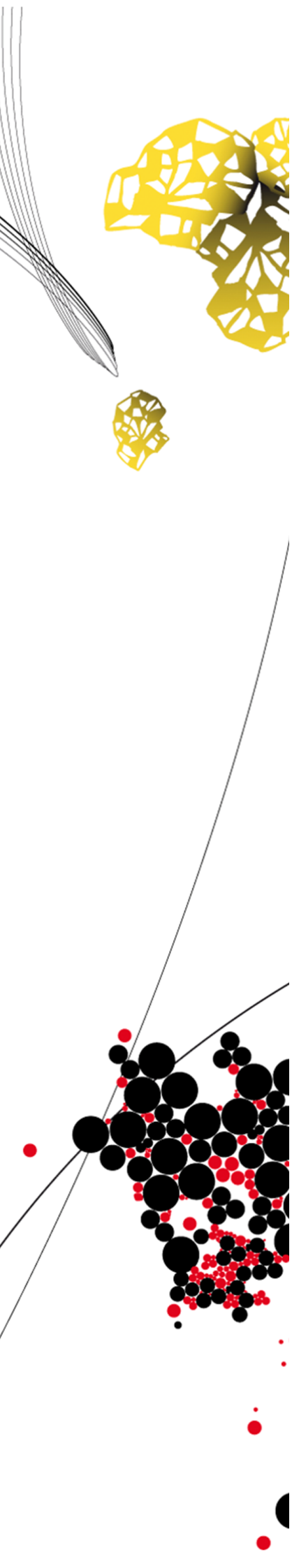


UNIVERSITY OF TWENTE.

Faculty of Behavioural Management, Management & Social Sciences

Constructing the Service Control Tower

Rogier Harmelink

PDEng Thesis

Graduation committee:
dr. Sergei Miller
prof. dr. Jos van Hillegersberg
dr. Engin Topan
prof. dr. Paul van Fenema
ing. Berend Jongebloed

Preface

The chairman of NWO, Marcel Levi, said in the summer of 2021 that science is a top sport. At this same time, I finished the work that lay in front of you. While I can confirm that working in academia sometimes feels like a top sport, the rewards are there and are worth the effort. The past couple of years within the University of Twente and especially within the IEBIS group have been a great pleasure, even during the uncertain COVID-19 pandemic.

There are a few people that I would like to thank upfront. First of all, I would like to express my gratitude to Jos van Hillegersberg, who has given me this opportunity to work on the MARCONI project and within this research group. The trust and confidence you gave me to do the research have hopefully benefited the end result, but it meant a lot for me. Next, I would like to thank Engin Topan for his expertise and even more for his pleasant personality to work with. You are one of the nicest persons in academia, and I am looking forward to working with you on the topics of this thesis.

I need to express my appreciation for Berend Jongebloed from a practical perspective. Berend, thank you for all feedback and the exciting discussions we had during multiple meetings in the past couple of years. Your valuable insights always gave a new perspective on some of the concepts we discussed. Also, I need to thank Paul van Fenema. You were an integral part of the MARCONI research. Last but not least, I need to thank Sergei Miller, Peter Jansen and the rest of the PDEng in Civil Engineering team for offering people a great opportunity after their Master's.

This research has taken place within the MARCONI consortium. I need to thank some of the people within this consortium for their feedback and support. Especially Henk Zijm, Geert-Jan van Houtum, Alp Akcay, Friso Kuipers, Dominik Mahr, Metehan Dilaver, Fiona Sloothaak, Kars Mennens and Tom Schiefer. The companies that made this research possible also deserve credits, especially those working within the Royal Dutch Navy and IHC. Thank you, Bart Pollman, Wieger Tiddens and Marc Boer, for the excellent collaboration. At last, some students have been part of this research. In chronological order, Tijmen van den Elzen, Daniël de Vries, Ian van der Plas and Abel Keijzer. It was a pleasure to have you on-board of this research. Thank you for the valuable results.

While doing this research, I was part of the IEBIS group within the BMS faculty of the University of Twente. I have to say, I could not have wished for a better working environment. Especially the people in room Ravelijn 3301, Asad, Martijn and Sebastian. We had great times before the lockdowns started. The laughter coming from our room even built us a particular reputation within the department and motivated me to continue to work. Next to that, thank you Elke and Gea, for all the hard work you put into keeping the department running and support when needed. Also, thank you Mike (the coach) and all the people in the running group that we have formed within the department. The multiple runs we have during the week are a pleasant distraction to all the serious stuff we do during the weeks. Finally, I need to thank Reinoud, without whom I would have never ended up in this position and continue pursuing a PhD. Our weekly lunches and discussions have been invaluable to me.

Of course, my parents, family and friends deserve credits. Thank you for the (in)direct support you offered me during the past couple of years. But I also need to thank Laura, which has always been on my side, and for whom I might have been a pain in the ass regarding my spare time.

Rogier Harmelink

Summary

In this PDEng thesis, we give a collection of guidelines for the construction of Service Control Towers. From an academic and practical perspective, many questions are unanswered on how to construct Service Control Towers. Therefore, we open this thesis with a problem setting, research questions and framework. We also introduce the practical context of this research, i.e., the MARCONI consortium and the maritime sector. The second chapter focuses on the literature currently available on (Service) Control Towers. Next, we find multiple architectures and definitions and discuss them in detail. These form the basis for the rest of the research.

In the third chapter, we investigate different business and technical dilemmas in Service Control Tower development and collaboration. Then, we define the four levels of the Service Control Tower and link them with the dilemmas in the collaboration process. In total, we explore eighteen dilemmas, their solutions and some of their architectures.

Following the dilemmas in Service Control Tower development, we discuss the Data Sharing Dilemma, a particular dilemma that we analyze more in-depth. Finally, we investigate the rationality of data sharing in collaborations. Literature on (knowledge) sharing states that sharing is always individually irrational. We use that as input for a Game Theoretical model in which sharing leads to supply chain optimization. Our results indicate that, contrary to the literature, the sharing of data is individually rational.

In the concluding chapter, we construct the Service Control Tower with the help of a reference architecture based on the MARCONI use case. Organizations can use this reference framework to develop the Service Control Tower with the described dilemmas in earlier chapters. We start by discussing reference architectures from literature. Then, we define requirements and functionalities based on potential Service Control Tower users' needs and link them towards software entities. The result is an abstract software architecture for the Service Control Tower. We rank the different elements of the Service Control Tower in terms of priority in development. Concluding, we discuss how the Service Control Tower can be offered as a Service.

Contents

Chapter 1

Introduction

Technology has changed the world and will keep on doing so. Multiple revolutions have been identified, like the Agricultural Revolution in the 17th century and the Industrial Revolution in the 18th century. Drucker (1961) argues that technology has been far ahead of science and transformed science with the help of systematic technology. With the next technological revolution taking place, Saris (1989) already envisioned that the development of computers would lead to a rise in data and alter the way it is collected. Now we are in what is called by Klaus Schwab and the World Economic Forum the Fourth Industrial Revolution.

Discussion is ongoing on implementing the enormous increase in data collection and which technologies should support this. Simultaneously, new concepts like service logistics pop up and support the transition to new ways of doing business. The combination of these concepts paves the way for new applications in industries.

In this thesis, we focus on the concept of a Service Control Tower in a maritime context. In the last few decades, technologies like the Internet (of Things), Enterprise Resource Planning and Cloud Computing have revolutionised how we do business. However, when looking at maintenance in the maritime context, we seem to be still at more conventional methods. One should, however, not downplay the importance of the maritime sector. Maritime transport contains 80% of the global merchandise trade [38]. Maintenance costs of the maritime fleet are an essential contributor to operational costs, accountable for 25%-35% [94].

We discuss the relevance of a Service Control Tower solution for service logistics in maritime maintenance. First, this chapter introduces the context of service logistics in the maritime sector and the control tower concept. The next step is to describe the maritime sector's problem to implement a Service Control Tower solution. After that, a research objective and framework is constructed. Together with research questions, these should provide the maritime sector with a method to implement Service Control Tower(s) to optimise the maintenance of their fleet. Finally, we end this chapter with a discussion of the thesis design and aim.

1.1 Context introduction

The research project behind this thesis is the MARCONI project. MARCONI is an acronym for Maritime Remote Control Tower for Service Logistics Innovation. The maritime sector plays a key role in global logistics handling. Therefore, minimising costs while keeping logistics networks operating is crucial. To keep operational levels high and efficient, the availability of maritime assets needs to be high. Developments in technology give rise to the possibility to remotely monitor ships and their conditions.

The MARCONI project's primary focus is to adopt new technologies to make data-driven maintenance and operation of maritime assets possible. To adapt the new technologies for the higher availability of maritime assets, we focus on three topics in the Marconi project. The development of new service logistics decision models, designing a control tower operating model and last but not least, a control tower architecture. This thesis is on the topic of control tower architectures.

Nevertheless, first, we introduce the topics of service logistics, maintenance and the control tower. Service logistics is defined by Davis and Manrodt (1991) as "the management of responsive activities. Service logistics is the management of dynamic organisations which can respond to a wide variety of needs". A more precise definition, aligning with the MARCONI context, is given by Cohen et al. (1997) as "After-sales service logistics systems support the provision of service parts, maintenance and repair services to product end-users." It is critical to move from offering products to services in service logistics, fulfilling the end user's needs. The process of going from product-based business to services is called servitisation. Defined by Baines et al. (2009) as "Servitization is the innovation of an organisations capabilities and processes to better create mutual value through a shift from selling product to selling product-service systems.".

When looking at maintenance in the maritime sector, we include all processes, resources and activities related to high-value assets' upkeep. In the case of the maritime sector, these are usually ships, often with different purposes. For example, ships can dredge, provide safety from the sea (naval ships), or transport goods or persons. However, these ships' complexity (e.g., containing a multitude of (system of) systems) makes it hard to plan and execute the maintenance support within a supply chain of organisations. As a result, the business of maintenance in the maritime sector is still primarily product-based. However, a technological possibility could accelerate the transformation to servitisation by introducing the concept of a control tower.

Control towers are (inter-) organisational systems that are designed to optimise supply chain performance. For example, a Service Control Tower optimises service

logistics processes (in)directly related to maritime assets' availability in maritime maintenance. However, this is how we perceive the control tower in the MARCONI context. There are different definitions for a control tower, which we discuss in the second chapter of this thesis.

1.2 Problem introduction and research objective

First, we need to introduce the problem context of (service) control towers within the MARCONI project. The main topic is divided into three parts of research (i.e., optimisation models, business models and the IT infrastructure); this research focuses on the IT infrastructure of a service control tower (SCT). Topan et al. (2020, p.403) describe in their research five companies that either already have an SCT, are developing an SCT solution or want to develop one. Most companies in the MARCONI consortium are in a similar business context as the companies discussed in the paper by Topan et al. (2020). However, little is known about the IT components in an SCT and their functionality when looking at literature. We could confirm this with participants in the MARCONI consortium.

