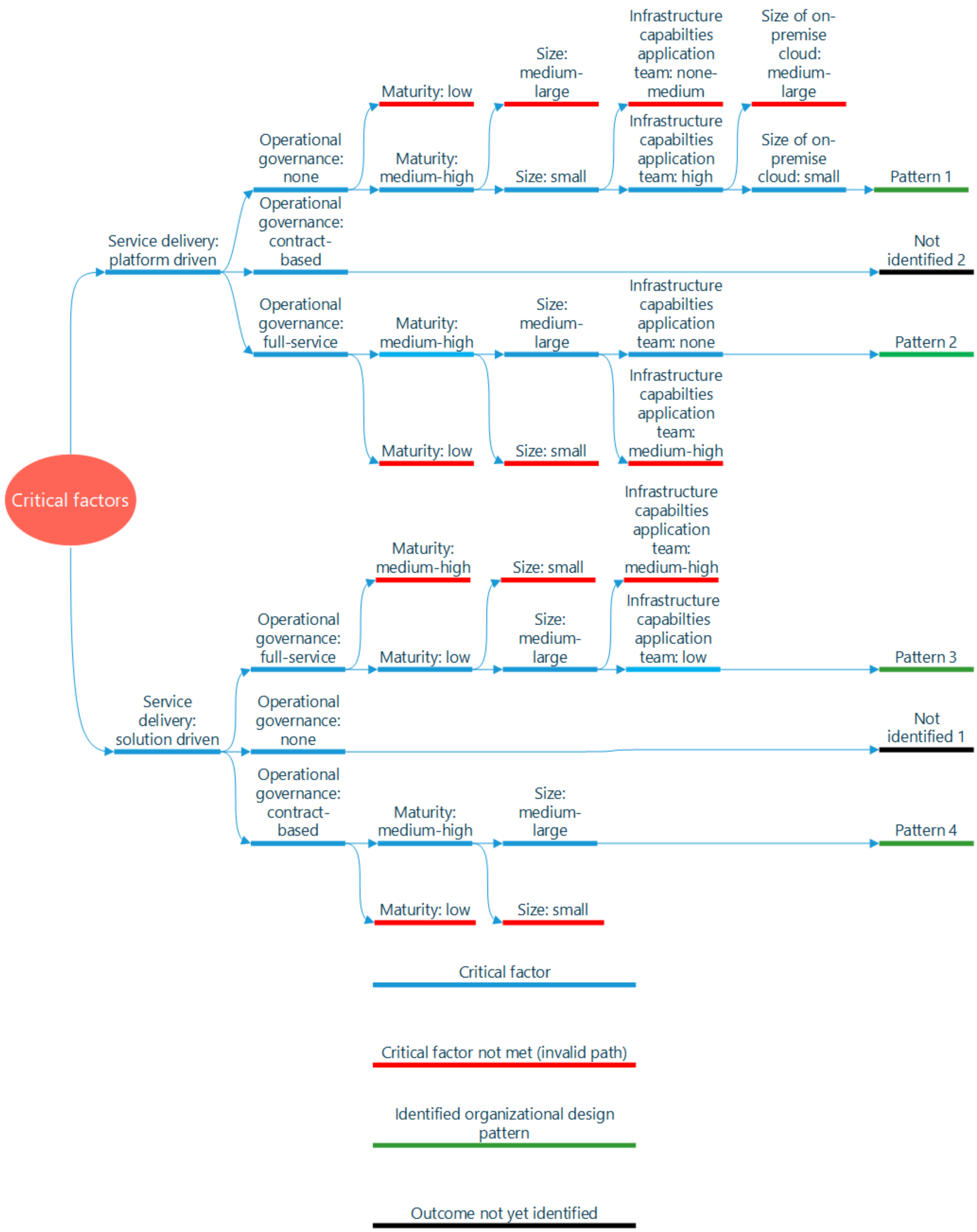

Figure 22: Decision tree critical factors

6.4 Scenario creation

The previous section described four patterns that IaaS providers can use to form their organizations. This section will present two scenarios for a service provider who applied the model using the decision tree. First, the critical factors will be used to determine the suitable pattern, after which the scenarios will be created.

The host organization for this thesis, will serve as one of the sample organizations for the study. The host organization is a large IT service provider within the Dutch government sector. A specific department within the host organization operates as an IaaS provider, and this part of the organization will be the focus of the analysis. The scenarios will only contain the visualization of the structure with a short explanation, as this is the most essential component of a pattern. Due to time constraints, other components could not be included.

In the visualization the service model of the NIST reference architecture [34] will be used. The NIST reference architecture describes a service model consisting of three layers: physical resources, resources abstraction & control and service. More information on the NIST reference architecture can be found in 2.1.4. For the four identified patterns in 6.2, the layers are used in the same visualization as the team configurations. However, applying the model to a specific organization made it apparent that this would create incomprehensible visualizations. Therefore, the scenarios will contain multiple visualizations. The first visualization will focus on the team configurations and the connections across teams, where a team delivers services to the team above. A second visualization will apply the NIST service model.

To create the scenarios, 6 employees of the host organization were asked to follow the decision tree of critical factors. Before using the decision tree, they were provided information on the patterns and the team configurations. Furthermore, a workshop was held with two employees to gather information on the current structure of the host organization as well as information on the structure of the host organization when choosing the platform-driven managed service provider pattern.

6.4.1 Critical factors

To determine which pattern would be suitable for the host organization 6 employees were asked to follow the decision tree of critical factors. All employees followed the same path, which ended in pattern 2 platform-driven managed service provider. Scenario 1 will focus on how the host organization will look when fully adopting the platform-driven managed service provider pattern. Furthermore, scenario 2 will focus on the current situation. Next the visualization of the scenario's will be given.

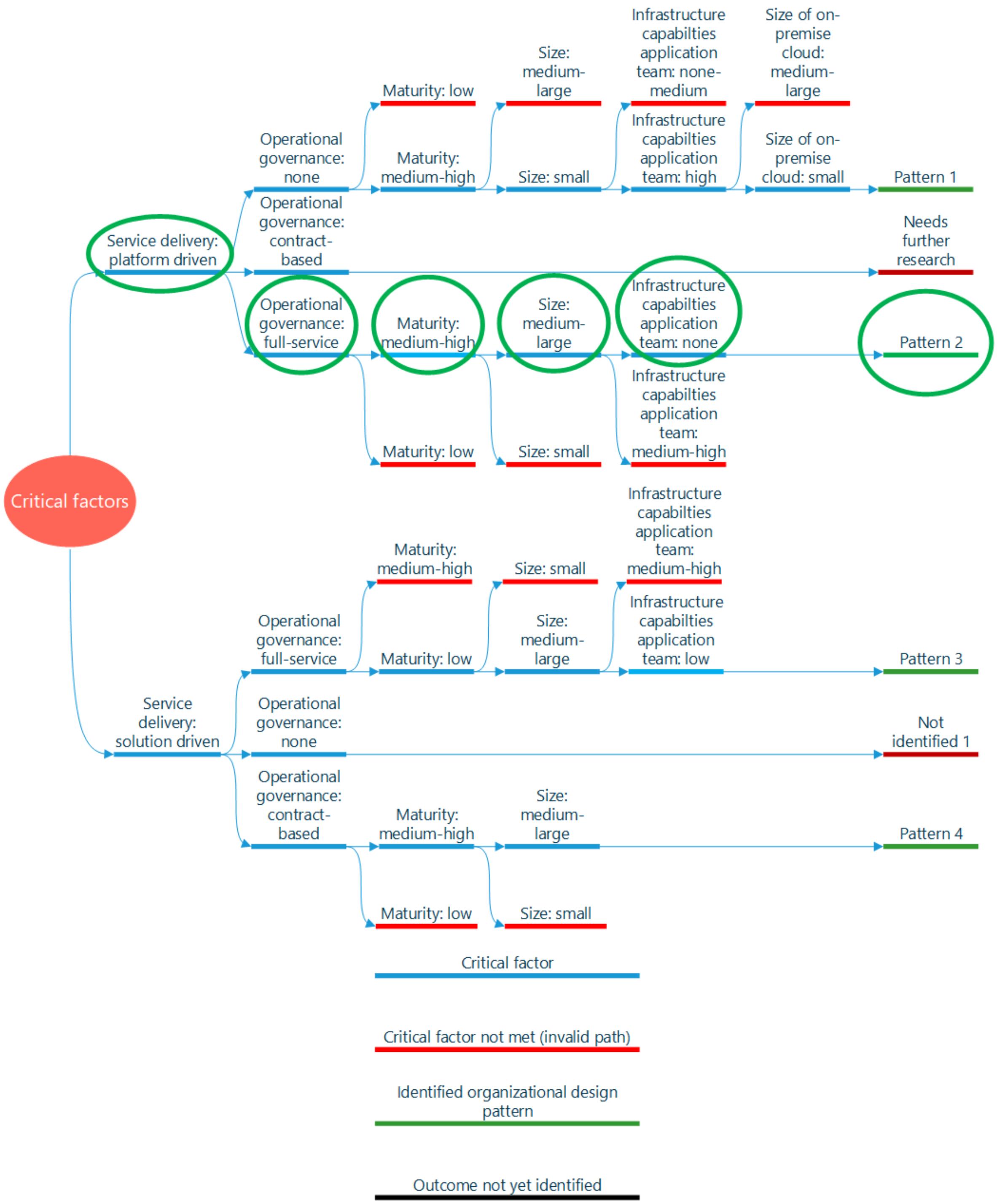

Figure 23: Critical factors of the host organization

6.4.2 Scenario 1 Platform-driven managed service provider

Scenario 1 visualizes (Figure 24) how the structure of the host organization will look like when fully adopting the platform-driven managed service provider pattern. Visualization 1 focuses on the teams and how they are related to each other; visualization 2 will focus on the place of the teams on the service model from NIST. It is important to note that this is a visualization of a future structure that the host organization has not yet implemented. Furthermore, there it is a wish to host multiple on-premises platforms; currently, the number of on-premises platforms is lower.

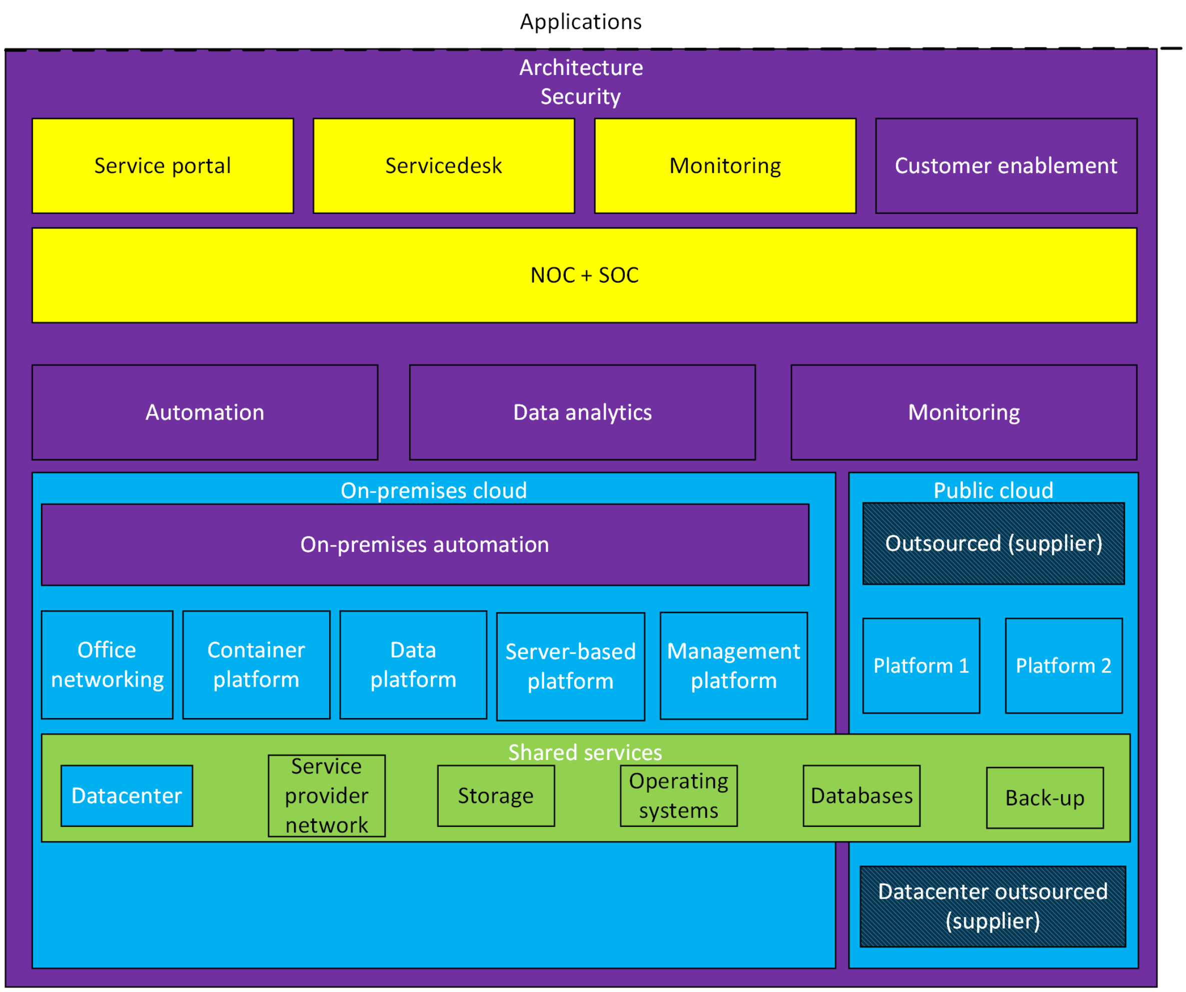

Figure 24: Scenario 1 Service driven managed service provider

Compared to pattern 2, the host organization wants to create a shared services layer in which specific technology-related components are offered to the platforms. The platform is built using its own technology stack but also relies on specific technologies provided by shared services. The platforms consume standardized components such as databases, storage, and operating system images. The platform will then be used to offer services to the client. The shared service layer aims to keep specific knowledge of technology components within the same team.

Shared services consist of the following teams: data center, service provider networking, storage, operating systems (OS), databases, and backup. This shared services layer is designed to support the multidisciplinary platform teams by providing centralized, reusable technology services. It should deliver services that are either consumed by multiple platforms or are too complex to decentralize due to the highly specialized and scarce technological expertise required. In addition, the shared services layer plays a key role in promoting standardization, as it enables the consistent delivery of technology components across different platforms. This standardization is essential for the host organization's strategic goal of achieving interoperable platforms, where workloads can be seamlessly transferred between platforms. For future work, it would be interesting to research the concept of shared services further and set up criteria for when a team cloud participate in shared services.

On top of the shared services are the platforms offered to the customer consuming the technology components. There are four on-premises platforms: the container, data, and server-based platforms are provided to the customer, whereas the management platform is used to maintain the platforms and their components. Furthermore, the host organization wants to offer public cloud platforms in the future.

A separate on-premises automation team is necessary for the on-premises platform. This team supports the other teams within the on-premises platform to implement IaaS multicloud capabilities, both in the cloud and on-premises environments. For the public cloud platform, this is done by the vendor. In addition, there is an overachring automation team focusing on provisioning the platforms' services automatically, so that when a customer requests a service in the service portal, provisioning is fully automatic. The data analytics and monitoring teams are responsible for the observability of the entire environment.

Customers can contact the networking operations center (NOC) for networking issues via the service desk; the SOC monitors security. On top of the NOC and SOC is the layer with direct customer contact. Customers can request their services in the service portal, after which they are provisioned with as much automation as possible. Moreover, the (skilled) service desk helps customers with their issues and redirects questions to the right team when necessary. In addition to this, the monitoring team monitors the provisioned services 24/7 and can be contacted when the service desk is closed. Lastly, the customer enablement team helps customers request the right services. This team is essential as the infrastructure capabilities of the application teams are low.

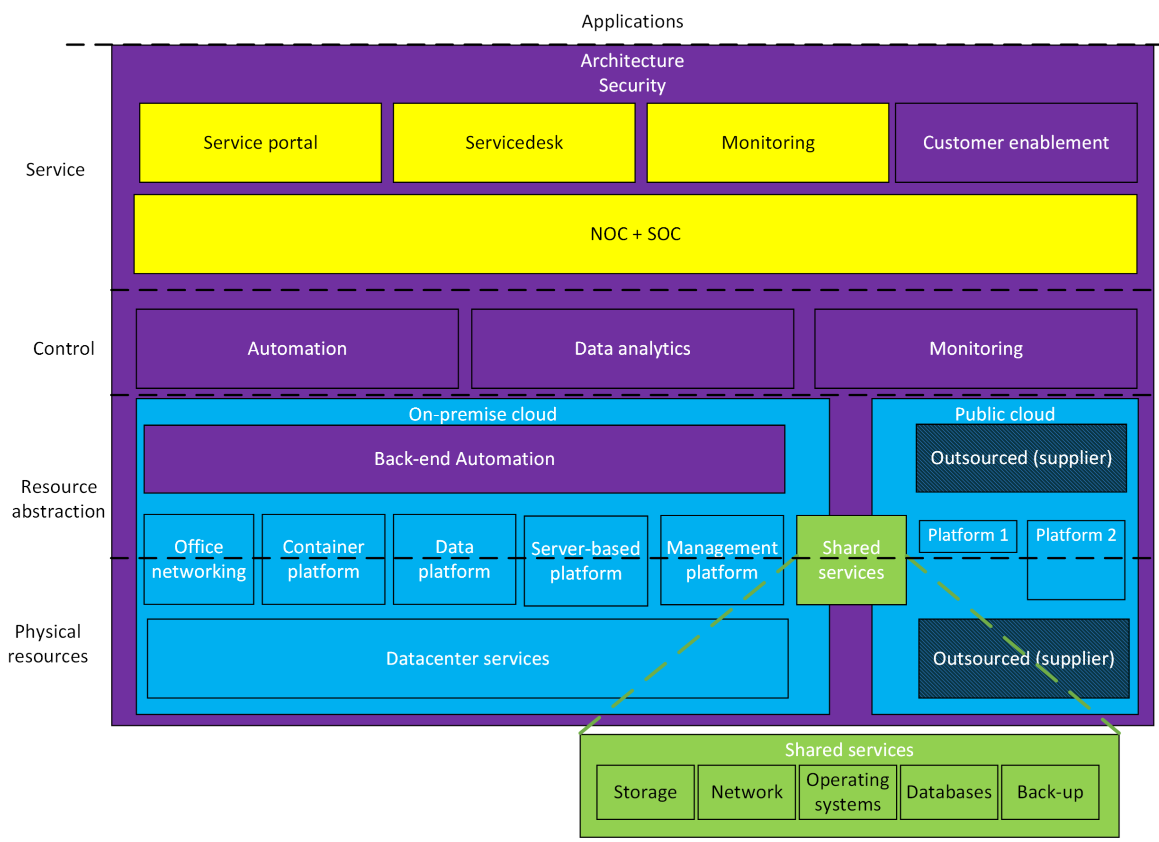

Figure 25: Scenario 1 Service driven managed service provider with NIST reference architecture layers

When plotting the layers of the service model from the NIST reference architecture on Figure 24, the shared services could not be visualized on the boundary of physical resources and resource abstraction while being below the platforms, as the shared services deliver services to the platforms. Therefore, a separate visualization is created. Note that in this visual, the layering in the structure no longer has a meaning.

The physical resources layer contains the team operating the data center, the platforms, and the shared services. The platform teams are responsible for offering physical resources as a platform to the customer. Doing this requires resource abstraction of physical resources, which makes platform teams accountable for both layers. Platform teams manage some physical resources themselves, such as part of the network and servers. The physical resources are then taken a layer higher using virtualization techniques, after which they can be offered as a platform to the customer.Furthermore, the storage, network, database, and backup teams use physical and virtual hardware components and offer these components as a service to the platform teams. This requires activities in the physical resources and resource abstraction layer. The operating systems team is solely involved in resource abstraction, and the office networking platform functions without dependencies on shared services.

Two scenarios are possible when considering the public cloud. The public cloud vendor might offer direct access to physical resources, and the platform would be involved in both the physical and the abstraction layers. Another option is that the public cloud vendor does not offer access to physical resources; the platform would only be involved in the resource abstraction layer.

The layers of the NIST service model are still applicable from an architectural point of view. However, the fact that teams are involved in multiple layers and offer sub-components as services to other teams shows that these lines are blurring from an organizational point of view.

6.4.3 Scenario 2 Current situation

Although the framework is left out of the scenario, it is essential to mention that in the current situation of the host organization, only a part of the organization uses the SAFe framework. A single agile release train (ART) "modernization" focuses on new platforms and services. The organization's non-agile part involves modernization and life cycle management activities. Consequently, there is a strong distinction between the activities and methods used in these two parts of the organization. Therefore, the ART will be visualized using a white dotted line.

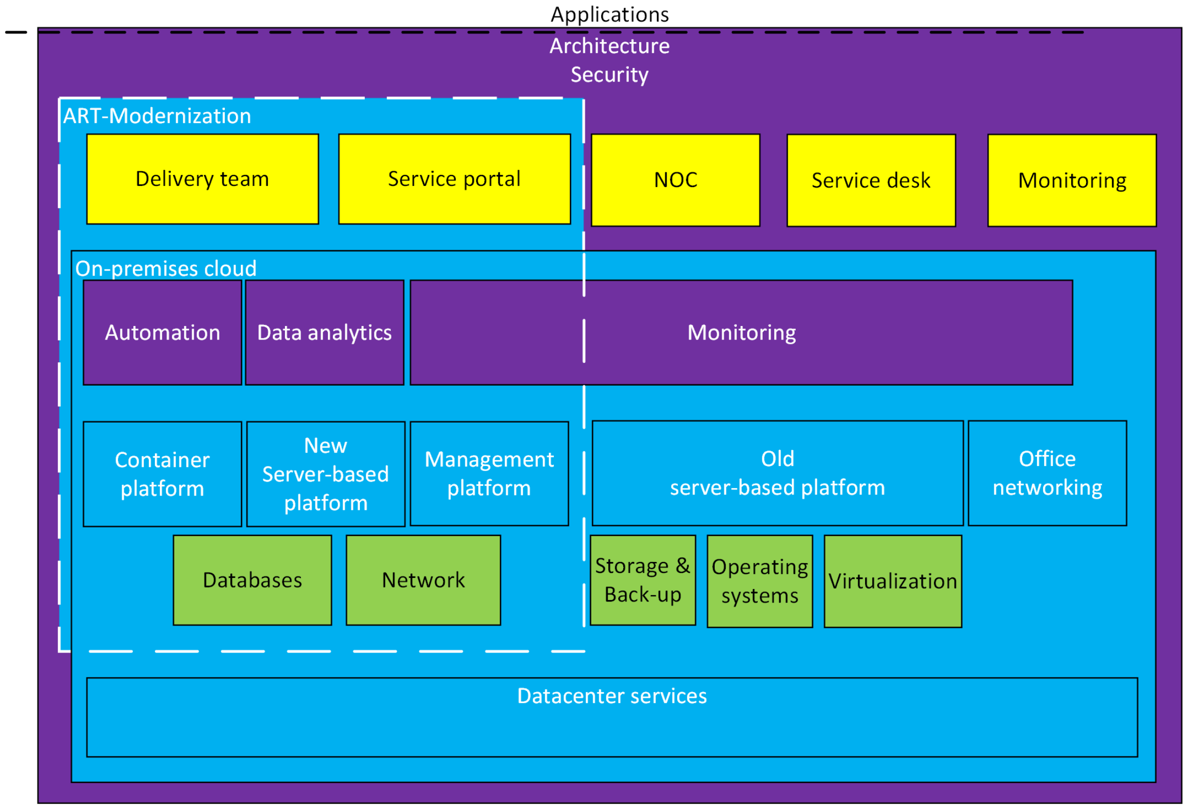

Figure 26: Scenario 2 current situation

There are two differences between the current situation and the first scenario. First, there is no public cloud yet. Second, the shared services are not yet bundled; in fact, the technology component teams are divided between ART modernization and the non-agile part of the organization. The technology components still deliver services to all platforms, apart from office networking; the office networking platform functions without dependencies on technology components.

There are three platforms in the ART modernization: a container platform, a new server-based platform, and a management platform, which is used to maintain platforms and their components. In the non-agile part of the organization, there is an old server-based platform, which is larger than the three platforms of the ART modernization.

The automation and data analytics teams solely focus on the new platforms in ART modernization. The monitoring team is responsible for the observability of the entire environment. All teams within the control layer have no customer contact and are focused on offering services internally.

The degree of automation in the current situation is lower than in the first scenario. Consequently, there is a need for a team coordinating service provisioning; this is the delivery team. The delivery team is responsible for requesting and clustering the separate services requested by the customer. Furthermore, there is a service portal, however, not all services can yet be requested in the service portal. The delivery and service portal team is solely responsible for provisioning services for new platforms; in fact, no new services are provided by the old platform. As in the first scenario, the non-skilled service desk can be contacted for issues, after which questions can be redirected to the right team if necessary. Furthermore, the network operations center is responsible for networking issues, and the monitoring team is responsible for monitoring provisioned services 24/7 and can be contacted when the service desk is closed.

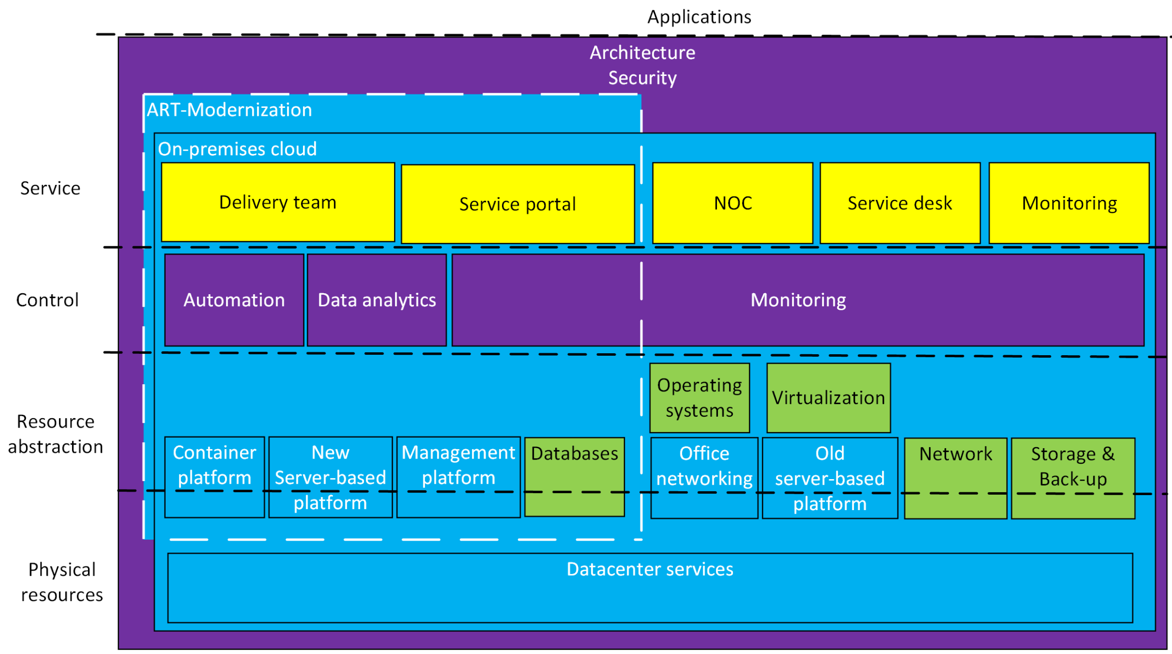

Figure 27: Scenario 2 current situation with NIST reference architecture

When plotting the layers of the service model from the NIST reference architecture in Figure 26, the same issue as for the first scenario occurs. The technology components cannot be visualized on the boundary of physical resources and resource abstraction while being below the platforms, as the technology components deliver services to the platforms. Therefore, a separate visualization is created. Note that in this visual, the layering of the structure no longer has a meaning.