Therefore, science and practice need to define what an SCT is from an IT architecture perspective. The IT architecture of an SCT can be seen as a design problem. Wieringa (2014, p.15) defines a design problem as: "a problem to (re)design an artifact so that it better contributes to the achievement of some goal."

The objective of this research is to design this artifact and therefore contribute to solving the design problem. Wieringa (2014, p.16) concertises a design problem with the use of a design problem template, filled in for the IT infrastructure research on SCT, and we end up with:

- Improve the (service) logistics maintenance (maritime) supply chain
- by designing a (service) control tower
- that satisfies functional software and IT infrastructure requirements
- in order to increase the availability (in trade-off to costs) of high-value assets

Next to designing the SCT from an IT architecture perspective, this research tries to give insights into the rationale of developing such a system. Supply chain integration is a complex subject, and before systems like an SCT can be constructed, there needs to be an individual benefit for a participant. The primary function of such a system is to share data within the SCT environment. Without data on the current status of the supply chain, optimisation of that same supply chain is challenging. Therefore, a participant in an SCT environment needs to have an individual benefit to start sharing the data in the system.

Furthermore, when an individual organisation is willing to construct an SCT environment, a network of organisations can also be constructed. When designing an SCT in a network, there are multiple trade-offs (which we will refer to in this thesis as dilemma's) in the design of the system (e.g. which data standards are used, which software is needed, how is the ownership divided), which we also discuss in this thesis. Finally, we want to offer companies aiming to construct an SCT a reference framework that can be used to design and develop their solution.

The following objectives are pursued in this research, next to designing a (service) control tower.

- Review the literature on control tower(s)
- Investigate which dilemma's occur in development of SCT environments
- Investigate whether constructing an SCT is (individually) rational
- Develop a reference framework for SCT development

1.3 Research questions

Research objectives can be attained by asking the right research questions. In this research, we use multiple research questions. According to Wieringa (2014, p.14), designing an artifact is fed by knowledge goals formulated in the form of knowledge questions. In our case, the artifact has been described (the Service Control Tower). Therefore, we formulated appropriate knowledge questions that support the design goal and set the tone for this report's structure.

Wieringa (2014, p.17) states that knowledge questions can be divided into empirical knowledge questions and analytical knowledge questions. The former needs real-life data to answer them, while the latter can be conceptual and purely theoretically based. In this thesis, we use both concepts.

We formulate four research questions, all coherently covering a single (earlier described) objective of this thesis. They all work towards designing the Service Control Tower.

- What is currently known in the literature on (service) control towers?
- Which dilemma's do organisations face in the development of a (service) control tower?
- Is it individually rational to share data in an inter-organisational system?
- Which software functionalities for service logistics does the architecture for a (service) control tower contain?

We will discuss the rationale behind these research questions in a more contextual matter in the next section. There we describe the research framework and thesis design, in which the research questions play an important role.

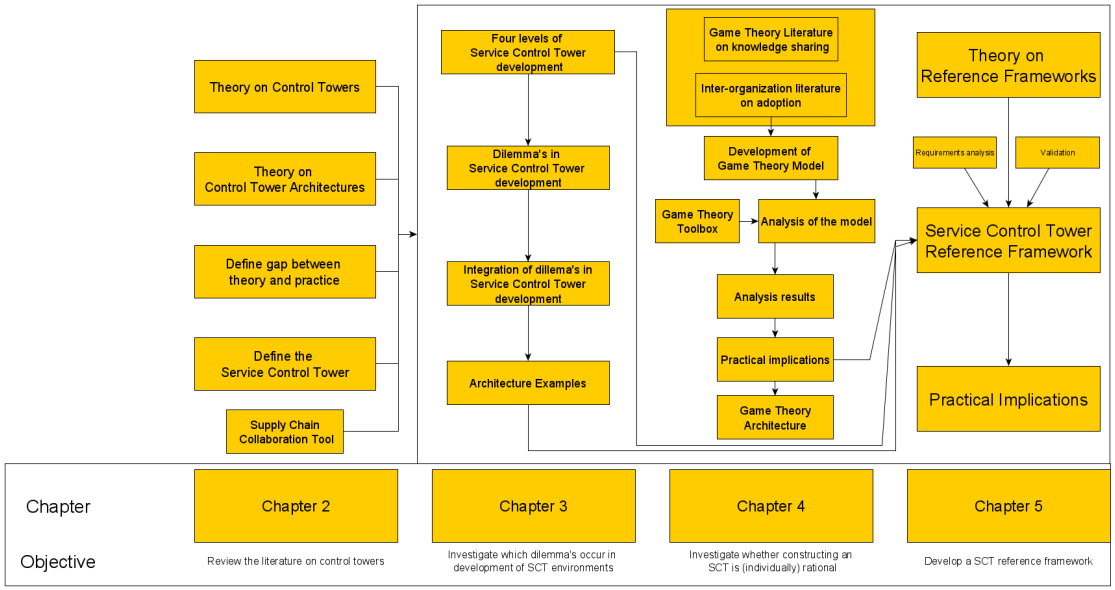


Figure 1.1: Research framework

1.4 Research framework and thesis design

The research framework is constructed to solve the problem statements and work towards attaining the research objectives. Therefore, the framework provides a scope to obtain different objectives and shows how they are interconnected with the earlier described research questions. This thesis's central research methodology follows the lines of Design Methodology as described by Wieringa. A design process is a creative process in which the design is driven by a social context, while on the other side using appropriate theoretical methods to construct the artifact.

In Figure 1.1, we show the research framework for this research. First, the research framework is categorised based on the research objectives and questions. Then, we will discuss each chapter with its rationale and link to the framework's other aspects. All in all, this research should provide supply chain partners guidelines and a rationale to develop a (Service) Control Tower solution.

First, we discuss Chapter 2, in which we review the literature on control towers. Therefore, we analyse the theory on control towers and control tower architectures and look for the gap between theory and practice. In the end, we define the Service Control Tower and explain how collaboration in a supply chain should be practiced from a structural point of view. All literature stated and discussed in this chapter feeds into one of the following chapters in the Service Control Tower design.

Chapter 3 continues our research with an investigation into the dilemma's that organisations face in control tower development. Service Control Towers are often inter-organisational systems; in collaboration, specific topics require design deci-

sions by parties (e.g., by the stakeholders in the MARCONI consortium). We discuss these design dilemmas with a Supply Chain Collaboration Tool, part of Chapter 2, which we link to the different Service Control Tower levels. In the end, we show how all these aspects are interconnected with a few architectural examples. This chapter forms a basis for research done in Chapters 4 and 5.

One of the dilemma's that organizations face is regarding the sharing of data. We already discussed potential solutions in Chapter 3, but in Chapter 4, we investigate this Data Sharing dilemma further, including the rationale of adopting a Service Control Tower. From this chapter, we can indicate the rationale of Service Control Tower adoption, in which the lack of trust (e.g., described by te Lindert (2013)) plays an essential role. Without this rationale, no individual organisation would be willing to construct such a system. Investigating these dynamics is done with the help of Game Theory literature and literature on inter-organisational systems. We construct a Game Theoretical model that we analyse with a software package called the Game Theory Toolbox. The results should indicate the rationale.

The last chapter, Chapter 5, describes how organisations can develop a Service Control Tower. First, we review the theory based upon reference frameworks, extract requirements for a Service Control Tower with workshops (i.e., with stakeholders from the MARCONI consortium), and transform these into a reference framework for the (maintenance-oriented) Service Control Tower. Finally, we end the thesis by discussing this reference framework and its practical implications.

Chapter 2

The current state of control tower research

In this chapter, we focus on the current state of control tower research. Before defining and designing the artifact of interest, the Service Control Tower, we need to look for the current knowledge available on control towers and the history preceding the first control tower applications. Therefore, we start with a chapter on the Control Tower's origin in an aviation context and concepts related to the control tower; (the history of) inter-organizational systems, supply chain management software, and Enterprise Resource Planning systems are discussed. Then, we continue with the story of the current literature on control towers. Here we discuss the concept of the control tower in the scientific literature. We follow that up with a discussion of the architectures available.

Based on the literature available, we analyse what is currently there from a theoretical and practical perspective. The analysis feeds into defining the gap in the control tower research done in the MARCONI project and creating a rationale for this research. Last but not least, we define the Service Control Tower in the context as we see it. We construct our definition and put it into this thesis's context following the earlier discussed literature and definitions. The Service Control Tower definition will be used as the basis for discussion in this thesis's chapters. Last but not least, we discuss how to collaborate in a supply chain with the help of the Dinalog collaboration tool.

2.1 The Origin(s) of the Control Tower

The term control tower has its origin in the aviation industry. The role of early control towers at airports was to provide pilots with information on the air traffic and the weather. In later years, the role of the control tower expanded due to the growth in

air traffic. Not only does the control tower provide information to the pilots, but it also controls whether aeroplanes are allowed to land, take off and are directed towards. As Meekings and Briault (2013) also mention, one could not imagine what would happen if individual pilots decide when they want to take off without keeping other aeroplanes in mind. Therefore, the control tower's function in optimizing air traffic is indispensable for safe air traffic and provides an incentive for individual airlines to adjust their business process to align with the conditions for take-off and landing at an airport.

In the past decade, the term control tower has been increasingly used by companies for an IT application that aims to optimize their logistics and supply chain processes. Applications domains range from transportation to spare parts management and distribution of goods. The term control tower is lately also used by Reich (2020) as a metaphor to examine the governance of the COVID-19 pandemic. As it seems, the control tower terminology, which originated in aviation, is now increasingly linked towards visibility and governance in supply chain mechanisms. However, before we can dive deeper into the literature and different interpretations on the control tower, we first examine the history and preceding terminology that led to the control tower's concept.

The inter-organizational system concept can be seen as the mother of integrating IT systems across organizational borders to optimize supply chain performance. Kaufman (1966) first introduced the term inter-organizational system, which was later popularized in the scientific literature and business practice. Barrett and Konsynski (1982) defined the inter-organizational system as "a general term referring to systems that involve resources shared between two or more organizations". A literature analysis by Suomi (1992) states that inter-organizational systems are different from intra-organizational systems, as the latter has only one organization that fully controls the systems, which is also the organization responsible for the costs and benefits generated by the system. Suomi (1992) states that every inter-organizational system has a sharing of data between two or more organizations based on computed technology as its key components.

An inter-organizational system aims to reduce costs, increase productivity, and increase the product market strategy [12]. Well-known adopted inter-organizational systems are, e.g., SWIFT for financial transactions in the financial industry, AIS for vessel identification and eIDAS for electronic identification of SMEs in Europe. Inter-organizational systems could also be used for supply chain management, as explained by Humphreys et al. (2001). Huemer et al. (2008) state that previously, technology was often driving the business, while in their opinion, the business should drive the technology. Examples of field labs from which business is driving technology are SCSN and RoSF. One could see this thesis's approach as similar to the

proposed steps by Huemer et al. (2008). However, we also need to reflect upon the history of business IT within an intra-organizational context for the past decades.