Currently, the platforms are at the boundary of physical resources and the abstraction layer, as in the first scenario. The operating systems and virtualization team solely perform activities in the resource abstraction layer.

6.5 Reflection on the team topologies model

During the thesis, team configurations are used to distinguish and visualize different types of teams. In the analysis and discussion phase, it became apparent that the team configurations of the team topologies model lacked clarity. This section will reflect on this model.

The authors of the book team topologies describe four configurations: stream-aligned, complex-subsystem, enabling, and platform [49]. More information on the configurations can be found in 2.3.4. The authors state that software engineering organizations can be structured for fast flow when using the proposed team configurations. However, this thesis focuses on developing and operating infrastructure services instead of software engineering. Software engineering designs, implements, and operates applications that infrastructure providers host. An infrastructure provider designs, implements, provisions, and manages physical and virtual components such as networks, servers, storage, and other hardware components.

The team topologies model seemed suitable when writing the literature review. However, during the analysis phase, it became apparent that software engineering and IT infrastructure activities require different configurations than those described in the book. In addition to this, other critiques on the team configurations were found. This section will explain how the team configurations will look for IT infrastructure providers while also reflecting on team topologies model in a broader sense.

6.5.1 The stream-aligned configuration is difficult to operationalize

Stream-aligned teams are teams aligned to a stream of work. The authors describe a stream as "the continuous flow of work aligned to a business domain or organizational capability. Continuous flow requires clarity of purpose and responsibility so that multiple teams can coexist, each with their own flow of work."[49] A flow of work is defined as: "a single product or service, a single set of features, a single user journey, or a single user persona"[49]. These definitions are high-level, and no detailed definition of a 'stream' or 'flow of work' is given. This section will provides a more detailed definition of a stream and the steps in a stream for IT infrastructure providers to create a sharp definition of the stream-aligned configuration.

When looking at value streams for service organizations, two types of value streams can be identified: value streams for consumer goods and value streams for capital goods [88]. When compared to consumer goods services in capital goods are[88]:

- tailored to match each buyer's requirements;
- offer a wider range of services and increased depth of services;
- generate higher profit margins and recurring revenue over the long product life cycle;
- take place before, during and after the service is provisioned to the buyer.

In 6.1, two distinct types of service delivery were identified: platform-driven and solution-driven. These reflect fundamentally different value streams that shape how IaaS organizations structure and deliver their services. Solution-driven service delivery is characterized by customized offerings tailored to the specific needs of individual clients. This high level of customization implies that the services function as capital goods—unique, client-specific assets that require intensive collaboration and integration. In contrast, platform-driven service delivery emphasizes standardized, reusable offerings. These services are produced at scale and delivered to a broad user base with minimal customization, functioning as consumer goods—standardized and consumable services designed for self-service and efficiency.

This distinction has direct implications for the organizational design patterns identified in this thesis. The platform-driven self-managed platform and platform-driven managed service provider patterns align with consumer goods value streams, focusing on standardization, automation, and platform scalability. On the other hand, the solution-driven managed service provider and solution-driven Dev & Ops provider patterns support capital goods value streams, where close collaboration with clients and tailored solutions are key. These patterns reflect different operational priorities and competencies, underscoring how service delivery type shapes not only what is delivered, but how the organization must be structured to deliver it effectively.

Value streams within capital goods can be defined as: " the entire stream of activities required to deliver a finished product or service to the final consumer"[88, p.728]. The steps within the value stream for capital goods are[88]:

1. Manufacturing: concerning hardware and software, raw materials, and sub-assemblies are converted into physical components and subsystems.

2. Systems integration: designing and integrating components that have been developed internally and externally into a system that operates efficiently, while simultaneously coordinating the activities of the teams that developed the components.

3. Service provision: delivering the service to the final customer.

4. Operational services: maintaining and operating the provisioned service.

All value stream steps are executed for each product's capital goods. In contrast, for consumer goods, services are manufactured and integrated once, after which they can be provisioned and operated for multiple customers.

Now that a value stream is clearly defined, the role of stream-aligned teams within a value stream can be studied. The book states that a stream-aligned team works on the whole value chain, without requiring other teams to perform part of their work [49]. However, in none of the cases was any team working on the whole value chain. As shown in the results, solely the platform teams of cases A & B are involved in the platform's manufacturing, integration, provisioning, and operation. However, the platform team is not responsible for operating the services, showing that a single team can not work on the entire value stream. Therefore, a stream-aligned team is a team that is aligned to one or more steps of the value stream. The stream-aligned team remains the main team configuration in an organization where the other team configurations should reduce the cognitive load of the stream-aligned team.

6.5.2 Stream-aligned teams do not have to be multidisciplinary

The authors state that a stream-aligned team will not consist of specialists. Instead, the team members should be able to cover different areas of expertise to ensure that stream-aligned teams have all the capabilities necessary to complete their tasks. According to the authors' definition of streams, a stream-aligned team should be multidisciplinary and able to complete all steps in a value stream. With the new definition for IT infrastructure, a stream-aligned team should be multidisciplinary to complete one or more steps in a value stream.

Only in case L & N were multidisciplinary stream-aligned teams, while all other stream-aligned teams strongly focused on a single technology component, such as storage. Teams with a strong focus on a single technology component can still align to a step in the value stream. Thus, stream-aligned teams do not have to be multidisciplinary for IT infrastructure.

6.5.3 Sub-configurations of the stream-aligned configuration

The thesis found two new sub-configurations of the stream-aligned configuration: technology- and process-based teams. Definitions are given below.

Technology-based teams:
Throughout the thesis, teams that strongly focus on a single technology component are called technology-based teams. These teams can be found in cases: *C, D, F, G, H, I, J, K, & M.* Examples of technology-based teams are: storage, backup, networking, databases, and operating systems. In all cases, the technology-based teams form a platform or a stream, such as the on-premises cloud.

In all cases apart from K, M, the technology-based teams work according to DevOps. In other words, the technology-based teams are responsible for the manufacturing and operating parts of the system. This means they are aligned to steps one and four of the value stream and follow the definition of a stream-aligned team. Furthermore, the technology-based teams show that stream-aligned teams do not have to be multidisciplinary.

Process-based teams: In case L & N, there are multidisciplinary teams focused solely on development or operational tasks. Across cases K, M, the technology-based teams form a multidisciplinary department focusing on development or operational tasks. Throughout the thesis, teams or departments that focus on development or operational tasks are called process-based teams/departments.

The process-based configuration shows that stream-aligned teams can be multidisciplinary only when the customer offers development and operational services separately. This requires an entirely different business model than taken with a DevOps approach. Furthermore, the process-based configuration shows that stream-aligned teams do not have to be DevOps teams, even though the book says that stream-aligned teams should follow the you build it you own it principle [49].

Teams that do not focus on one technology, development tasks, or operational tasks but are aligned to one or more steps in the value stream are referred to as stream-aligned teams throughout the thesis.

6.5.4 No complex subsystem teams were identified due to the high level of knowledge required

The authors of the Team Topologies model state that a subsystem should only be classified as complex if it requires Ph.D.-level knowledge to develop or maintain [59]. Across all cases studied, none of the teams were composed exclusively of specialists with Ph.D.-level expertise. While IT infrastructure is often perceived as complex, no team in the case organizations formally qualifies as a complex subsystem team based on this definition. However, in practice, IT infrastructure engineers are typically subject matter experts who hold multiple certifications that show their expertise. Future research could explore whether such qualifications and domain-specific expertise justify reclassifying certain infrastructure teams as complex subsystem teams under an adapted interpretation of the model.

6.6 Threats to validity

In this research, the sampling bias is the main threat to validity. The external validity, construct validity and reliability have already been addressed in the methodology 4.1. The samples used in case studies can be decisive in shaping the outcomes. Therefore, an adequate sampling is an essential part of a research design [89]. A sampling bias is a consistent error that occurs when incorrect conclusions are drawn from a non-representative sample [90]. Especially cases with a small sample size are prone to a selection bias [91]. One of the strategies to increase representativeness is to use a random sample strategy. However, we were not in a position to do this due to the limited number of options available and the willingness of people to participate in an interview. This means sampling bias threatens the validity of the research, of which we were aware.

Using literal replications, which produce an anticipated result, and theoretical replications, which produce an anticipated different result [73], reduce the sampling bias when these cases differ on the key independent variable. Two or more of literal replications are sufficient to reduce the sampling bias [91]. In this research, two cases that offer limited operational support and can therefore not be fully considered as a managed service provider are included as theoretical replications. This is further described in the case selection strategy in 4.2. Similarities were found in these cases compared to the literal replications, even though these cases formed their own pattern.

Furthermore, all cases are located within the Netherlands; no other (organizational) cultures are considered in the research. This limits the external validity of the patterns to the same area. Performing the same study with the same case selection criteria might lead to new insights. Even though culture could influence organizational design, we expect most companies to be classified as part of one of the four identified patterns, i.e., we expect the results to be applicable outside the Netherlands as well.

In addition to the sample bias, the quality of the interviews is a key factor to the quality of the research. We considered this during the setup and execution of the interviews. A semi-structured interview was chosen, allowing for the interviewee's personal input, showing that the interview setup was not strictly controlled. The interviews took approximately 1.5 hours; this duration allowed for interviewees to feel at ease and provide insightful information. Furthermore, interviewees were guaranteed anonymity, resulting in candid conversations.

7 Conclusion

The goal of this research was to develop guidance on organizational redesign for MSPs offering IaaS through identifying and analyzing organizational design patterns for managed service providers operating within the infrastructure as a service (IaaS) domain. Organizational designs play a critical role in enabling IaaS providers to operate efficiently and provision scalable resources. As the IaaS sector increasingly shifts toward standardization and automation, effective organizational structures become essential. The identified organizational design patterns offer practical guidance for structuring teams and processes. A multiple case study was conducted across 14 organizations to identify the organizational design patterns. The cases represented a range of medium to large organizations. In some cases, the interviewees were part of an IT service provider offering IaaS to clients, while in others, the internal IaaS department within an organization was studied. These internal departments are responsible for delivering infrastructure services to internal application teams. The cases also included a mix of governmental and non-governmental organizations, providing a diverse perspective on how IaaS is structured and managed across different sectors. In total, 14 interviews were held, leading to 1205 minutes of recordings and 1123 pages of transcripts.

The main contributions of the research are: (1) a description of four organizational design patterns, (2) the configurations of the teams within an organizational pattern and (3) the critical factors that characterize an organizational pattern.
First, four organizational design patterns were identified: (1) platform-driven self-managed platform organization, (2) platform-driven managed service provider organization, (3) solution-driven managed service provider organization, and (4) solution-driven Dev & Ops provider organization. An organizational design pattern helps shape an organization, covering aspects like organizational structure and processes. Each of the identified organizational design patterns show distinct characteristics and adopt a different approach to standardization and operational governance. The third pattern currently lags behind in terms of standardization and automation and is likely to evolve toward the second pattern, potentially becoming non-existent over time. In contrast, the remaining patterns play a central role in enabling scalable managed IaaS. They provide actionable guidance for organizations aiming to advance the standardization and automation of IaaS offerings.

Second, the configurations of the teams within an organizational pattern were examined. The team topologies model was used to define the role of teams within an organizational pattern. This model provides a useful framework by defining four team configurations: stream-aligned, enabling, platform, and complex subsystem teams. However, during the analysis of case data, it became apparent that categorizing the case teams using only the original four configurations was challenging. This was mainly due to the fact that the stream-aligned configuration is difficult to operationalize. As a result, two new sub-configurations of the stream-aligned team were identified and added: the technology-based and process-based configurations. These sub-configurations focus respectively on a technology domain or development/operations. These new configurations appear to be specific to IT infrastructure providers and offer valuable guidance to IaaS organizations on how to structure teams and define their roles within the organization.

Third, six critical factors were identified that characterize an organizational design pattern: (1) service delivery, (2) operational governance, (3) maturity, (4) organization size, (5) infrastructure capabilities within application teams, and (6) size of on-premises cloud. The critical factors form a decision tree that can be used to determine which pattern applies to an organization.

We may conclude, the infrastructure as a service (IaaS) sector is expected to continue its rapid growth, outpacing the broader cloud market. Alongside this growth, managed service providers (MSPs) operating within the IaaS domain are also expanding, driven by increasing demand and rising complexity. While the early adoption of IaaS was often simpler and more uniform, today's landscape includes diverse hybrid environments and a broader range of technologies. As the sector continues to shift toward greater standardization and automation, MSPs face new challenges in delivering consistent, scalable, and efficient services. In this evolving context, organizational design patterns offer guidance by providing information on how to structure the organization, teams, and processes of IaaS providers. Ultimately, the proposed organizational designs are a critical enabler of standardization and automation in IaaS, serving as a guiding framework for organizations seeking to scale and automate their IaaS provisioning.

7.1 Future work

This research is an exploratory multiple case study that provided an organizational design framework for MSPs offering automatic IaaS. This section will present the areas that are still open for research or require further research.

First, two patterns have not been identified in this research. Future studies could focus on exploring these patterns by incorporating criteria specific to these patterns in the case selection criteria. The criteria of these patterns are given for service delivery and operational governance.

Second, the research scope could be expanded to include more cases, which would help reinforce the identified patterns and potentially uncover new ones. Additionally, the scope could be broadened to include an international perspective.

Third, the technology component and process configuration identified in this research require further research. The study could focus on further defining these configurations and their advantages. Furthermore, for these configurations it is still to be proven that they prevent silos. The researcher assumes the configurations are specific to IT infrastructure and recommends further research in this field.

Fourth, the implementation of the patterns can be researched. This research focuses on which patterns are implemented in cases and identifies patterns; however, it does not include information on the implementation of a pattern. Furthermore, it is also interesting to research how transitioning from the solution-driven managed service provider to the platform-driven managed service provider would look.

Fifth, the team interaction modes can be explored in greater detail in future research. While this study focuses on high-level organizational design patterns, it does not deeply examine how teams collaborate, coordinate, or facilitate each other's work in practice. For example, the collaboration interaction mode between stream-aligned teams, offers a promising direction for more granular analysis.

Lastly, future research could focus on the concept of the shared services stream suggested in the scenario creation. The Shared services stream should deliver reusable infrastructure products to platforms.

References

[1] Miguel Caballer et al. “Dynamic Management of Virtual Infrastructures”. en. In: *Journal of Grid Computing* 13.1 (Mar. 2015), pp. 53–70. ISSN: 1572-9184. DOI: 10.1007/s10723-014-9296-5. URL: https://doi.org/10.1007/s10723-014-9296-5 (visited on 05/30/2025).

[2] Sebastian Floerecke, Florian Felix Röck, and Franz Lehner. “Success-Driving Business Model Characteristics of IaaS Providers – Towards a Quantitative Evaluation”. In: *International Journal on Cloud Computing: Services and Architecture* 10.5 (Oct. 2020), pp. 1–10. ISSN: 22316663. DOI: 10.5121/ijccsa.2020.10501. URL: https://aircconline.com/ijccsa/V10N5/10520ijccsa01.pdf (visited on 02/19/2024).

[3] Richard M. Burton and Børge Obel. “The science of organizational design: fit between structure and coordination”. en. In: *Journal of Organization Design* 7.1 (Mar. 2018), p. 5. ISSN: 2245-408X. DOI: 10.1186/s41469-018-0029-2. URL: https://doi.org/10.1186/s41469-018-0029-2 (visited on 04/30/2025).

[4] Michael Armbrust et al. “A view of cloud computing”. en. In: *Communications of the ACM* 53.4 (Apr. 2010), pp. 50–58. ISSN: 0001-0782, 1557-7317. DOI: 10.1145/1721654.1721672. URL: https://dl.acm.org/doi/10.1145/1721654.1721672 (visited on 05/30/2025).

[5] Reimar Weissbach et al. “The Complex and Opaque Cloud Ecosystem: Recommendations for IaaS Providers for a Successful Positioning in the Ecosystem”. In: *Advances in Systems Analysis, Software Engineering, and High Performance Computing*. Ed. by Saikat Gochhait, David Tawei Shou, and Sabiha Fazalbhoy. IGI Global, 2020, pp. 134–158. ISBN: 9781799812944 9781799812968. DOI: 10.4018/978-1-7998-1294-4.ch007. URL: http://services.igi-global.com/resolvedoi/resolve.aspx?doi=10.4018/978-1-7998-1294-4.ch007 (visited on 05/30/2025).

[6] Gartner. *Forecast Analysis: Sovereign Cloud IaaS, Worldwide*. English. Document. Oct. 2024. URL: https://www.gartner.com/en/documents/5863580#:~:text=The%20%2437%20billion%20sovereign%20cloud,operational%20independence%20and%20technological%20autonomy. (visited on 05/03/2025).

[7] Gartner. *Gartner Forecasts Worldwide Public Cloud End-User Spending to Total $723 Billion in 2025*. English. Press Release. May 2024. URL: https://www.gartner.com/en/newsroom/press-releases/2024-11-19-gartner-forecasts-worldwide-public-cloud-end-user-spending-to-total-723-billion-dollars-in-2025 (visited on 05/03/2025).

[8] Peter Mell and Tim Grance. *The NIST definition of cloud computing*. 2011. URL: https://nvlpubs.nist.gov/nistpubs/legacy/sp/nistspecialpublication800-145.pdf (visited on 02/07/2024).

[9] Gartner. *ManagedServiceProvider (MSP) - Gartner Information Technology Glossary*. en. URL: https://www.gartner.com/en/information-technology/glossary/msp-management-service-provider (visited on 02/19/2024).

[10] Narayanan Kumbakara. “Managed IT services: the role of IT standards”. en. In: *Information Management & Computer Security* 16.4 (Oct. 2008), pp. 336–359. ISSN: 0968-5227. DOI: 10.1108/09685220810908778. URL: https://www.emerald.com/insight/content/doi/10.1108/09685220810908778/full/html (visited on 04/30/2025).

[11] Amy Kates and Jay R. Galbraith. *Designing Your Organization: Using the STAR Model to Solve 5 Critical Design Challenges*. en. Google-Books-ID: w8JeLUIbtoEC. John Wiley & Sons, Dec. 2010. ISBN: 9781118047514.

[12] Gregory Vial. “Understanding digital transformation: A review and a research agenda”. en. In: *The Journal of Strategic Information Systems* 28.2 (June 2019), pp. 118–144. ISSN: 09638687. DOI: 10.1016/j.jsis.2019.01.003. URL: https://linkinghub.elsevier.com/retrieve/pii/S0963868717302196 (visited on 02/27/2024).

[13] Sascha Kraus et al. “Digital Transformation: An Overview of the Current State of the Art of Research”. en. In: *SAGE Open* 11.3 (July 2021), p. 215824402110475. ISSN: 2158-2440, 2158-2440. DOI: 10.1177/21582440211047576. URL: http://journals.sagepub.com/doi/10.1177/21582440211047576 (visited on 02/27/2024).

[14] Hasmat Malik, Gopal Chaudhary, and Smriti Srivastava. "Digital transformation through advances in artificial intelligence and machine learning". In: *Journal of Intelligent & Fuzzy Systems* 42.2 (Jan. 2022). Ed. by Hasmat Malik, Gopal Chaudhary, and Smriti Srivastava, pp. 615–622. ISSN: 10641246, 18758967. DOI: 10.3233/JIFS-189787. URL: https://www.medra.org/servlet/aliasResolver?alias=iospress&doi=10.3233/JIFS-189787 (visited on 02/27/2024).

[15] Thomas Hess et al. *Options for Formulating a Digital Transformation Strategy. — MIS Quarterly Executive — EBSCOhost*. en. June 2016. URL: https://openurl.ebsco.com/contentitem/gcd:115879199?sid=ebsco:plink:crawler&id=ebsco:gcd:115879199 (visited on 02/27/2024).

[16] Ntandoyethu S.M. Mhlungu, Jeff Y.J. Chen, and Peter Alkema. "The underlying factors of a successful organisational digital transformation". en. In: *SA Journal of Information Management* 21.1 (Apr. 2019). ISSN: 1560-683X, 2078-1865. DOI: 10.4102/sajim.v21i1.995. URL: http://www.sajim.co.za/index.php/SAJIM/article/view/995 (visited on 02/27/2024).

[17] Oksana Makovoz and Lysenko Segii. "Evolution of digital transformations in IT companies — Proceedings of London International Conferences". en-US. In: (Mar. 2024). ISSN: 2977-1870. DOI: 10.31039/plic.2024.9.199. URL: https://londonic.uk/js/index.php/plic/article/view/199 (visited on 03/26/2024).

[18] Blesson Varghese. *History of the cloud — BCS*. en. 2017. URL: https://www.bcs.org/articles-opinion-and-research/history-of-the-cloud/ (visited on 03/27/2024).

[19] Omid Givehchi, Henning Trsek, and Juergen Jasperneite. "Cloud computing for industrial automation systems — A comprehensive overview". In: *2013 IEEE 18th Conference on Emerging Technologies & Factory Automation (ETFA)*. ISSN: 1946-0759. Sept. 2013, pp. 1–4. DOI: 10.1109/ETFA.2013.6648080. URL: https://ieeexplore.ieee.org/abstract/document/6648080 (visited on 03/27/2024).

[20] Xun Xu. "From cloud computing to cloud manufacturing". en. In: *Robotics and Computer-Integrated Manufacturing* 28.1 (Feb. 2012), pp. 75–86. ISSN: 07365845. DOI: 10.1016/j.rcim.2011.07.002. URL: https://linkinghub.elsevier.com/retrieve/pii/S0736584511000949 (visited on 03/27/2024).

[21] Tyko Kieskamp. *From Cloud-native to Hybrid and back again: Picnic's on-premises computing journey*. en-US. Feb. 2024. URL: https://picnic.app/careers/from-cloud-native-to-hybrid-and-back-again-picnics-on-premises-computing-journey (visited on 03/27/2024).

[22] Xia and King. "Determinants of Organizational IT Infrastructure Capabilities : An Empirical Study by Weidong". In: 2002. URL: https://www.semanticscholar.org/paper/Determinants-of-Organizational-IT-Infrastructure-%3A-Carlson/26c056f7e10f557950bc61fef9576c6c07c85472 (visited on 03/28/2024).

[23] M. Broadbent, P. Weill, and B. S. Neo. "Strategic context and patterns of IT infrastructure capability". In: *The Journal of Strategic Information Systems* 8.2 (June 1999), pp. 157–187. ISSN: 0963-8687. DOI: 10.1016/S0963-8687(99)00022-0. URL: https://www.sciencedirect.com/science/article/pii/S0963868799000220 (visited on 03/28/2024).

[24] Sjaak Laan. *IT Infrastructure Architecture: Infrastructure building blocks and concepts*. Lulu.com, Oct. 2011. ISBN: 9781447881285.

[25] Gartner. *IAASGrowth-Gartner Says Worldwide IaaS Public Cloud Services Revenue Grew 30% in 2022, Exceeding $100 Billion for the First Time*. IAAS growth. July 2023. URL: https://www.gartner.com/en/newsroom/press-releases/2023-07-18-gartner-says-worldwide-iaas-public-cloud-services-revenue-grew-30-percent-in-2022-exceeding-100-billion-for-the-first-time (visited on 03/28/2024).

[26] Joel Gibson et al. "Benefits and challenges of three cloud computing service models". In: *2012 Fourth International Conference on Computational Aspects of Social Networks (CASoN)*. Nov. 2012, pp. 198–205. DOI: 10.1109/CASoN.2012.6412402. URL: https://ieeexplore.ieee.org/abstract/document/6412402 (visited on 02/07/2024).

[27] Sunilkumar S. Manvi and Gopal Krishna Shyam. "Resource management for Infrastructure as a Service (IaaS) in cloud computing: A survey". en. In: *Journal of Network and Computer Applications* 41 (May 2014), pp. 424–440. ISSN: 10848045. DOI: 10.1016/j.jnca.2013.10.004. URL: https://linkinghub.elsevier.com/retrieve/pii/S1084804513002099 (visited on 02/05/2024).

[28] Arash Deldari and Alireza Salehan. "A survey on preemptible IaaS cloud instances: challenges, issues, opportunities, and advantages". en. In: *Iran Journal of Computer Science* 4.3 (Sept. 2021), pp. 1–24. ISSN: 2520-8438, 2520-8446. DOI: 10.1007/s42044-020-00071-1. URL: https://link.springer.com/10.1007/s42044-020-00071-1 (visited on 02/22/2024).

[29] Jia-Bin Yuan et al. "Building an intelligent provisioning engine for IaaS cloud computing services". In: *2011 13th Asia-Pacific Network Operations and Management Symposium*. Sept. 2011, pp. 1–6. DOI: 10.1109/APNOMS.2011.6077009. URL: https://ieeexplore.ieee.org/document/6077009/ (visited on 02/07/2024).

[30] Jiangtao Zhang, Hejiao Huang, and Xuan Wang. "Resource provision algorithms in cloud computing: A survey". In: *Journal of Network and Computer Applications* 64 (Apr. 2016), pp. 23–42. ISSN: 1084-8045. DOI: 10.1016/j.jnca.2015.12.018. URL: https://www.sciencedirect.com/science/article/pii/S1084804516000643 (visited on 02/07/2024).

[31] Borja Sotomayor et al. "Virtual Infrastructure Management in Private and Hybrid Clouds". In: *IEEE Internet Computing* 13.5 (Sept. 2009), pp. 14–22. ISSN: 1941-0131. DOI: 10.1109/MIC.2009.119. URL: https://ieeexplore.ieee.org/document/5233608/ (visited on 02/19/2024).

[32] Johannes Kirschnick et al. "Toward an architecture for the automated provisioning of cloud services". In: *IEEE Communications Magazine* 48.12 (Dec. 2010), pp. 124–131. ISSN: 1558-1896. DOI: 10.1109/MCOM.2010.5673082. URL: https://ieeexplore.ieee.org/document/5673082/ (visited on 04/02/2024).

[33] Nishant Kumar Singh et al. "Automated provisioning of application in IAAS cloud using Ansible configuration management". In: *2015 1st International Conference on Next Generation Computing Technologies (NGCT)*. Dehradun, India: IEEE, Sept. 2015, pp. 81–85. ISBN: 9781467368094. DOI: 10.1109/NGCT.2015.7375087. URL: http://ieeexplore.ieee.org/document/7375087/ (visited on 02/27/2024).