Since the 1990s, Enterprise Resource Planning (ERP) has been an umbrella term for software packages that offer integrated business management applications. The hype surrounding ERP followed from the growth of usage of material resource planning (MRP) and the lack of the interfacing of business software [47]. An example of ERP solving the interfacing problem is that the finance application is directly connected to procurement. The planning application can use data from human resources; in other words, the applications do not operate as a standalone. Currently, ERP is widely used, effective for management processes and even seen as a necessary tool for companies to remain competitive [84]. However, ERP is not the holy grail solution it might seem to be, Srivardhana and Pawlowski (2007) have shown that ERP can both enable but also constrain business process innovation. The focus is slowly shifting to achieve optimal performance of the firm and over the whole supply chain.

Over the last decades, ERP has increasingly been used for Supply Chain Management (SCM). Akkermans et al. (2003) state that ERP only has a modest role in improving supply chain effectiveness, with the risk that it could potentially limit progress. On the contrary, Wieder et al. (2006) found that adopters of ERP that also adopted supply chain management systems have significantly higher business performance. In 2002, Tarn et al. already saw that industry was moving from ERP to the integration of ERP with SCM. Reducing costs and increasing interactions gives a rationale for integrating IT systems in a supply-chain context. The control tower, which is the artifact of interest in this thesis, can optimize (parts) of a supply chain. Therefore we put the control tower in the context of the earlier development of inter-organizational systems, ERP but also an increasing interest in SCM.

2.2 Current literature

First, we analyse what is already known in the scientific literature regarding Control Towers. Control Towers for supply chain optimization are being constructed in different forms and for different purposes. A Supply Chain Control Tower could be an overarching term for all control tower applications in supply chain optimization. Trzuskawska-Grzesińska (2017) discusses three practical applications of Supply Chain Control Towers, either named a Supply Chain Control Tower or as a Logistics Control Tower. Other terms used for Control Towers are Integrated Logistics Control Tower (e.g., in the off-shore [64].), Transportation Control Tower (e.g., [15]) and a Rail Enabled Control Tower (e.g., [63]). Next to that, we want to discuss two other cases. The term Cross Chain Control Center (4C) is discussed by multiple au-

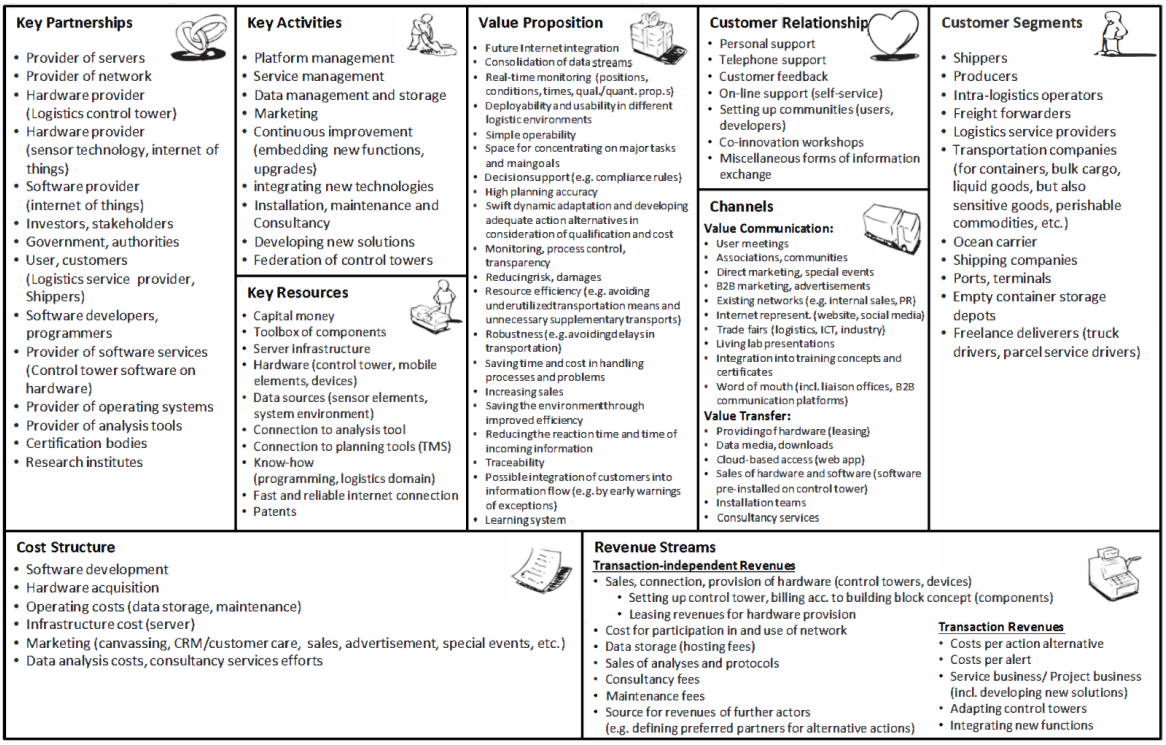


Figure 2.1: Business Model Canvas for a Control Tower [5].

thors (e.g., [93], [29], [90]). The Cross Chain Control Center can best be described as a network of Control Towers, operating together to achieve optimal supply chain performance. The last type we want to discuss is the artifact of interest in this thesis, the Service Control Tower. The term Service Control Tower was introduced by Topan et al. (2020). The Service Control Tower focuses specifically on service logistics applications; Topan et al. (2020) reviews multiple applications for a Service Control Tower in an after-sales support context. We use this chapter as an important stepping stone towards a Service Control Tower architecture. However, we first explore why companies are interested in Control Tower solutions.

Nevertheless, why are companies so interested in developing these Control Tower applications? We already reflected upon IT's history in a business context, the rationale of solutions like ERP, but what makes the Control Tower solution so appealing? Alias et al. (2015) have done research into constructing an extensive and detailed Business Model Canvas (see Figure 2.1) for a Control Tower. Every Business Model Canvas starts with a good value proposition. For Control Towers visibility, better decision making, (real-time) monitoring and gains in efficiency and, as a result, cost savings are important value propositions that drive Control Tower development [5]. On the other side, one needs to develop hardware and software to construct an IT infrastructure that supports Control Tower operations. However, these costs can either be turned into a revenue model by offering them as a service or imposing costs

per transaction [5]. In the end, a Control Tower solution could offer both users and services providers a unique business opportunity to increase supply chain performance.

However, in the construction of Control Tower solutions, usually, not everything goes according to plan. te Lindert (2013) describes three major issues: the lack of trust, intelligent IT tools, and fair sharing mechanisms. In this thesis, we try to address the lack of trust and intelligent IT tools issue. Nevertheless, others have already covered aspects of intelligent IT. Dalmolen et al. (2015) define key requirements for Control Tower solutions: modularization of services, product and processes, and capabilities for coordination, collaboration, quick connect, relationship management and risk management. Next to that, some already provide an architecture or framework for intelligent IT (e.g. [15], [96]). Still, most issues surrounding the lack of trust, intelligent IT, and fair sharing are little addressed.

As stated earlier, in this thesis, we focus on the Service Control Tower. Although different types of Control Towers exist, there also exist multiple definitions. Verma et al. (2020) define the Control Tower as ”a concept which comprises integrated supply chain and decision-making related to its needs to be fulfilled by the system capable of real-time tracking, alerts, visualization, and giving quick solutions is achievable.” Alias et al. (2014) uses the following definition: ”According to the German Institute for Standardization (DIN), control towers are decision-support systems merging different data streams from various subordinate levels and displaying the consolidated information at a higher level for monitoring and control of processes while pursuing the goal of optimal process operation.”. Milenković (2019) uses the definition ”a rail enabled Control Tower (CT) or in other words, an information-sharing platform that will support planners in supply chain optimization and fulfill the shipper’s requirements for real-time visibility in whole transport chain”. Trzuskawska-Grzesińska (2017) describes multiple definitions of Control Towers, often from white papers, having in common that Control Towers have an essential role in supply chain visibility, improving decision making and achieving strategic objectives. We end this section by introducing the concept of the Service Control Tower.

2.3 Current architectures

The previous section discussed what is currently known in the literature on Control Towers from definitions and business rationale. This section looks into architectures available surrounding Control Tower solutions, which help us sketch the current state-of-art in Control Tower architectures. These could help us with developing a reference framework for the Service Control Tower. We will start with general architectures for Supply Chain Control Tower, focusing on a few specific cases related

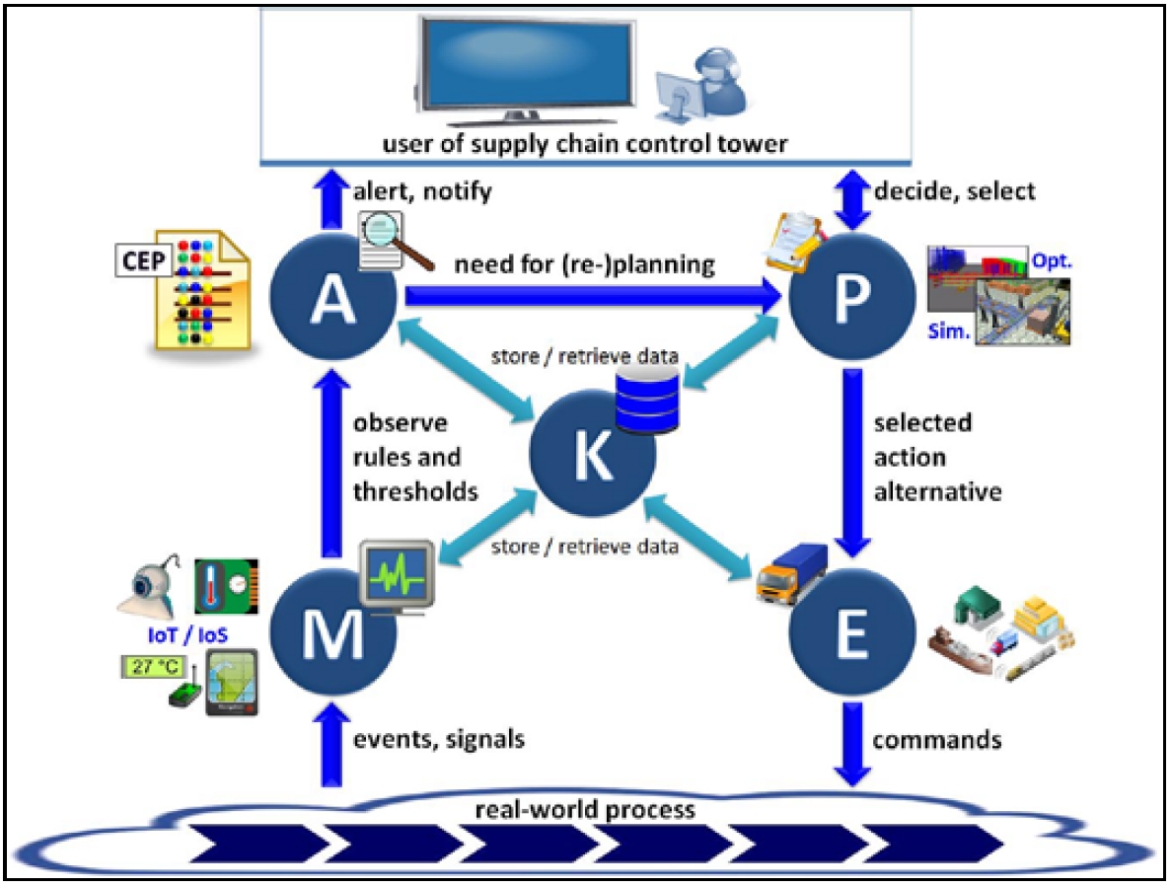


Figure 2.2: MAPE-K architecture for a Control Tower [3].

to transportation and logistics and end with architectures specifically in the SCT context.