[34] Fang Liu et al. *NIST Cloud Computing Reference Architecture*. English. Sept. 2011. URL: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication500-292.pdf (visited on 02/19/2025).

[35] Richard M. Burton and Borge Obel. *Strategic Organizational Diagnosis and Design: The Dynamics of Fit*. en. Google-Books-ID: 6HlmO0Ho3aYC. Springer Science & Business Media, 2004. ISBN: 9781402076855.

[36] André Hanelt et al. "A Systematic Review of the Literature on Digital Transformation: Insights and Implications for Strategy and Organizational Change". en. In: *Journal of Management Studies* 58.5 (July 2021), pp. 1159–1197. ISSN: 0022-2380, 1467-6486. DOI: 10.1111/joms.12639. URL: https://onlinelibrary.wiley.com/doi/10.1111/joms.12639 (visited on 02/27/2024).

[37] Jack Greenfield and Keith Short. "Software factories: assembling applications with patterns, models, frameworks and tools". en. In: *Companion of the 18th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications*. Anaheim CA USA: ACM, Oct. 2003, pp. 16–27. ISBN: 9781581137514. DOI: 10.1145/949344.949348. URL: https://dl.acm.org/doi/10.1145/949344.949348 (visited on 04/30/2025).

[38] Michael A. Cusumano. *Japan's software factories: a challenge to U.S. management*. USA: Oxford University Press, Inc., Feb. 1991. ISBN: 9780195062168.

[39] Rohit Bhapkar et al. *Scaling a transformative culture through a digital factory — McKinsey*. 2017. URL: https://www.mckinsey.com/capabilities/mckinsey-digital/our-insights/scaling-a-transformative-culture-through-a-digital-factory (visited on 02/07/2024).

[40] *Digital Factory: The Energy of Change — Accenture*. en-US. 2023. URL: https://www.accenture.com/us-en/case-studies/utilities/factoria-digital-iberdrola (visited on 02/07/2024).

[41] *What is a Digital Factory*. en. URL: https://www.opentext.com/what-is/digital-factory (visited on 02/07/2024).

[42] *Digital Factory: Scaling Up Innovation*. en-US. URL: https://www.accenture.com/us-en/case-studiesnew/banking/eurobank (visited on 02/07/2024).

[43] Corvin Meyer-Blankart. "A new way to design digital innovation units: a model for growing new ways of working in established enterprises". In: (July 2020), pp. 1–302. URL: https://ediss.sub.uni-hamburg.de/handle/ediss/8530 (visited on 02/07/2024).

[44] LMU Munich, Institute for Information Systems and New Media, Munich, Germany et al. "Embedding Digital Innovations in Organizations: A Typology for Digital Innovation Units". In: *WI2020 Zentrale Tracks*. GITO Verlag, Mar. 2020, pp. 780–795. ISBN: 9783955453350. DOI: 10.30844/wi_2020_h1-barthel. URL: https://library.gito.de/2021/07/13/wi2020-zentrale-tracks-57/ (visited on 02/07/2024).

[45] David Nadler, Michael Tushman, and Mark B. Nadler. *Competing by Design: The Power of Organizational Architecture*. en. Google-Books-ID: 0U0SDAAAQBAJ. Oxford University Press, USA, July 1997. ISBN: 9780195099171.

[46] Keith Merron. *Riding the wave: designing your organization's architecture for enduring success*. English. New York: Van Nostrand Reinhold, 1995. ISBN: 9780442018030.

[47] Alan Gutterman. "Organizational Design". en. In: *SSRN Electronic Journal* (2023). ISSN: 1556-5068. DOI: 10.2139/ssrn.4541482. URL: https://www.ssrn.com/abstract=4541482 (visited on 04/22/2024).

[48] J.D. Herbsleb and R.E. Grinter. "Architectures, coordination, and distance: Conway's law and beyond". In: *IEEE Software* 16.5 (Oct. 1999), pp. 63–70. ISSN: 07407459. DOI: 10.1109/52.795103. URL: http://ieeexplore.ieee.org/document/795103/ (visited on 04/02/2024).

[49] Matthew Skelton and Manuel Pais. *Team topologies: organizing business and technology teams for fast flow*. First edition. Portland, OR: IT Revolution, 2019. ISBN: 9781942788843 9781942788829 9781942788836.

[50] Hossein Vatanpour, Atoosa Khorramnia, and Naghmeh Forutan. "Silo Effect a Prominence Factor to Decrease Efficiency of Pharmaceutical Industry". In: *Iranian Journal of Pharmaceutical Research : IJPR* 12.Suppl (2013), pp. 207–216. ISSN: 1735-0328. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3813367/ (visited on 03/13/2024).

[51] Mahlomola Motingoe and Josef J. Langerman. "New Organisational Models That Break Silos in Organisations to Enable Software Delivery Flow". In: *2019 International Conference on System Science and Engineering (ICSSE)*. ISSN: 2325-0925. July 2019, pp. 341–348. DOI: 10.1109/ICSSE.2019.8823257. URL: https://ieeexplore.ieee.org/document/8823257/ (visited on 03/13/2024).

[52] Frans Cilliers and Henk Greyvenstein. "The impact of silo mentality on team identity: An organisational case study". In: *SA Journal of Industrial Psychology* 38.2 (Jan. 2012), 9 pages. ISSN: 2071-0763, 0258-5200. DOI: 10.4102/sajip.v38i2.993. URL: http://sajip.co.za/index.php/sajip/article/view/993 (visited on 03/13/2024).

[53] Fabio Bento, Marco Tagliabue, and Flora Lorenzo. "Organizational Silos: A Scoping Review Informed by a Behavioral Perspective on Systems and Networks". en. In: *Societies* 10.3 (July 2020), p. 56. ISSN: 2075-4698. DOI: 10.3390/soc10030056. URL: https://www.mdpi.com/2075-4698/10/3/56 (visited on 03/13/2024).

[54] Mirghani Mohamed, Michael Stankosky, and Arthur Murray. "Applying knowledge management principles to enhance cross-functional team performance". In: *Journal of Knowledge Management* 8.3 (Jan. 2004), pp. 127–142. ISSN: 1367-3270. DOI: 10.1108/13673270410541097. URL: https://doi.org/10.1108/13673270410541097 (visited on 03/13/2024).

[55] Karen L. Spencer. "The Wisdom of Teams: Creating The High-Performance OrganizationThe Wisdom of Teams: Creating The High-Performance Organization By KatzenbachJon R. and SmithDouglas K. Boston, MA: Harvard Business School Press, 1993, 291 pages, $24.95". en. In: *Academy of Management Perspectives* 7.3 (Aug. 1993), pp. 100–102. ISSN: 1558-9080, 1943-4529. DOI: 10.5465/ame.1993.9411302368. URL: http://journals.aom.org/doi/10.5465/ame.1993.9411302368 (visited on 03/08/2024).

[56] Michael A. West, Dean Tjosvold, and Ken G. Smith, eds. *The essentials of teamworking: international perspectives*. Chichester, England ; Hoboken, NJ: John Wiley & Sons, Inc, 2005. ISBN: 9780470015483.

[57] Michael A. West, Dean Tjosvold, and Ken G. Smith. *International Handbook of Organizational Teamwork and Cooperative Working*. en. Google-Books-ID: Bkf1q4O0T0kC. John Wiley & Sons, May 2008. ISBN: 9780470864838.

[58] Bruce W. Tuckman. "Developmental sequence in small groups." en. In: *Psychological Bulletin* 63.6 (June 1965), pp. 384–399. ISSN: 1939-1455, 0033-2909. DOI: 10.1037/h0022100. URL: https://doi.apa.org/doi/10.1037/h0022100 (visited on 03/11/2024).

[59] Matthew Skelton and Manuel Pais. *What are the core team types in Team Topologies?* en-GB. URL: https://teamtopologies.com/key-concepts-content/what-are-the-core-team-types-in-team-topologies (visited on 03/11/2025).

[60] Yucong Duan et al. "Everything as a Service (XaaS) on the Cloud: Origins, Current and Future Trends". In: *2015 IEEE 8th International Conference on Cloud Computing*. New York City, NY, USA: IEEE, June 2015, pp. 621–628. ISBN: 9781467372879. DOI: 10.1109/CLOUD.2015.88. URL: http://ieeexplore.ieee.org/document/7214098/ (visited on 03/12/2024).

[61] Gartner. *Definition of Business Model - Gartner Finance Glossary*. en. URL: https://www.gartner.com/en/finance/glossary/business-model (visited on 02/06/2024).

[62] Sebastian Floerecke and Franz Lehner. "Success-driving business model characteristics of IaaS and PaaS providers". In: *International Journal on Cloud Computing: Services and Architecture* 08.06 (Dec. 2018), pp. 01–22. ISSN: 22316663, 22315853. DOI: 10.5121/ijccsa.2018.8601. URL: http://aircconline.com/ijccsa/V8N6/8618ijccsa01.pdf (visited on 02/05/2024).

[63] Sebastian Floerecke and Franz Lehner. "Dominant Business Model Patterns of Regional IaaS Providers – An Exploratory Multiple-Case Study". en. In: *Economics of Grids, Clouds, Systems, and Services* 11819 (2019). Ed. by Karim Djemame et al., pp. 140–153. DOI: 10.1007/978-3-030-36027-6_12. URL: http://link.springer.com/10.1007/978-3-030-36027-6_12 (visited on 02/06/2024).

[64] Sebastian Floerecke and Franz Lehner. "Business Model Characteristics for Local IaaS Providers for Counteracting the Dominance of the Hyperscalers". en. In: *Economics of Grids, Clouds, Systems, and Services* 11113 (2019). Ed. by Massimo Coppola et al., pp. 137–150. DOI: 10.1007/978-3-030-13342-9_12. URL: http://link.springer.com/10.1007/978-3-030-13342-9_12 (visited on 02/06/2024).

[65] Barry Berman. "Referral marketing: Harnessing the power of your customers". en. In: *Business Horizons* 59.1 (Jan. 2016), pp. 19–28. ISSN: 00076813. DOI: 10.1016/j.bushor.2015.08.001. URL: https://linkinghub.elsevier.com/retrieve/pii/S0007681315000956 (visited on 02/06/2024).

[66] Roger Evered. "So what is strategy?" en. In: *Long Range Planning* 16.3 (June 1983), pp. 57–72. ISSN: 00246301. DOI: 10.1016/0024-6301(83)90032-8. URL: https://linkinghub.elsevier.com/retrieve/pii/0024630183900328 (visited on 02/27/2024).

[67] Anna Wiedemann, Andy Weeger, and Heiko Gewald. "Organizational Structure vs. Capabilities: Examining Critical Success Factors for Managing IT Service Delivery". In: *2015 48th Hawaii International Conference on System Sciences*. HI, USA: IEEE, Jan. 2015, pp. 4564–4574. ISBN: 9781479973675. DOI: 10.1109/HICSS.2015.544. URL: http://ieeexplore.ieee.org/document/7070364/ (visited on 03/19/2024).

[68] Mark Saunders, Philip Lewis, and Adrian Thornhill. *Research methods for business students*. eng. Ninth edition. Harlow: Pearson, 2023. ISBN: 9781292402727 9781292402734 9781292402741.

[69] Aleksandras Melnikovas. "Towards an Explicit Research Methodology: Adapting Research Onion Model for Futures Studies". en. In: *Journal of Futures Studies* 23.2 (Dec. 2018). DOI: 10.6531/JFS.201812_23(2).0003.

[70] Kaya Yilmaz. "Comparison of Quantitative and Qualitative Research Traditions: epistemological, theoretical, and methodological differences". en. In: *European Journal of Education* 48.2 (June 2013), pp. 311–325. ISSN: 0141-8211, 1465-3435. DOI: 10.1111/ejed.12014. URL: https://onlinelibrary.wiley.com/doi/10.1111/ejed.12014 (visited on 04/02/2025).

[71] Thomas Greckhamer et al. “Studying configurations with qualitative comparative analysis: Best practices in strategy and organization research”. en. In: *Strategic Organization* 16.4 (Nov. 2018), pp. 482–495. ISSN: 1476-1270, 1741-315X. DOI: 10.1177/1476127018786487. URL: https://journals.sagepub.com/doi/10.1177/1476127018786487 (visited on 04/02/2025).

[72] Robert A. Stebbins. *Exploratory research in the social sciences*. eng. Qualitative research methods v. 48. OCLC: 738380046. Thousand Oaks [Calif.] ; London: SAGE, 2001. ISBN: 9781412984249.

[73] Robert K. Yin. *Case study research: design and methods*. eng. 4. ed., [Nachdr.] Applied social research methods series 5. Los Angeles, Calif.: Sage, 2014. ISBN: 9781412960991.

[74] Kathleen M. Eisenhardt. “Building Theories from Case Study Research”. In: *The Academy of Management Review* 14.4 (1989), pp. 532–550. ISSN: 0363-7425. DOI: 10.2307/258557. URL: https://www.jstor.org/stable/258557 (visited on 04/02/2025).

[75] Johanna Gustafsson. “Single case studies vs. multiple case studies: A comparative study”. In: 2017. URL: https://api.semanticscholar.org/CorpusID:19081372.

[76] William C. Adams. “Conducting Semi-Structured Interviews”. en. In: *Handbook of Practical Program Evaluation*. Ed. by Kathryn E. Newcomer, Harry P. Hatry, and Joseph S. Wholey. 1st ed. Wiley, Aug. 2015, pp. 492–505. ISBN: 9781118893609 9781119171386. DOI: 10.1002/9781119171386.ch19. URL: https://onlinelibrary.wiley.com/doi/10.1002/9781119171386.ch19 (visited on 04/07/2025).