Alias et al. (2014) describes a Control Tower's architecture based upon the main functionality: monitoring and control. For that, Alias et al. (2014) use a concept called the MAPE-K control loop. MAPE-K stands for Monitor, Analyze, Plan, Execute and Know. Figure 2.2 shows the relations between the steps in the Control Tower. Real-life processes feed the Control Tower with events and signals, which are part of the monitoring process. The data that is being retrieved needs to be analyzed and alerts the Control Tower user of a need for decision-making. Based on this, a user should decide what happens in the Supply Chain Control Tower's planning, which triggers the Plan step where the user gets alternatives from which it could choose. The action is done at the Execute step when the user has decided, impacting the real-world process. The feedback loop is applied continuously in all processes with the Know step. The Control Tower stores data and learns from the data to improve the decision making process [3]. The work of Alias et al. (2014) provides a starting point to construct the logic of a Control Tower but does not explicitly state what this logic must contain.

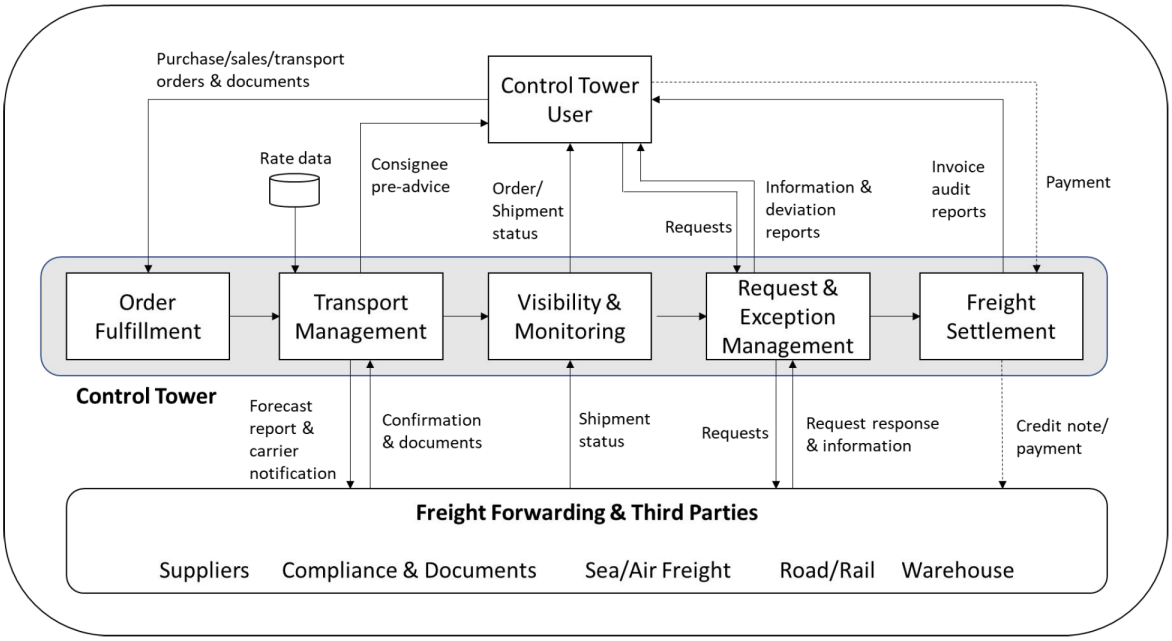


Figure 2.3: Control Tower Functional Architecture [58].

Another architecture, from a functional perspective, is provided by Liotine (2019). Figure 2.3 shows the Control Tower architecture for a Supply Chain Control Tower. The Control Tower's role is mainly to monitor or track and trace the process of an (in this case, pharmaceutical) order until delivery of the freight. The architecture describes the process in the Control Tower and between the Control Tower user and the service providers (e.g., suppliers). The Control Tower produces information about the state of the order and forecasts and plans the freight delivery for the end customer. To improve decision making, data is being shared on (real) lead times, product changes, demand patterns, interruptions in the supply chain, and inventory [58]. Liotine (2019) provides a clear architecture focused on the delivery of goods but does only reflect in a limited fashion regarding service delivery.

Shou-Wen et al. (2013) describes the concept of the 'Supply Chain Information Control Tower'. Their architecture, visible in Figure 2.4, contains five layers. At the bottom of the architecture, we see the supply chain business layer. This layer represents all activities on the supply chain level, i.e., manufacturing activities, warehousing, purchasing activities, transport, and the different actors in the supply chain (e.g., suppliers, manufacturers, sellers, etcetera.). The layer on top of the supply chain business layer is the information perception layer; Shou-Wen et al. (2013) says that it uses "Internet of Things technology to achieve real-time sensing and transmission of supply chain quality". Technologies powering this layer are RFID, GPS and other sensor technology. These feed into the information control layer, which links the supply chain's control with the information perception layer's data.

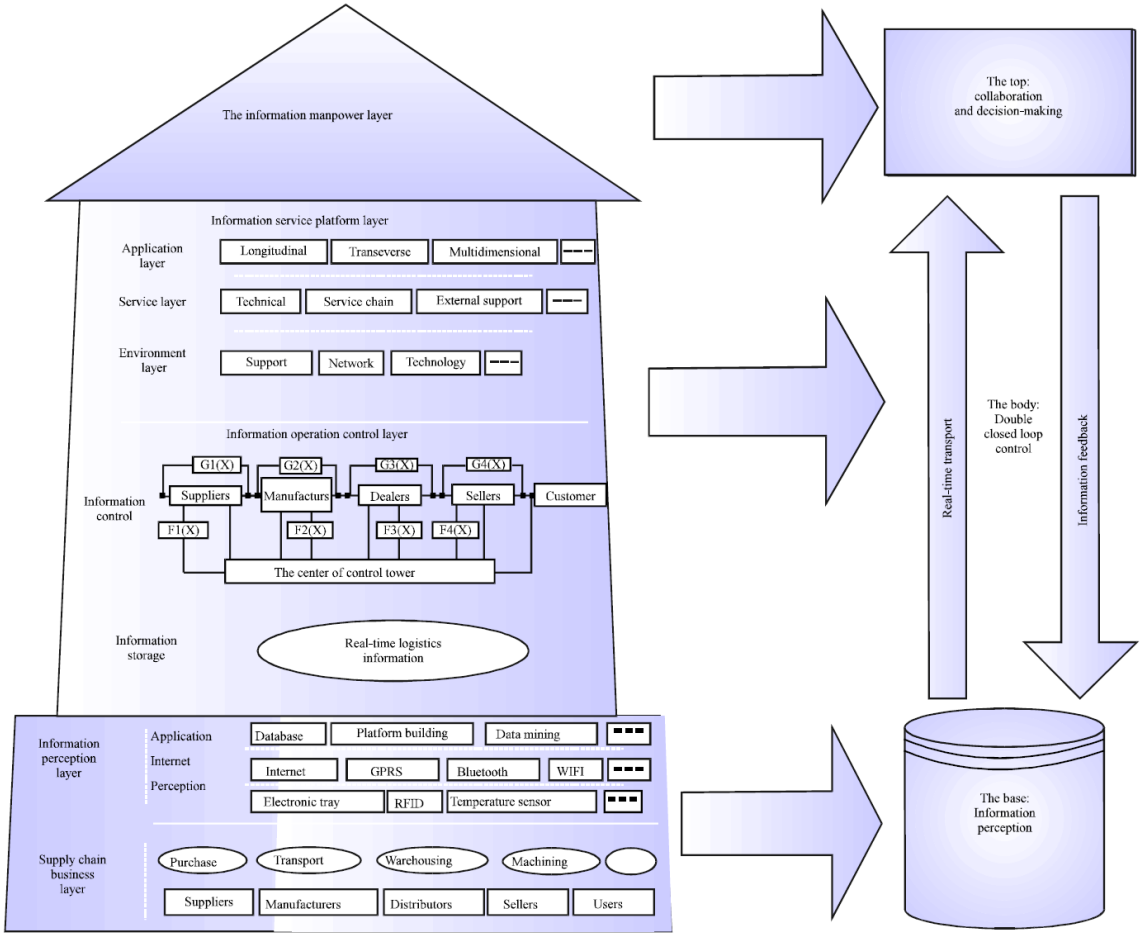


Figure 2.4: Supply Chain Information Control Tower Architecture [82].

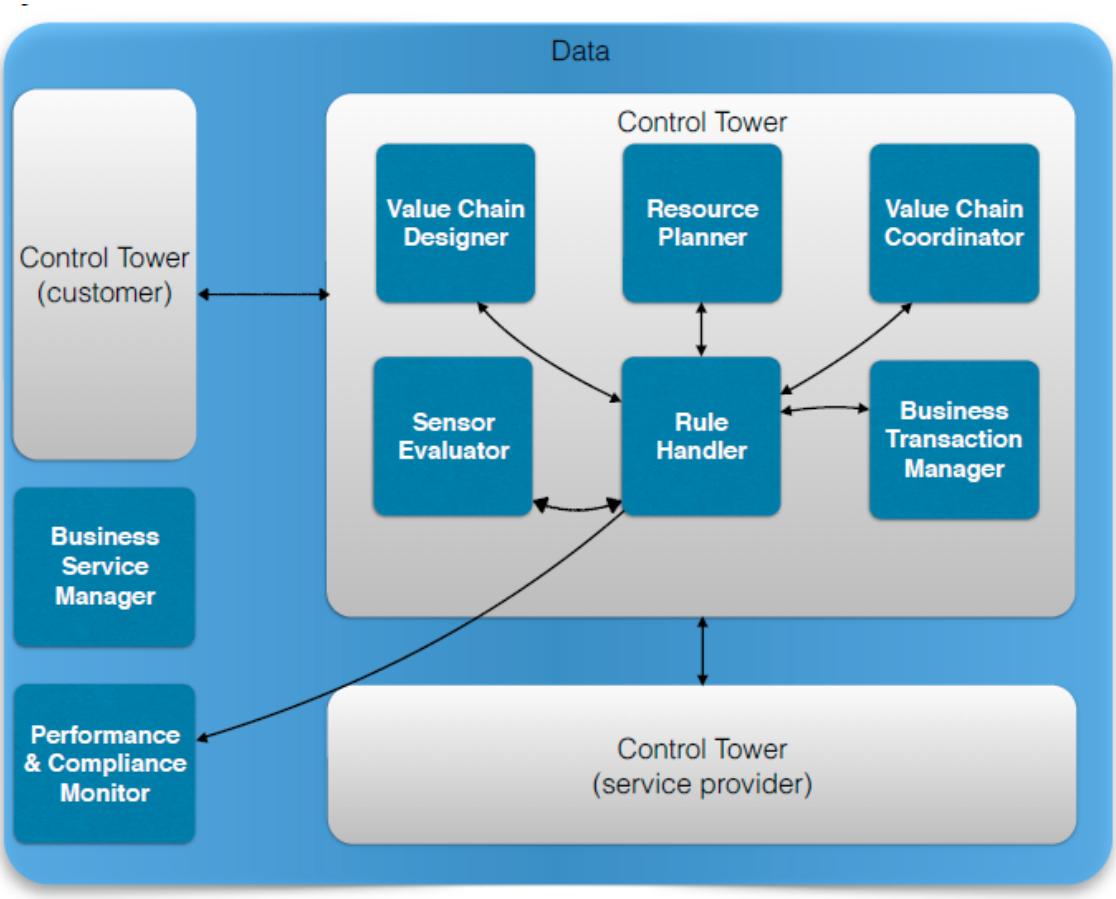


Figure 2.5: Control Tower Architecture [44].