[77] Louay Karadsheh. “Applying security policies and service level agreement to IaaS service model to enhance security and transition”. en. In: *Computers & Security* 31.3 (May 2012), pp. 315–326. ISSN: 01674048. DOI: 10.1016/j.cose.2012.01.003. URL: https://linkinghub.elsevier.com/retrieve/pii/S0167404812000077 (visited on 05/30/2025).

[78] Ann Langley. “Strategies for Theorizing from Process Data”. In: *The Academy of Management Review* 24.4 (1999), pp. 691–710. ISSN: 0363-7425. DOI: 10.2307/259349. URL: https://www.jstor.org/stable/259349 (visited on 04/16/2025).

[79] Pieter-Jan Nefkens. *Transforming campus networks to intent-based networking*. eng. OCLC: 1082562050. Hoboken, NJ: Cisco Press, 2020. ISBN: 9780135466339.

[80] Karen Eilers, Christoph Peters, and Jan Marco Leimeister. “Why the agile mindset matters”. In: *Technological Forecasting and Social Change* 179 (June 2022), p. 121650. ISSN: 0040-1625. DOI: 10.1016/j.techfore.2022.121650. URL: https://www.sciencedirect.com/science/article/pii/S0040162522001822 (visited on 10/29/2024).

[81] *About SLTN*. en-US. URL: https://sltn.nl/en/about-sltn/ (visited on 01/13/2025).

[82] Richard S. Lytle and John E. Timmerman. “Service orientation and performance: an organizational perspective”. en. In: *Journal of Services Marketing* 20.2 (Feb. 2006), pp. 136–147. ISSN: 0887-6045. DOI: 10.1108/08876040610657066. URL: https://www.emerald.com/insight/content/doi/10.1108/08876040610657066/full/html (visited on 03/23/2025).

[83] Michael K. Brady and J. Joseph Cronin. “Customer Orientation: Effects on Customer Service Perceptions and Outcome Behaviors”. en. In: *Journal of Service Research* 3.3 (Feb. 2001), pp. 241–251. ISSN: 1094-6705, 1552-7379. DOI: 10.1177/109467050133005. URL: https://journals.sagepub.com/doi/10.1177/109467050133005 (visited on 03/23/2025).

[84] Leila Agha Kasiri and Shaheen Mansori. “Standardization, customization, and customer loyalty in service industry”. en. In: *Journal of Marketing Analytics* 4.2 (July 2016), pp. 66–76. ISSN: 2050-3326. DOI: 10.1057/s41270-016-0002-2. URL: https://doi.org/10.1057/s41270-016-0002-2 (visited on 03/23/2025).

[85] Carsten Glohr, Jörn Kellermann, and Holger Dörnemann. “The IT Factory: A Vision of Standardization and Automation”. en. In: *The Road to a Modern IT Factory*. Ed. by Ferri Abolhassan. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 101–109. ISBN: 9783642402180 9783642402197. DOI: 10.1007/978-3-642-40219-7_13. URL: https://link.springer.com/10.1007/978-3-642-40219-7_13 (visited on 03/23/2025).

[86] Xiaobo Xu, Weiyong Zhang, and Reza Barkhi. “IT infrastructure capabilities and IT project success: a development team perspective”. en. In: *Information Technology and Management* 11.3 (Sept. 2010), pp. 123–142. ISSN: 1573-7667. DOI: 10.1007/s10799-010-0072-3. URL: https://doi.org/10.1007/s10799-010-0072-3 (visited on 03/23/2025).

[87] John R. Hollenbeck, Bianca Beersma, and Maartje E. Schouten. "Beyond Team Types and Taxonomies: A Dimensional Scaling Conceptualization for Team Description". en. In: *Academy of Management Review* 37.1 (Jan. 2012), pp. 82–106. ISSN: 0363-7425, 1930-3807. DOI: 10.5465/amr.2010.0181. URL: http://journals.aom.org/doi/10.5465/amr.2010.0181 (visited on 04/02/2025).

[88] A. Davies. "Moving base into high-value integrated solutions: a value stream approach". en. In: *Industrial and Corporate Change* 13.5 (Oct. 2004), pp. 727–756. ISSN: 1464-3650. DOI: 10.1093/icc/dth029. URL: https://academic.oup.com/icc/article-lookup/doi/10.1093/icc/dth029 (visited on 10/29/2024).

[89] Philip Cash et al. "Sampling in design research: Eight key considerations". en. In: *Design Studies* 78 (Jan. 2022), p. 101077. ISSN: 0142694X. DOI: 10.1016/j.destud.2021.101077. URL: https://linkinghub.elsevier.com/retrieve/pii/S0142694X21000880 (visited on 05/09/2025).

[90] David Collier and James Mahoney. "Insights and Pitfalls: Selection Bias in Qualitative Research". en. In: *World Politics* 49.1 (Oct. 1996), pp. 56–91. ISSN: 0043-8871, 1086-3338. DOI: 10.1353/wp.1996.0023. URL: https://muse.jhu.edu/article/36365 (visited on 05/09/2025).

[91] Dirk Leuffen. "Case Selection and Selection Bias in Small-n Research". en. In: *Research Design in Political Science*. Ed. by Thomas Gschwend and Frank Schimmelfennig. London: Palgrave Macmillan UK, 2007, pp. 145–160. ISBN: 9781349285648 9780230598881. DOI: 10.1057/9780230598881_8. URL: http://link.springer.com/10.1057/9780230598881_8 (visited on 05/09/2025).

Appendix A: Definitions

Managed service provider (MSP)	An MSP "delivers services, such as network, application, infrastructure and security, via ongoing and regular support and active administration on customers' premises, in their MSP's data center (hosting), or in a third-party data center." [9]
Infrastructure as a service	"The capability provided to the consumer is to provision processing, storage, networks, and other fundamental computing resources where the consumer is able to deploy and run arbitrary software, which can include operating systems and applications. The consumer does not manage or control the underlying cloud infrastructure but has control over operating systems, storage, and deployed applications; and possibly limited control of select networking components (e.g., host firewalls)."[8, p.3]
Organizational design	"The deliberate process of configuring structures, processes, reward systems, and people practices to create an effective organization capable of achieving the business strategy."[11, p.1]
Team topology	The team topologies model defines four distinct types of team configurations: stream-aligned, enabling, platform, and complicated-subsystem teams [49]. While the original term topology is used to describe the structural and interaction patterns between teams, it can be ambiguous in the context of Infrastructure as a Service (IaaS), where topology typically refers to network or system layout. To avoid confusion, we refer to these team types as team configurations throughout this thesis.
Team configuration	A team configuration represents a typical arrangement outlining a team's structure, key functions, collaboration methods, and relationships with other teams.
Case	A case represents an individual company in terms of the components of organizational design: structure, strategy, frameworks, capabilities, culture, and security.
Organizational design pattern	An organizational design pattern, on the other hand, is a collection of cases that share common characteristics across these same components, emphasizing high-level similarities rather than individual specifics.
Scenario	A representation of a sample organization illustrating how the proposed model can be applied.

Appendix B: Pattern description

This section provides more insight into the patterns established in this thesis by showing more information on structure, maturity, frameworks, culture, and security per pattern.

For each pattern, a high-level structure based on the cases is created. The visualizations of the structures per pattern is next. The NIST cloud reference architecture is used to provide more information on the role of the teams within the different topologies [34]. NIST provides four layers: service, control & resource abstraction, and physical resources. More information on the service model can be found in 2.1.4. The layers of the service model are visualized on top of the team configurations using dotted lines. It is important to mention that a team can be active in multiple layers. The service model provides an architectural view in which teams are not restricted to activities within one layer.

Although the team interaction modes are not explicitly visualized, the teams operate in alignment with the interaction principles outlined in the Team Topologies model. In this structure, platform teams provide X-as-a-Service capabilities to stream-aligned teams, enabling them to offer support and guidance that enhances the effectiveness of these teams. As team interactions were not the primary focus of this study, they were not examined in depth and are therefore not depicted in detail. For example, the collaboration interaction mode presents an interesting opportunity for further exploration. The visualization adopts a layered approach, where teams deliver services to those positioned above or below them in the structure. This layered interaction is further detailed in the accompanying description of the visualization. The specific team interaction modes represent a valuable area for further investigation in future research.

Platform-driven self-managed platform

Structure:

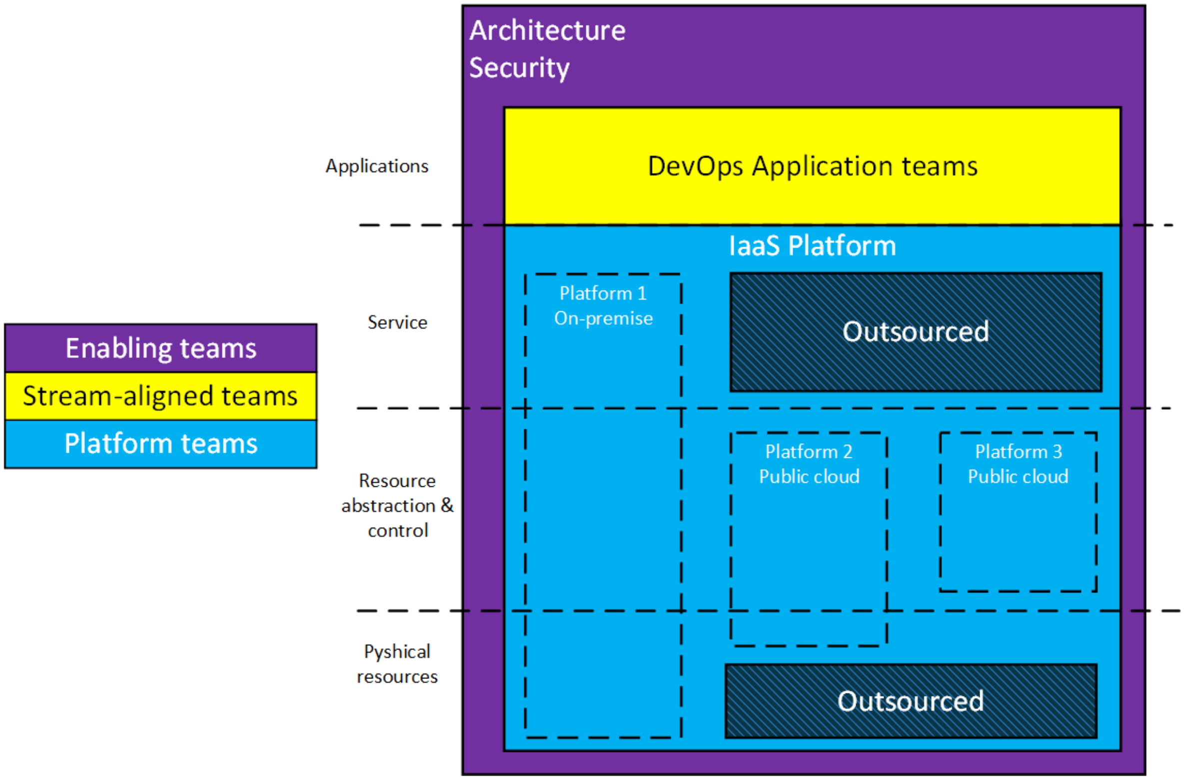

Figure 28: Structure platform-driven self-managed platform

Physical resources:
Within the platform-driven self-managed platform, the public cloud services are are predominantly used, outsourcing most of the hardware layer to public cloud providers. Outsourcing most of the hardware layer enables the platform team to keep their activities within a single team. Nevertheless, it is possible to have an on-premises platform, as case B shows. It is important to note that the on-premises platform of case B is small compared to the on-premises platforms of the platform-driven and solution-driven managed service providers. On the visualization of the structure, the second public cloud platform is drawn in the resource abstraction & control layer as well as the physical resource layer. This is because some public cloud providers allow direct access to specific hardware, meaning physical resource activities are performed. For this platform, the data center layer is still outsourced.

In conclusion, this pattern mainly uses hardware owned by the public cloud vendor (for example, Microsoft). However, when executed properly, a small on-premises platform (hardware owned by the company) is possible.

Resource abstraction & control:
The resource abstraction & control layer is visualized as a single layer in the platform-driven self-managed platform because no separate teams are working at one of these layers. Within this layer there are three platforms, all within the same IaaS platform team. The three platforms use different physical resources, so they are visualized with a dotted line to distinguish them. Furthermore, each platform has dedicated engineers, resulting in multiple clusters within the IaaS platform team. The IaaS platform team is responsible for keeping the platform running and up-to-date, and maintaining the pipeline used to request services.

Service layer:
The public cloud platforms use the service portal offered by the public cloud vendor, i.e., the service

layer is outsourced for the public cloud platforms. The on-premises platform team needs to develop an on-premises platform service layer. When the on-premises is small, the on-premises platform team can choose not to make a service portal and use direct connections with the employees working on the on-premises platform to receive requests.

Application layer: It is essential to mention the application layer for the platform-driven self-managed platform, as infrastructure specialists are within the application teams. These infrastructure specialists are responsible for requesting and operating the infrastructure services. The IaaS platform team can be kept small because the responsibility of requesting and operating infrastructure services is assigned to the application teams.

Enabling teams: The enabling teams of the platform-driven self-managed platform are architecture and security. These teams are part of the larger IT organization where the IaaS platform team operates and are not solely scoped on infrastructure. There is no automation enablement team for the infrastructure platform, as the platform team is responsible for automation and standardization.

Frameworks:
First of all, the IaaS platform team in the platform-driven self-managed platform is a single team that is part of a larger (IT) organization. Because of this, the platform team might be forced to work according to the frameworks of the larger organization. Secondly, it is notable that the two cases of pattern 1 have an insignificant overlap in their frameworks; case A focuses more on an agile way of working (scrum & SAFe) supplemented by DevOps and ITIL, whereas case B depends on mature self-steering teams working DevOps and are project-based. Even though the number of cases (2) on which this pattern is based might be unrepresentable, some common factors can be found.