Finally, the control tower's upper layers are the information service platform layer and the information manpower layer. The first one monitors the supply chain while the latter one is the centre of decision-making [82]. The architecture by Shou-Wen et al. (2013) provides a clear overview of global relations but does that in a very abstract matter while not defining the exact functional relations between the levels.

To continue, we discuss two architectures with a more specific focus on transportation control towers. The first one is the Control Tower Architecture for Multi- and Synchromodal Logistics by Hofman (2014). Figure 2.5 shows the proposed architecture of the Control Tower. The Control Tower is part of a cloud-based data storage, in which the control tower contains multiple components. One of them is the resource planner, which optimizes the resources available. The sensor evaluator looks for sensor data changes, and the rule handler is the sort of central piece of the Control Tower, which coordinates the event states. In the end, the architecture can handle real-time exception handling, and transaction coordination in a Transport Management System [44].

Additionally, Baumgrass et al. (2014) discuss software components for a Transportation Control Tower. In Figure 2.7, we display the proposed components by

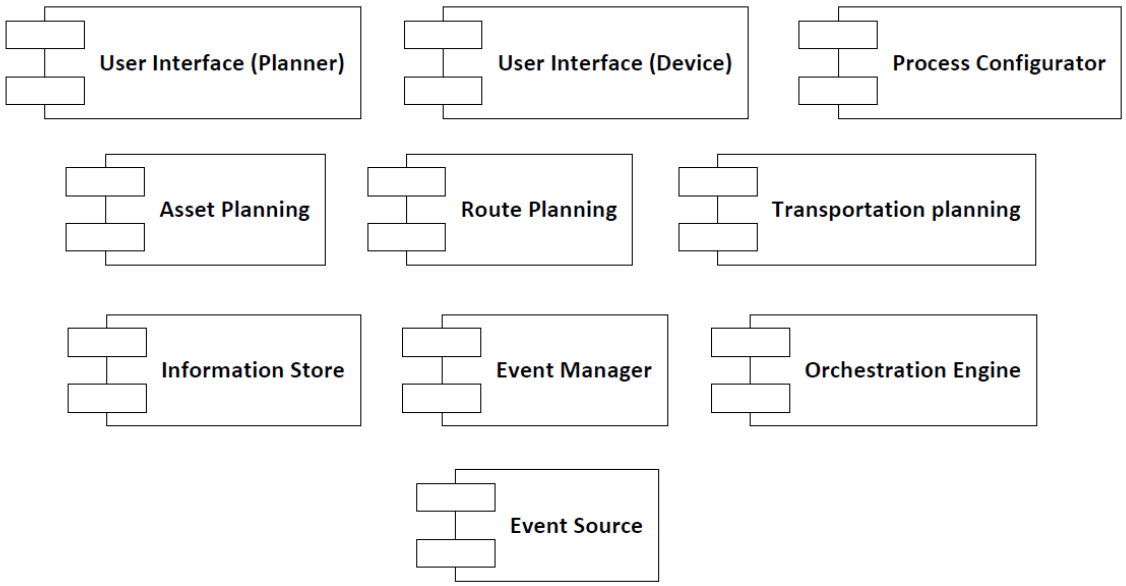


Figure 2.6: Transportation Control Tower Architecture [15].

Baumgrass et al. (2014). The key for this architecture is that transportation can be dynamically adjusted based upon the current state of resources and tasks and adjustments to the transportation plan. Multiple components are proposed, such as a route planner who does the planning of the routes, the event manager logs all events happening at a source (e.g., a truck or a ship). The process configurator is the orchestrator of a transportation process and describes the systems in which steps need to be taken and in which order. The previous two architectures provide an interesting focus on the transportation application of a Control Tower. However, they are only partially related to the context of the Control Tower in a maintenance setting.

Finally, we discuss two essential architectures related to the topic of the Service Control Tower. Rustenburg (2016) describes an IT architecture for the planning of spare parts services. The goal for this planning is to efficiently and optimally allocate spare parts for the use of maintenance. Figure 2.7 shows the architecture discussed by Rustenburg (2016). Data is fed from ERP systems within the Control Tower data manager, which feeds into the Control Tower database. Based on the database's data, a planning tool runs to allocate the spare parts within the supply chain optimally. A dashboard is there to visualize the performance of the Control Tower. Next to that, the requirements surrounding the ICT are defined. A Control Tower ICT architecture should be secure, scalable, future proof, in-house maintainable and universal. Although the paper clarifies the ICT architecture requirements, it is highly focused on Microsoft software and lacks detail on the exact configuration.

The final architecture we discuss in this section is by Topan et al. (2020). In

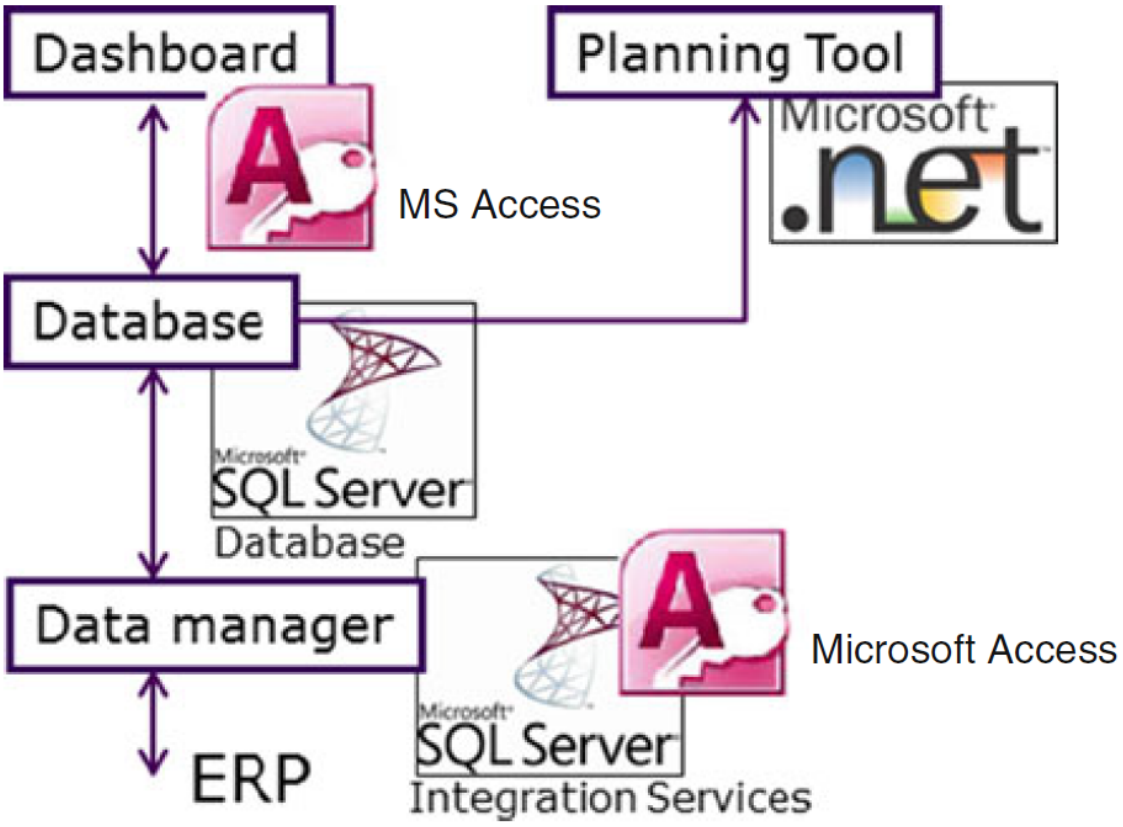


Figure 2.7: Control Tower ICT Architecture for spare parts planning [74].

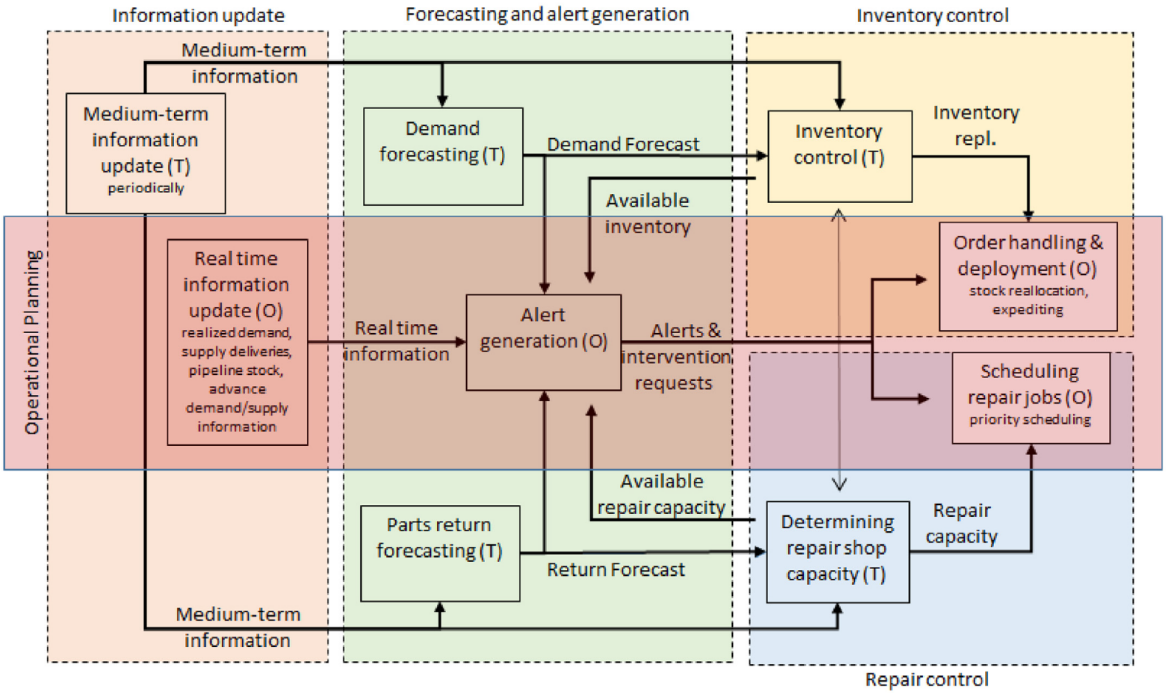


Figure 2.8: Service Control Tower process architecture [92].