The goal in pattern 1 is to maintain and operate the IaaS platforms as efficiently as possible with as few employees as possible. The frameworks should support this. The tasks of the IaaS platform team must include:

- offering infrastructure services to the application teams;
- maintaining and updating the platforms to keep the platforms secure and up-to-date with the latest market trends and the application teams' requests;
- focus on standardization instead of customization.

As an extra service to the application teams, the platform team might:

- help the application teams request services from the right platform;
- provide a pipeline that the applications can use to request infrastructure services. This pipeline should add security measures to the request to secure the provided service.

Although the platform team might help application teams to request the right services, it is not responsible. The platform team is solely responsible for delivering the requested infrastructure services. However, when a pipeline is offered to make requesting services more manageable for the application teams, the platform team is responsible for maintaining the pipeline.

Maturity:
To execute these tasks efficiently, having experienced/senior employees *(A & B)* is essential. Even though case A has an experienced team, the overall maturity score is average, whereas the team and organization level maturity at case B is high. Offering multiple platforms and operating using as much standardization and automation as possible while connecting with the larger IT organization requires a high maturity of the frameworks used.

Frameworks:
The IaaS platform team should use DevOps and ITIL *(A & B)*. ITIL can support operational tasks, and leveraging a DevOps way of working will support the combination of development and operational tasks linked to the platform. The daily approach of dividing development and operational tasks can differ. For example, an Ops lead of the day can be used where a single employee is responsible for the

operational tasks of the day *(A)*. This role might rotate across team members. Another option is minimizing operational tasks through automation and holding the whole team responsible for operational tasks that arise *(B)*.

In this pattern, the application teams are most involved in requesting and maintaining the infrastructure. The tasks of the application teams must include:

- request the right infrastructure services;
- integrate the infrastructure services in the application;
- operate the requested infrastructure services.

When infrastructure services are provisioned to the application teams, they operate them. This responsibility includes, for example, expanding storage. Application teams should have one or more infrastructure specialists to execute infrastructure-related tasks. Organizations might choose to establish a team to which operational tasks of application teams can be outsourced, relieving application teams of these operational tasks*(A)*.

As mentioned, the IaaS platform team is part of a larger organization. This organization might work according to scrum and SAFe *(A)*, work more project-based *(B)*, or have another operating model.

Security:
The IaaS platform team is responsible for offering a secure platform, i.e., there should be a secure environment in which the infrastructure services are provisioned. Furthermore, the platform team is responsible for providing guardrails on security when the application team requests services. The application teams are ultimately responsible for requesting the right components to form a safe infrastructure environment.

Platform-driven managed service provider

Structure:

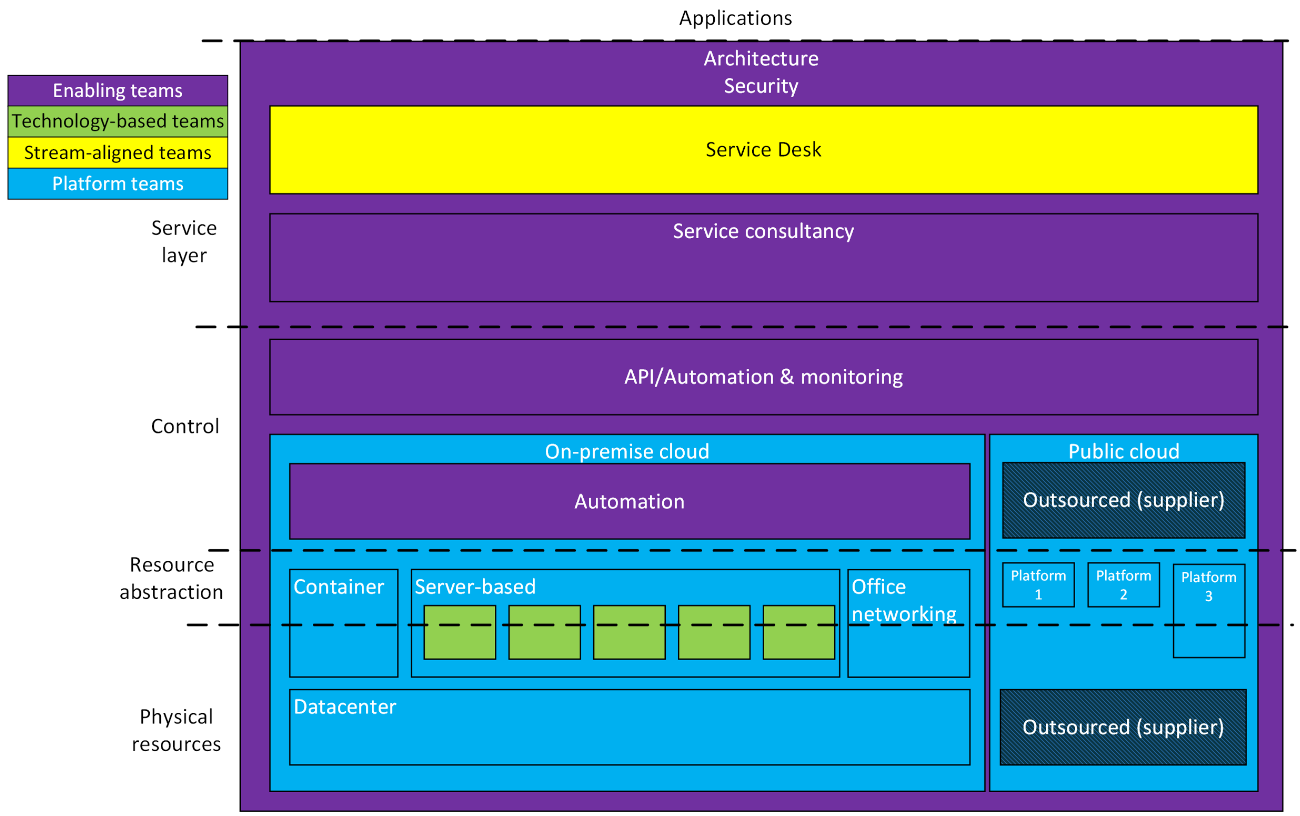

Figure 29: Structure pattern 2

Physical resource layer:
Most of the hardware layer within the platform-driven managed service provider is on-premises. In addition, the public cloud, where the datacenter layer is outsourced, is used. Not all hardware is outsourced, and operational governance is high, so pattern 2 requires many teams.

The data center manages the hardware (racks, switches, servers, storage, etc.) and needs to function as a platform to reduce the cognitive load of the platforms above it. The container and server-based team should not be directly involved in physical hardware-related activities. Instead, they are enabled by the hardware delivered by the data center.

Team topologies model state that platform teams should deliver the services through a (self-service) platform [49]. However, as the data center works with physical components that come with high investmens, it does not use a self-service platform. When new orders are sent to the data center, delivery times often depend on external parties and are long. Even though the data center lacks a self-service platform, it is still best visualized as a platform because it delivers services to reduce the cognitive load of other teams. The data center platform consists of a single team responsible for floor management, the boots on the ground in the data center. Floor management is responsible for managing and ordering hardware.

The platforms on top of the data center platforms are drawn in two layers. This is because the layers described in the NIST reference architecture are strict when looking at it from an architectural point of view. However, teams can be involved in multiple layers. The platforms offer a customer physical components as a service using resource abstraction. Therefore, the on-premises platforms are visualized in two layers. Furthermore, a public cloud platform might offer direct access to specific physical hardware; in this case, a public cloud platform is involved in the physical resources and resource abstraction layer. Both options are included in the visualization.

Resource abstraction:
The resource abstraction layer of the platform-driven managed service provider consists of 3 on-premises and several public platforms. First, the server-based platform. This platform consists of multiple technology-based teams. A technology-based team focuses on a single technology domain. For a server-based platform, examples of this are storage, backup, compute, or operating systems (Windows & Linux). Next to the server-based platform, there is a container-based platform. In this pattern, these technology domain teams are DevOps teams that integrate into a platform. Thirdly, the office network team offers hardware in office buildings, such as connectivity. It has no direct link to the activities within the data center.

In the platform-driven managed service provider, two on-premises and three public platforms are visualized because this matches the platforms across cases C-H. However, adding more platforms to the resource abstraction layer is possible.

Control layer:
The control layer of the platform-driven managed service provider is separated from the resource abstraction layer, as separate teams are involved in control tasks. The tasks of the control layer include resource allocation, access control, and usage monitoring [34]. For this pattern, resource allocation should happen mostly automatically. For the public cloud, this is implemented by the public cloud vendor; for the on-premises environment, a separate automation team is needed. This team enables the on-premises platform to provision services automatically.

On top of the on-premises platforms, there is an API/Automation team that enables platforms to automate their resource allocation as much as possible to offer their services in a shared portal. Furthermore, there is an overarching monitoring team that monitors all provisioned services.

Not all three teams of the control layer were present in all cases. However, properly implementing the control layer is crucial to let this pattern function correctly because this is where a high degree of automation is reached.

Service layer:
The service layer consists of two types of teams. Firstly, the service consultancy team focuses on enabling and helping clients request the right environment and services. Service consultancy is crucial when application teams have a low maturity concerning infrastructure or when there is a strong customer focus. The main goal of the service consultants is to help customers use as many standard components as possible on the right platform. This is especially helpful when customers do not have infrastructure competencies; service consultants help bridge the knowledge gap. Secondly, a service desk focuses on operational tickets and other questions. The service desk is responsible for routing the operational tickets to the right teams when necessary.

Enabling teams:
The architecture and security teams are the platform-driven managed service provider's enabling teams. These teams are dedicated to the IaaS provider, even though they can be part of a larger team supporting a larger IT organization. The architecture and security teams enable the other teams with their discipline.

Frameworks:
To function as a service driven managed service provider several functionalities are must-haves for pattern 2:

- offer and maintain infrastructure services;
- maintaining and updating the platforms to keep the platforms secure and up-to-date with the latest market trends and the application teams' requests;
- focus on standardization instead of customization;
- automate the provisioning of the on-premises environment where possible.

As an extra service to application teams the teams within pattern 2 could focus on:

- Offering service consultancy to help application teams request the right services.

Maturity:
To be able to execute both operational and development tasks, the maturity scores of the cases vary from medium to high. When maturity scores of cases are low, cases use less standardization and are more customer-focused; this is the case for pattern 3.

Frameworks:
The cases of this pattern overlap in their frameworks; all work according to Scrum, DevOps, and ITIL. Furthermore, apart from case D, all cases use a team overarching framework like SAFe or the Spotify model.

A team overarching framework is necessary to coordinate the large number of teams. SAFe prescribes processes as well as team set-up (*(C, E, F, H)*, whereas the Spotify model mainly focuses solely on the team set-up *(G)*. In case G, the teams have processes prescribed by Scrum at the team level. However, higher-level processes and overarching teams are lacking. The interviewee declared that the teams did not need processes to work together, as the right people could find each other.

All cases use a combination of Scrum, ITIL, and DevOps. DevOps is used to coordinate development and operational tasks within the same teams. An Ops lead of the day might be used to provide a straightforward daily approach within a team *(F, G)*. ITIL coordinates operational tasks, whereas Scrum coordinates the development tasks.

Security:
The technology components teams are responsible for creating and maintaining a secure part of the system. The security team can enable the technology components team, but it is not responsible for it. Technology component teams can contain an infrastructure specialist responsible for security.

Solution-driven managed service provider

Structure:

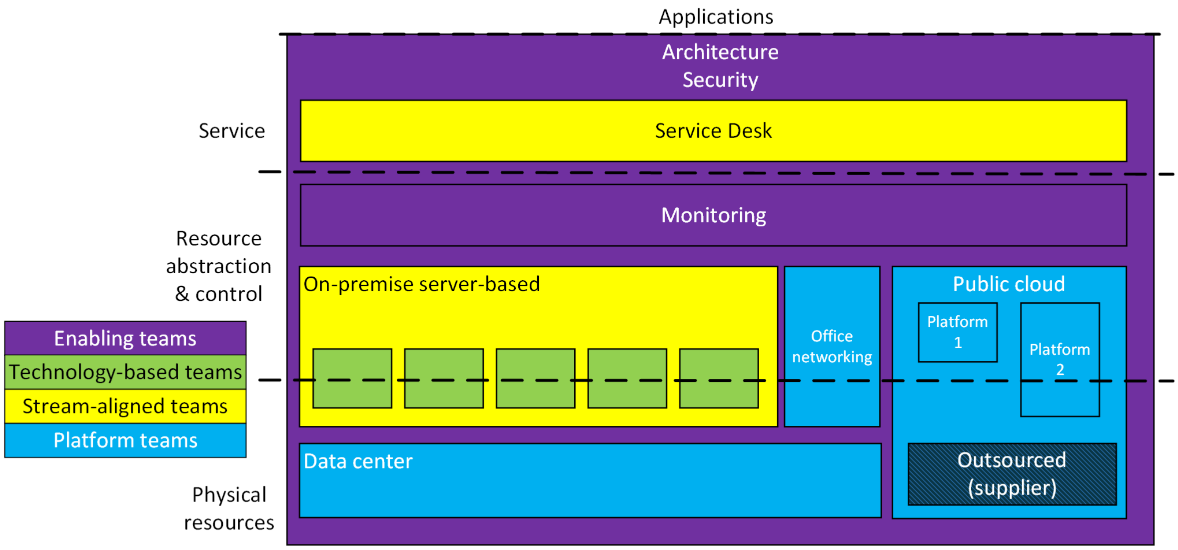

Figure 30: Structure pattern 3

Physical resources layer:
The hardware layer of the solution-driven managed service provider functions similarly to the hardware layer of the platform-driven managed service provider (pattern 2). There is an on-premises data center, and some hardware layer is outsourced to public cloud vendors. Furthermore, similar to pattern 2, the technology-based teams are in two layers, just like one of the public cloud platforms. This is because the teams perform activities in both architectural layers.