Paper	Focus	Strong points	Gaps
Alias et al. (2014)	Supply Chain Control Tower Architecture	Generic architecture Continuous feedback loop	No specific domain focus Not an IT architecture
Liotine (2019)	Pharmaceutical Supply Chain Control Tower	Functional architecture for pharmaceutical logistics A clear distinction between Control Tower, the user(s) and third parties	Main focus is on pharmaceutics Not an IT architecture
Shou-Wen et al. (2013)	Supply Chain Information Control Tower	Multi-layered Control Tower architecture Takes into account multiple technologies	Relations between components sometimes unclear Does not describe IT functionality
Hofman (2014)	Control Tower Architecture for Multi- and Synchromodal Logistics	Specific focus on logistics Shows that components are interfering with each other	Details on interfaces with other CTs is missing Unclear which software needs to be implemented
Baumgrass et al. (2014)	Transportation Control Tower	Specific focus on transportation Describes the different software components of the Control Tower	Not related towards a maintenance setting Unclear how components fit business requirements
Rustenburg (2016)	Control Tower ICT architecture for spare parts planning	Specific focus on spare parts Maintenance-oriented	Microsoft heavy IT architecture Exact configuration unknown
Topan et al. (2020)	Service Control Tower process architecture	Process architecture is well-described Strong empirical foundation of the Service Control Tower	No specific focus on IT architecture

Table 2.1: Summary of strengths and gaps in Control Tower architectures

contrast to Rustenburg (2016), Topan et al. (2020) does not describe the ICT architecture, but the architecture of tactical and operational processes in the Service Control Tower. Figure 2.8 are the processes running the Service Control Tower. These are divided into five categories. First, the information update processes update real-time information regarding demand, delivery of supplies and the pipeline of stock. Second, this information is fed into the forecasting and alert generation section, where demand and parts return is forecasted, and alerts are generated. Third, based on the forecasts and alerts, two planning processes are run. One on the control of inventory, which determines the amount of inventory. The other regarding the capacity available in the repair shop and the scheduling their-off. The middle layer represents all operational planning processes in Figure 2.8. An intriguing discussion is in the paper on the role of so-called Service Level Agreements (SLAs) and the performance measurement with Key Performance Indicators from these SLAs. The architecture provided by Topan et al. (2020) gives a clear indication surrounding the (business) processes the Service Control Tower supports; however it assumes that information is already there and does not focus on an IT architecture.

Table 2.1 summarizes our findings in the current literature on Control Tower architectures. Overarching in the literature is the lack of focus on maintenance-oriented Control Tower architectures (except for [92] and [74]). Next to that, some papers focus on (software) components (i.e., [44] and [15]), but on the contrary, others do not. Therefore, this thesis focuses on the IT architecture of the (Service) Control Tower and its functionality. In essence, our work is complementary and developing on the papers by Rustenburg (2016) and Topan et al. (2020).

2.4 The Service Control Tower: Context and definition

In the previous sections, we discussed the current literature and architectures available in the scientific literature. A wide range of Control Tower literature is available,

but the discipline of Control Tower research is still in the early stages. However, we see a business incentive to develop Control Tower environments (e.g. [5]). In this section, we discuss the artifact of interest, the Service Control Tower. Next, we discuss the change of focus for businesses to offer services instead of products. Next to that, we define the Service Control Tower as we see it. Finally, we portray the Service Control Tower in the landscape of (inter-/intra-) organizational systems.

The current literature has some exciting definitions, architectures and applications of Control Towers. They have in common that the Control Tower is a system that assists in decision-making by offering monitoring and control services. Next to that, the Control Tower is a platform for information sharing in a supply chain, increasing the visibility of the same supply chain. On the contrary, there are also examples of Control Towers, which are highly focused on internal business processes with only limited applicability on the whole supply chain. Therefore it seems to be the case that Control Towers are not exclusively an inter-organizational system but could be intra-organizational.

As the application domain of Control Towers differs (e.g., transportation, warehousing, logistics), it is essential to clarify for which purpose a Control Tower solution is used. In this thesis we focus upon the Service Control Tower, two papers align closely with this concept (i.e., Rustenburg (2016) and Topan et al. (2020)). The context of these papers is to use the Service Control Tower to optimize maintenance activities for high-value assets (e.g., vessels, machinery). Rustenburg (2016) focuses mainly on the planning of spare parts, while Topan et al. (2020) has a broader view and also includes the planning of maintenance as an important factor in the Service Control Tower. The latter also links the Service Control Tower to the concept of Service Logistics, which means that a Service Control Tower could be used in different contexts.

In the past decennium, companies have shifted their focus from offering products to offering services. The process of going to offering a service instead of a product is called servitization. Servitization is defined by Baines et al. (2009) as: ”Servitization is the innovation of an organisations capabilities and processes to better create mutual value through a shift from selling product to selling product-service systems.”. In other words, organizations are not solely selling the product but a service, focusing on the purpose and goal for the end-user (e.g., paying for the use of a car in terms of kilometres driven instead of buying a car). The process of shifting from selling a product to service is best visualized by Jovanovic et al. (2016). For example, figure 2.9 shows how one could go from product sales to so-called service performance contracts.

A type of service performance contract is also discussed in the paper of Topan et al. (2020). There it is referred to as Service Level Agreements (SLAs). An SLA

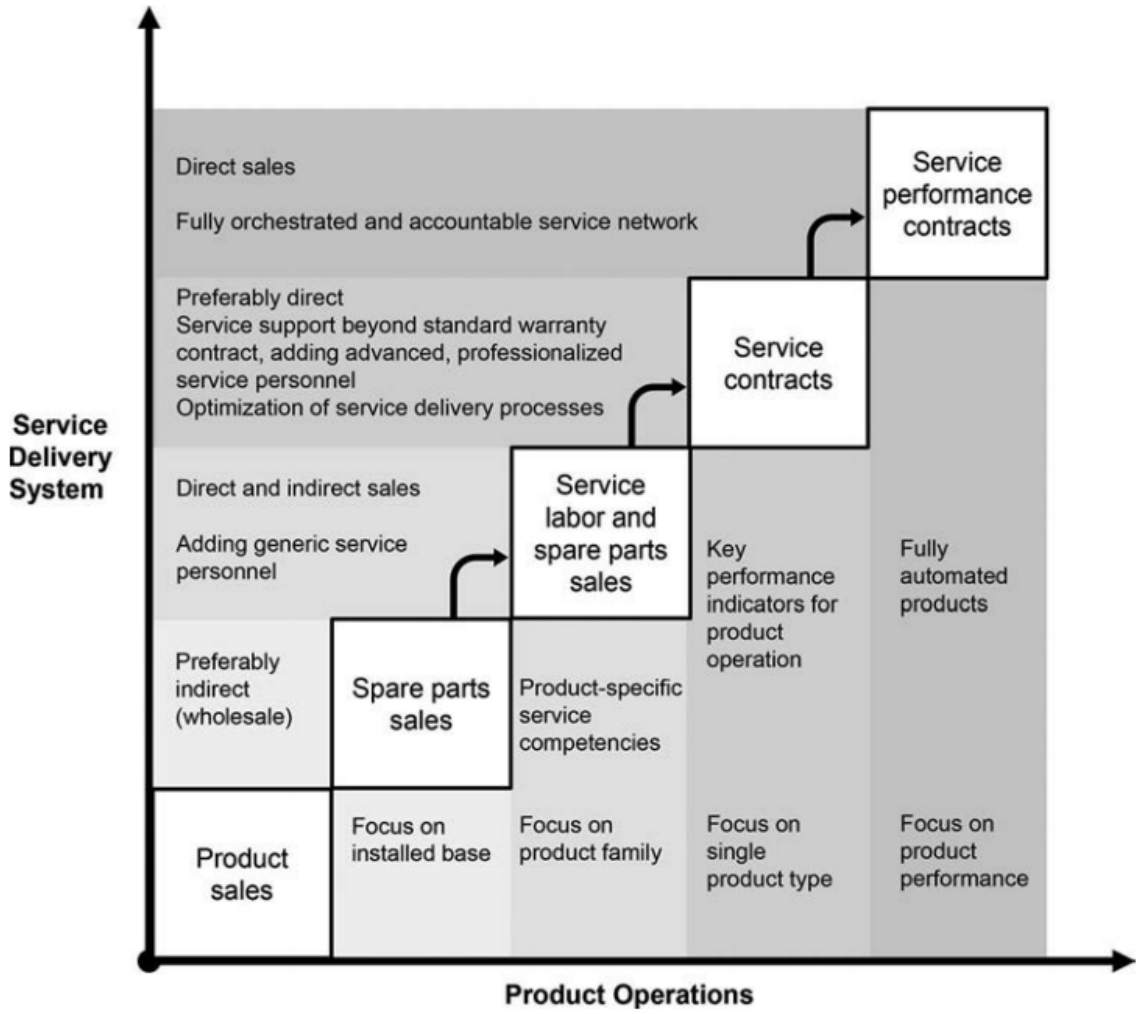


Figure 2.9: From product operations to service performance contracts [51].

is a service contract in which (at least) two parties (a service provider and service receiver) define which service is being delivered and how performance is measured (with so-called Key Performance Indicators). These SLAs can be defined at multiple levels of service provision. For example, an SLA could dictate the lead time on the delivery of spare parts (aligning with spare parts sales) and define the availability of an asset (a service performance contract). For example, in the MARCONI project *, one could align the Service Control Tower's role with a service performance contract, which defines the availability of a high-value asset (e.g., availability of 95% during the total lifespan).

The Service Control Tower plays a vital role in service logistics, i.e., offering logistics as a service. The Service Control Tower could look at the optimal allocation of resources given a specific context and deliver a specific service defined in an SLA. The Service Control Tower application could be internally oriented in an organization (between divisions), externally oriented or supply chain-wide. The Service Control Tower could even be used as a service itself; optimally allocating resources and helping in the decision-making of a business context could be valuable on its own. On the other side, the application of a Service Control Tower depends highly on its need, the business rationale, and the business processes it will support.

Topan et al. (2020) uses the following, adapted definition from Bleda et al. (2014): "A service control tower acts as a centralized hub that uses real-time data from a company's existing, integrated data management and transactional systems to integrate processes and tools across the end-to-end supply service chain and drives business outcomes.". The definition could be feasible for specific applications of the Service Control Tower. However, it is not necessary that a Service Control Tower uses real-time data and does not need to be a centralized hub. Therefore, we propose the following, more straightforward, definition for a Service Control Tower: "An (inter-) organizational system which uses IT to optimize specifically (a part of) the service logistics supply chain".

In Figure 2.10 we propose a classification for the (Service) Control Tower in terms of (inter-/intra-) organizational systems. We picture the (Service) Control Tower on the boundary of both types. We explicitly do not define which type of organizational system a Service Control Tower needs to be in the definition. In our opinion, the Service Control Tower can cross organizational boundaries, be deployed in a network of Control Towers (i.e., a Networked Control Tower), and act as an intra-organizational system, optimizing processes between departments in an organization. Next to that, we think that the Service Control Tower differs per organization and organizational context.

*See Section 1.1/1.2 for an introduction on the MARCONI project

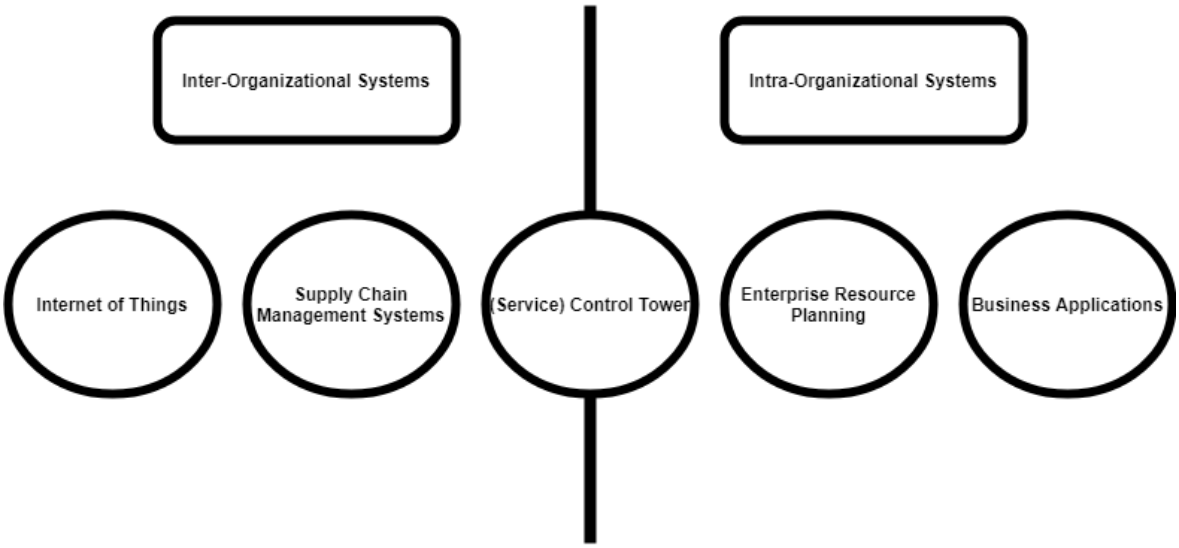


Figure 2.10: Placement of the (Service) Control Tower.

2.5 Supply Chain Collaboration Tool

The last section of this chapter describes The Supply Chain Collaboration Tool by Dinalog [33]. This tool aims to help supply chain partners to collaborate in a supply-chain context. However, empirical evidence for this tool is limited available. Therefore, it has been constructed from practical evidence. Supply chain collaboration can be in two dimensions, horizontally and vertically. The earlier one describes cooperation between supply chain players offering the same service (e.g., fashion retailers or wholesalers of vegetables). Usually, horizontal cooperation is problematic because of similar interests; if they are not in the same organization, they often do not cooperate because of individual interest and antitrust law. Vertical cooperation is in that sense easier, as it is between two players that offer different services (e.g., a retailer and a wholesaler of vegetables) but have a common interest in cooperating to solve or minimize possible supply chain problems and therefore optimize supply chain performance. The supply chain tool is built for vertical collaboration, which means horizontal collaboration is out of the equation.

Four phases have been defined to set up a vertical collaboration (shown in Figure 2.11) in the Supply Chain Collaboration Tool by Dinalog. At first, organizations need to identify the strategic interests of cooperation in the identification stage. Then, if a common strategy and business case has been developed, the collaboration design is needed on a tactical level. Here, the consortium needs to agree on the governance of the collaboration, rules to enter and exit the collaboration, and how value is created and shared. In the third stage, the collaboration efforts are implemented in practice on an operational level. There is an explicit collaboration agreement,

Figure 2.11: Supply Chain Collaboration Tool. [33]

and logistic and financial data are harmonized. Next to that, data sharing takes place in combination with IT integration. Planning control of the supply chain is now collaborative. The last step is to evaluate the collaboration process based upon the evaluation of Key Performance Indicators and expand, maintain or terminate the collaboration. We discuss the different phases in more depth related to the artifact (Service Control Tower) in the following subsections, based upon the tool described by Dinalog.

2.5.1 Identification

The beginning step is identifying a potential collaboration in which a Service Control Tower might be helpful. A collaboration strategy needs to be determined based on the organizations' long-term goals, available resources, and the potential partners' strategies. Next, the organizations need to determine whether it is feasible to execute such a collaborative strategy and set the first steps towards a business model (for the (Service) Control Tower). The second step is to determine the partners for collaboration. A profile should be constructed in which an optimal partner is sketched, based upon offered service or knowledge about products, technologies or markets. From this profile, the first list of potential partners can be constructed. Potential partners should be selected from this list to complement the organization's resources, culture, and strategy. In the third and last step of the identification process, a business case is developed. Potential partners discuss their expectations

and indicate their expected costs, benefits but also risks associated with collaboration. Based upon these assumptions, an analysis should be done, which should give the first indication of whether collaboration (construction of a (Service) Control Tower) is feasible and profitable [33].

2.5.2 Design

The second phase is to design the collaboration (on a tactical level). In our case, the (Service) Control Tower, the first design step is to determine the governance system. Multiple aspects need to be discussed and determined, e.g., legal entity, coping with potential issues, communication structure, financial agreements, and how to deal with trust and the collaboration culture. When governance is established, the supply chain partners should design the physical, financial and information flows. In the case of a (Service) Control Tower, this is the first step to know the scope of the business architecture in which the (Service) Control Tower can be deployed. During this design, the consortium can find opportunities to improve specific business processes in the collaboration.

In the sixth step, the collaborating organizations need to discuss what to do with value creation and gain sharing. In collaboration in the supply chain, synchronising business activities can lead to cost reduction or new service offerings. Both are a form of value creation and result again for the organizations involved. In this sixth step, agreements need to be made on how gains in the collaboration (or in the (Service) Control Tower) are shared; options for gain-sharing can be found in Cooperative Game Theory. Finally, the network needs to create a set of entry s for which the collaboration is set. Parties can leave the environment of collaboration, but also newcomers can join under a stated set of conditions. This set of rules could be set in a legal framework to ensure partners' commitment to a long-lasting partnership [33].

2.5.3 Implementation

In the third phase of collaboration in a horizontal supply chain context, the collaboration implementation occurs. For a (Service) Control Tower, this means that the system itself is being deployed on an operational level. Starting in this phase is to construct a collaboration agreement that is the legal basis for the collaboration. Things like a primary contract, general collaboration rules, and the Service Level Agreement can be determined in this agreement. In the following step, the organizations prepare internally with change management to align processes, personnel, and resources for the new situation. For managers, it is essential to get everybody

involved in the collaboration on the same page. The next step will focus on the harmonization of logistics and financial data. In this step, the organizations have agreed upon a certain data standard and communication method in which information and data are shared within the consortium. One could see this step in the context of constructing a (Service) Control Tower as one that lays the foundations for the data standards in use in the system and the semantics surrounding them.

The next step, step eleven, is how a system should be constructed (e.g., a (Service) Control Tower). Data will be shared, and IT is going to be integrated with this step. The supply chain collaboration partners need to determine which systems are being used, be constructed and how they are linked with existing infrastructure. This step's valuable tool is a cost/revenue analysis, indicating which IT systems are worth the investment for integration. The last step in the implementation phase is to do collaborative planning control of the supply chain. In the (Service) Control Tower, the system has some tools and systems that can help forecasting, finance, and provide a dashboard for Key Performance Indicators. It is essential for the consortium of partners that they also have protocols in place if there is a conflict in the collaborative planning and can mitigate those [33].

2.5.4 Evaluation

The last phase is the evaluation phase. The collaboration is now operational, and the consortium should reflect on the performance. In the first step of evaluation, organizations need to look at Key Performance Indicators, not only on a financial, strategical and operational level. But also from a knowledge and relational perspective. Important in this step is the transparency between organizations and the measurability of the Performance Indicators themselves. Based upon this analysis, organizations go to the last stage of the evaluation. In this stage, they determine whether they want to expand the current collaboration, maintain it, or terminate it because of lacking performance. The results of the evaluation phase can be used for input in the identification phase. The collaboration tool is, therefore, in itself an iterative process that has in the centre the building of trust as the essential cornerstone.

In this section, we described the collaboration tool from Dinalog. During this collaboration process, the consortium's multiple dilemmas will be faced, business and technical decisions need to be taken to proceed. The tool constructed by Dinalog will be used as a context for the dilemmas faced in constructing the (Service) Control Tower environment, which we describe in the next Chapter. Constructing a (Service) Control Tower is an iterative process as well. One cannot resurrect a system without building the foundations for such a system first.

In this chapter, we reviewed the state of the current literature on Control Tower research, and from that, we see that Control Towers depend on the context they are developed in. For the remainder of this thesis, we aim to fill the partially discussed gap in this chapter. First, in Chapter 3, we discuss the steps that need to be taken to develop a (Service) Control Tower and the dilemmas in development. Then, Chapter 4 discusses an in-depth analysis of one of the dilemma's in (Service) Control Tower development, namely the Data Sharing Dilemma, in which we also discuss the potential lack of trust (e.g., described by [90] (2013) in supply chain collaboration. Last but not least, we provide an IT architecture for a maintenance-oriented Service Control Tower in Chapter 5.

Chapter 3

Decisions in (Service) Control Tower design

In the previous chapter, we explored the literature surrounding (Service) Control Tower. However, when constructing a (Service) Control Tower, the second step is to design the system. Designing such a system is not set in stone and is highly dependent on the business environment. This chapter explores the different decisions, stages, and dilemmas organizations face when designing a (Service) Control Tower.

We start this chapter in Section 3.1, describing the four levels (i.e., the technological milestones) of the Service Control Tower. Constructing a Service Control Tower is, in our opinion, a process of different stages. We describe these stages linked towards the functionality of the Service Control Tower. Finally, in Section 3.3, we link the collaboration tool from Section 2.5 and the four levels of the Service Control Tower with organizations' dilemmas from a business or technical perspective. We illustrate different decisions in these dilemmas with architectural examples.