Resource abstraction & control layer:
The resource abstraction & control layer consists of an on-premises stream with technology-based teams. A technology-based team focuses on a single technology domain. Examples of this for a server-based platform are storage, backup, compute, or operating systems (Windows & Linux). In this pattern, the technology-based teams do not function as a platform due to a lack of automation, standardization, and a straightforward (often manual) orchestration process. Therefore, the teams are visualized as a regular stream, offering more customer-focused services. Next to the on-premises platform is a public cloud platform functioning similarly to the one in pattern 2.

In the solution-driven managed service provider, a single on-premises and two public platforms are visualized because this matches the platforms across cases I & J. However, adding more platforms in this layer is possible. However, pattern 2 would fit better when scaling up.

Service layer:
The service layer in this pattern consists solely of a service desk. The service desk is responsible for routing the operational tickets to the right teams when necessary. When compared to pattern 2, this pattern is missing the service consultants. The service consultants in pattern 2 help the customer pick the right standardized building blocks. Without service consultants, the customer has more direct contact with the technology-based teams, resulting in more customized services.

Enabling teams:
The architecture and security team follow the same pattern as in pattern 2.

Frameworks:
To function as a customer-driven managed service provider, several functionalities are must-haves for

pattern 3:

- offer and maintain infrastructure services;
- maintaining and updating the services to keep the services secure and up-to-date with the latest market trends and the application teams' requests.

Maturity:
The maturity for the cases of this pattern is low. For this pattern, this results in a more customer-focused organization instead of a process-focused one. Furthermore, standardization and automation are lower when compared to pattern 2.

Frameworks:
The cases of this pattern fully overlap in their frameworks. In both cases, an overarching team framework is lacking, resulting in lower cross-team coordination. Furthermore, there is no straightforward orchestration process; customers have more direct contact with the technology-based teams. This leads to more customized one-off solutions instead of standardized solutions.

Firstly, within this pattern, Scrum is used to guide the development work; however, as visible in the scores on agile attitudes (medium) and further explanation of the interviewees, not all teams have fully implemented Scrum. This results in even more discrepancy across teams. Secondly, the degree of ad-hoc work is relatively high in this pattern compared to other patterns. Thirdly, the technology-based and public cloud platform teams are responsible for the development and operational tasks. DevOps is used to coordinate development and operational tasks within the same teams, and an Ops lead of the day provides a straightforward way of working within a team. Lastly, the operational processes are executed according to the ITIL framework.

Security:
The security pattern of this pattern is the same as pattern 2.

Solution-driven Dev & Ops provider

Structure:

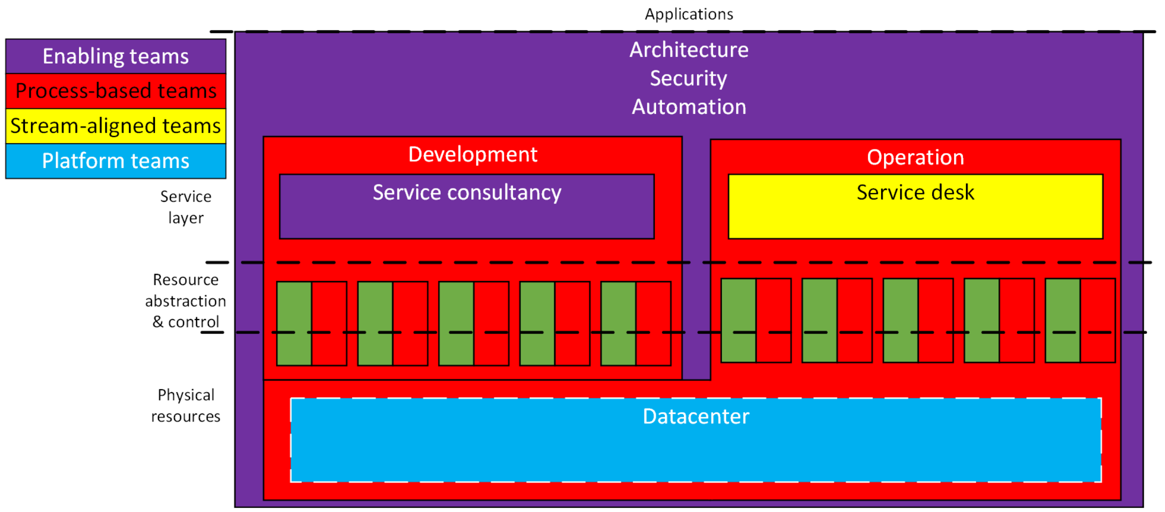

Figure 31: Structure pattern 4

Physical resources:
Cases K-N did not mention the data center in their organizational structure. However, cases K-N did mention that they use their own data center, sometimes in combination with a public cloud data center. Therefore, the data center is included in this pattern. The data center is visualized using a dotted line to emphasize the data center was not specifically mentioned in the cases during the organizational structure. Furthermore, the data center is visualized in the operation department, as mostly operational tasks happen in the datacenter.

Resource abstraction & control:
On top of the data center are the development and operations departments. The development department of pattern 4 focuses solely on developing new services and provisioning services. This department can consist of either multidisciplinary process-based teams, visualized in red *(L & N)*, or technology-based teams, visualized in green *(K & M)*. When multidisciplinary teams are used, teams can be focused on a single customer, increasing the customer focus. When technology-based teams are used, coordination across teams is more important. The development departments must create services that are easy to operate and allow the operation department to function efficiently.

The operation department is responsible for operating the services the development department provides. For the operation department, the same multidisciplinary or technology-based team approach is possible as the development department. When a company wants to be more customer-focused, multidisciplinary process-based customer teams can be used.

The teams in the technology/process-based teams in development and operations are involved in the physical resources layer as well as the resource abstraction & control layer.

Service:
When the services are provided to the customer, the customer can optionally outsource the operational tasks to the operation department. When the customer chooses to outsource tasks, the development department also hands over the product to the operation department.

In the service layer, there are two teams: a service consultancy team in the development department guiding clients through their intake or with which services to request. In addition to this, there is a service desk in the operations department that functions as a front of the operations department. This can be a skilled service desk *(N)* or a non-skilled service desk *K & L*.

Enabling teams:

The enabling teams of pattern 4 are architecture, security, and automation. The first two provide guidelines for the entire organization and can enable development and operation teams when necessary. Furthermore, the automation team enables teams in both departments to automate as much as possible.

Frameworks: To function as a solution-driven Dev & Ops provider, several functionalities are must-haves for this pattern:

- Offer infrastructure services with the option for maintenance;
- maintaining and updating the services to keep them secure and up-to-date with the latest market trends and the application teams' requests;
- offering service consultancy to help application teams request the right services;
- build services that the customer and the operations department can maintain.

Maturity:
The maturity for the cases of this pattern is medium. One of the reasons for this is that in most cases, no team overarching framework is used. Furthermore, not all processes are defined across all cases. This results in a medium maturity according to COBIT [79].

Frameworks:
The cases in this pattern show a near-complete overlap. All of them employ Scrum for certain development teams, ITIL for operational tasks, and follow a project-based approach. Case K is the only one that operates using SAFe and achieves the highest maturity score for this pattern. The emphasis on project-based work, compared to other patterns, can be attributed to the strong focus on external customers, with projects being undertaken for these clients. In Case K, project-based work is integrated with SAFe, as the entire IT organization follows the SAFe framework.

Appendix C: Interview guide

Universiteit Leiden

SSC-ICT
Ministerie van Binnenlandse Zaken en Koninkrijksrelaties

Case Interview

Interviewee:	*Interviewer:*
John Doe	Yorick Molema
Organisation	Graduation internship
john@example.com	MSc ICT in Business and the Public Sector

Context: For my master's thesis I am conducting several interviews as a part of my research, this interview is one of them. The goal of the interview is to identify common organizational designs for infrastructure providers to efficiently support the business.

I will conduct several interviews, I will keep you posted on the results of the study.

Before we begin it's important to highlight some important issues. Your response will be treated as confidential and anonymous. Participation in this interview is voluntary, you have the right to withdraw at any time without consequences. If you agree I will start a recording. The recording will be used to transcribe and analyse the interview. I appreciate your cooperation. I will send the transcript to see if you agree with everything I have put on paper. I appreciate your cooperation.

The thesis's research question is: *What are effective organizational design scenarios for a managed service provider which implemented automatic infrastructure as a service provisioning?* Allow me to give a short introduction.

Interview date:

---

1. Introduction

Question: Can you describe [INSERT ORGANISATION NAME] in a few sentences?

Question: What kind of services does your organization provide?

Question: How does IT play a role in this? What is the role of the infrastructure department within ASML?

Question: How is the infrastructure delivered to the customer?

Question: What kind of infrastructure services are delivered?

Question: How would you describe the business model of the IT infrastructure department?

Question: Do you provide services to internal or external customers?

Question: What are the customer segments you target?

Question: What is the size of your customer base? Application and/or user-wise?

Now we start with the questions about organizational design for the infrastructure department. We will discuss the structure, strategy & capabilities, processes, culture and security.

2. Structure

Question: What type of framework or operating model do you use in the infrastructure department? (SAFe, Scrum, ITIL, DevOPS)

Question: How many teams are there in your (infrastructure) department and how big are they?

Are there clusters? If so, how are they clustered? (e.g. ARTs/Trains, tribes, etc.)

What are their main roles and responsibilities? (What do they do?)

Are team members fully assigned to one team?

Why did you choose this structure? What is its origin?

Question: Can you draw the structure and connections between the teams? Add the clusters (What are the teams?).

If SAFe is being used: Do you have a portfolio or solution layer (solution layer)?

How are these connected to the rest of the department?

Can you share the organizational chart?

Question: How are the teams led?

Question: How is architecture addressed in the organization?

3. Strategy & capabilities

Question: Is there an internal IT strategy within the organization, and if so, can it be shared?

Question: Have key capabilities and skills for the teams been identified? (Use the organizational chart drawing)

If not, are you able to identify skills within the infrastructure department?

Question: What are key capabilities and skills outside the teams? What capabilities/skills are expected of a manager in your organization?

Question: Can you identify the two/three most important organizational skills that changed when you started working with the new management model?

Question: Can you name the two/three most important skills (at the team level) that changed when working with the new operating model began?

Example roles/capabilities:

- Solution architect
- Technical architect
- Enterprise architect
- Devops engineers
- Cloud engineers
- Data engineers
- Data scientists
- Data analytics
- Machine learning

4. Frameworks

Question: How are the teams connected to the customer?

Question: What kind of tools and tool chains do you use? (Linked to the processes)

Question: How are the teams interconnected, what processes do they use? (Focus on the high-level process)

Question: Have you had to make changes to the processes to implement the operating model and was that recent?

Question: Are the capabilities aligned with the processes/frameworks you use? Do you see gaps between the capabilities and the processes?

Question: Is the framework/operating model aligned with your strategic ambitions?

5. Culture

Question: How would you describe the culture in your organization and in the infrastructure department?

Question: What are characteristics for the culture in the infrastructure department?

Question: Based on your observation in the infrastructure department, how would you place the following attitudes on a scale of 1-5:

1 - Attitude is not visible at all

2 - Attitude is visible at very few times

3 - Attitude is occasionally visible

4 - Attitude is often visible

5 - Attitude is often visible throughout the infrastructure department

Attitudes:

- Attitude towards learning spirit degree: to which an actor evaluates openness and searches positively for new things regarding their work

- Attitude towards collaborative exchange degree to which actors value transparent work and share knowledge to solve problems
- Attitude towards empowered self-guidance Actors value reflection on themselves, their work processes, and take responsibility for their work
- Attitude towards customer co-creation extent to which an actor positively values being continuously oriented towards value for the customer and staying in direct contact with them

Question: Is the culture of the infrastructure department different from the culture of the rest of the organization?

6. Security

Question: Can you share about how the infrastructure department is dealing with the ever-changing cybersecurity challenges and is dealing with directives such as GDPR, NIS2, Digital Market Act (if applicable)?

7. Context

These questions can be asked if there is time left and for more context about the organization

7.1 Budget

For the study, we are interested in the relationship between the IT budget and revenue, as this has an impact on organizational design.

Question: What is the size of the infrastructure department? (FTEs)

Question: If possible, could you share figures on the annual IT budget and revenue?

Question: What is the relationship between revenue and infrastructure department budget?

Question: Are you under- or overstaffed?

Question: How many changes does the infrastructure department implement per year?

7.2 Maturity

Question: How much of an impact does legacy IT have on the infrastructure department and the services you provide?

Question: Where would you place the infrastructure department on the maturity curve (ad hoc, repetitive but intuitive, defined, managed and measurable, optimized)?

1. Ad hoc: Everything happens ad hoc, there is no standardization at all.
2. (Repetitive but intuitive) Repetitive tasks happen the right way, but there is no documentation at all.
3. Defined: Processes and responsibilities are defined and documented.
4. Managed and measurable: Processes are not only defined, but include version control and have specific checkpoints to verify that execution is correct. Risks to the business are identified, documented and potentially mitigated. Vendor strategy and lifecycle management strategies are used.

5. Optimized: Alerts are in place if the process is bypassed and management programs are in place to continuously improve risks, risk mitigations and procedures.

Thank you for participating in the interview, your input is greatly appreciated. I will share the transcript of the interview, at a later stage I will also share the results of the study.