3.1 The four levels of the Service Control Tower

In this section, we describe four levels we identified in constructing a Service Control Tower. When building such a system, one needs to carefully plan the goals for the system and develop the Service Control Tower in different phases. However, in the literature, we also find different interpretations of (Service) Control Towers levels. For example, Bhosle (2011) describes three phases (or levels) of visibility in a Control Tower, a first phase in which visibility is created on the operational level, phase two creates visibility through the supply chain with analytical capabilities. The third and last phase focuses on predictive visibility with the help of advanced self-learning algorithms. Another paper, by Liotine (2019), describes different technology levels as a pre-requisite for Control Tower operation. These are Data Capture Translation,

Data Transmission Upload and Data Access Actionability. However, the levels by Liotine (2019) focuses mainly on the requirements for data usage in a Control Tower.

In our opinion, we define the Service Control Tower levels in combination with the literal meaning of the individual words. The end goal is to have a system that can influence and achieve optimal supply chain performance based upon service contracts. However, before such a system is realized, one needs to get the earlier stages of the Service Control Tower running. These earlier stages can be related to the visibility of Bhosle (2011) or the data requirements of Liotine (2019). Nevertheless, it also addresses issues like the different modes of control in the Control Tower and the system's strategic direction.

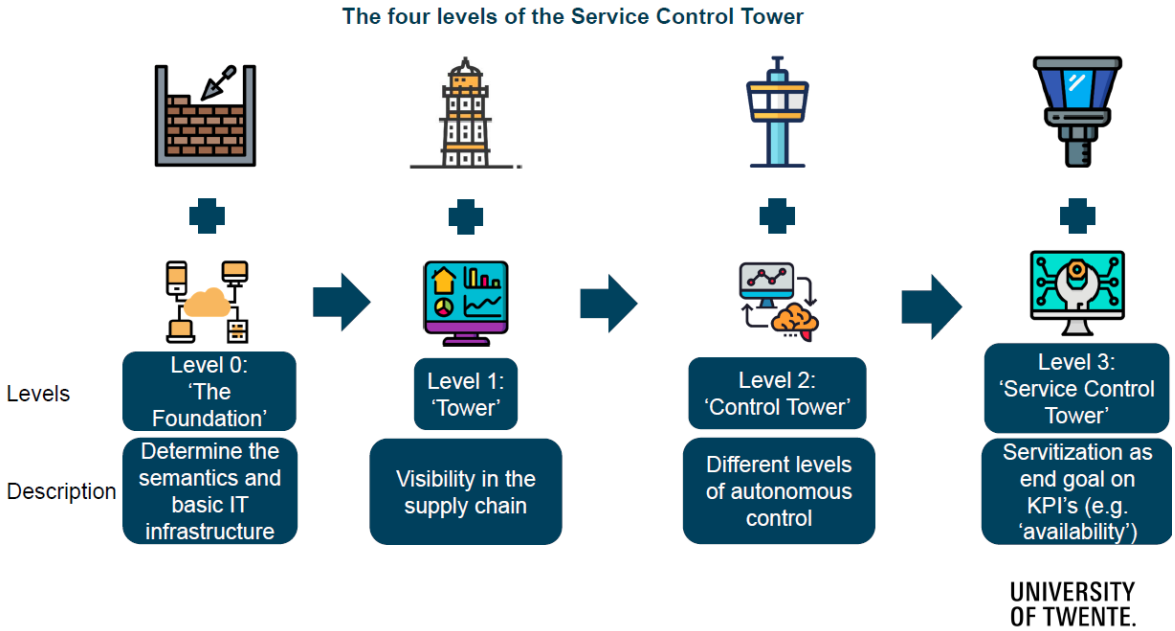


Figure 3.1: The four levels of the Service Control Tower.

The four levels that we envision are displayed in Figure 3.1. The first level in our four Service Control Tower levels is not even a first level but more of a level zero. However, this level is elementary and requires feedback in order to construct a decent basis. We define level zero as the basic foundation of the Service Control Tower. Supply chain partners first need to determine how they want to communicate (IT semantics) in a (Service) Control Tower environment. They also need to determine the IT architecture and the data standards for the (Service) Control Tower. When such a foundation has been laid, the development can go on to the Tower level. The Tower level is the first level where users of the (Service) Control Tower see the environment's first operationalization. At this level, the main focus is to create visibility in the supply chain. Dashboards are the primary tool and should give a clear insight into the supply chain's performance; one could classify this environment as a Decision Support System. It can offer participants an insight into better decisions without actually having a direct influence on the business processes.

The second level of the Service Control Tower introduces the word Control into the equation. Next to having visibility in the supply chain, the consortium can control aspects of the supply chain. Via intelligent algorithms and data analysis, better decision making can take place fully automated. However, it is also possible that participants opt to intervene in the business processes manually. In such a case, the Service Control Tower can learn from these decisions with Intelligence Amplification technology. The last level for the Service Control Tower is to design the system to operate and monitor the supply chain based upon agreed Service Control (e.g.,

Service Level Agreements).

An example would be that the Service Control Tower controls the supply chain in the maritime industry with high-value assets. An agreed level of asset availability is achieved, or spare parts are delivered within a specific Service Level. Although the goal of servitization within the environment seems like an ideal state, achieving this state is challenging and requires intensive collaboration between the supply chain participants on a strategic level. We will discuss the defined levels (from Figure 3.1) in more depth in the following subsections.

3.1.1 The Foundation

Our first level is the foundation. A foundation needs to be laid as the stable basis for functioning. In the literature, Liotine (2019) already addresses the need for basic technology requirements surrounding data exchange. Next, we see other critical key activities that should be taken in this development step, which are listed below.

1. Determine the context of the business architecture
2. Determine IT systems in use by the participant(s)
3. Determine available data
4. Determine data standard for usage in the Service Control Tower
5. Design an IT architecture for the Service Control Tower
6. Determine the main programming language and the IT tools/platforms for usage
7. Setup the first data architecture, data governance, interfaces for data exchange and databases

The Service Control Tower environment developers first need to determine the business context that the system will operate. One could align this with step 5 (Design of physical, financial and information flows) of Figure 2.11. Additionally, developers need to identify the IT systems used by the participant(s) to execute their business activities. This should give a first indication of the context in which the Service Control Tower would operate. Based on this information, developers should determine the available data, and a data standard should be set for usage in the Service Control Tower. Data standards could be an issue, as in multiple organizations, different standards could be in place. Even within organizations, different standards for different IT applications could cause issues. Therefore setting a common data standard is in the interest of all participating parties for successful Service Control Tower development.

When the development team has a clear picture of the business architecture, and the IT and data available, they need to design the IT architecture for the Service Control Tower itself. In this step, the different resources and business processes are linked schematically. The team should determine which programming language and supporting IT tools or platforms are needed when that is done. Last but not least, the first data architecture can be put into practice. Interfaces for data exchange should be developed, and the first databases should be constructed or, in case of only interfacing, linked with one another. The first step for data governance should also occur here; without clear rules in the system for data governance, problems could be expected in future situations. The main result from this level is that a functional IT architecture can link data from multiple systems and prepare the data for usage in the latter stages of the Service Control Tower. In other words, data harmonization and synchronization should be finished at this level.

3.1.2 The Tower

The first 'real' level is the Tower level; at this level, the Service Control Tower mainly has the functionality of a generic Tower, i.e., the system has an overview of the activities happening in the business processes in the supply chain context. One could link this level with the different levels of visibility described by Bhosle (2011). The classification of the level is closely related to the functionality of a decision support system, i.e., a system that helps with decisions in a particular context. However, it has no actual control over the analyzed operations. Liotine (2019) also describes the Control Tower environment as an enabler of monitoring operations. Baumgrass et al. (2014) states that a transportation Control Tower main functionality "facilitates transportation planners with easily monitoring and dispatching transportation resources". Shou-Wen et al. (2013) locates the monitoring of business processes in the information layers of the control tower. In these layers, real-time monitoring of the supply chain takes place. Last but not least, Alias et al. (2014) indicates that "Control towers have been named as the future tool of supply chain monitoring for quite a while."

In our view, monitoring is the first step in the Service Control Tower; it is an essential step and provides an overview of the ongoing operations. However, monitoring should not be the endpoint. Instead, monitoring should be the beginning of optimizing the supply chain and business processes across the organization (s). Therefore, when constructing a Tower, one should keep in mind the following activities to achieve monitor and analyze capabilities in the supply chain.

1. Perform data transformation and data cleaning activities to retain information
2. Extract insights from information into certain business processes

3. Develop (user-centric) dashboards for monitoring
4. Give meaning to measurements in the dashboards
5. Initiate and implement an alert generation and management system
6. Use dashboards within (day-to-day) operations for monitoring and analysis

The foundation of the Service Control Tower provides the basics necessary for data exchange, while the Tower level transforms data into information. Developers should, in this step, consider the data available and the information asked by the people working in the business processes. Next, the people operating the dashboards and people working in the business processes should be considered when developing dashboards, alerts and monitoring reports. Based upon their expertise and results from the data, meaning should be given to these dashboards. If the implementation is done correctly, using these dashboards can improve the running of (day-to-day) operations by monitoring and analyzing. However, workers in the supply chain can only intervene manually, i.e., the Service Control Tower with only Tower functionality cannot intervene directly and automatic in the supply chain operations.

3.1.3 The Control Tower

Intervening with the Service Control Tower into the supply chain is the next step in the process. The difference with the Tower level is that in the Control Tower level, the system has decision-making capabilities that directly intervene instead of indirect. Therefore we define the second level of the Service Control Tower as the level on which the system has multiple levels of autonomous control. That does not mean that the system is fully automated; supply chain players can choose manual control in a specific context. A user of the Service Control Tower is the decision-maker, not the algorithms embedded in the system. This should help to balance the amount of automatizing versus expert interference in the supply chain.

Rustenburg (2016) defines two types of control in a Control Tower. First, pro-active control means deciding upon specific actions upfront before they cause issues in the supply chain. The second one is re-active control, in which performance is checked after a specific time interval and problems are acted upon after checking. Hofman (2014) takes a different approach; in their Control Tower architecture, they define a set of rules which the Control Tower handles. Topan et al. (2020) describes that exception messages are automatically generated in practice, but interventions are still manual. Topan et al. (2020) states that automation could be possible, but only under certain circumstances (i.e., when enough data is available).

As seen in the literature, there are different possibilities to control the underlying business processes related to the Service Control Tower. Different levels of automa-

tion are applied, but also different strategies to controlling the Service Control Tower (e.g., [74], [44]). At the Control Tower level, participants in the collaboration need to undertake the following key activities to have different levels of autonomous control.

1. Determine the level of control necessary or available in a process
2. Create governance and authorization rules for decision-making agents
3. Setup the structure for different levels of (autonomous) decision-making
4. Implement decision making (business) rules in the Service Control Tower
5. Reflect upon the selected level of control and improvements shown in the supply chain

At first, organizations in a supply chain collaboration should determine the level of control they are willing to give, either manually, by confirming a system decision (semi-automatic) or wholly by the system (automatic). Next, the Service Control Tower should have a clear governance structure with authorization functionality to make certain decisions impacting the supply chain. When these are determined, a structure (or architecture) for such a control system should be created and implemented in the Service Control Tower environment. Next, the organizations in the collaboration should implement business rules that automate decisions in the Service Control Tower. Last but not least, one should reflect upon the selected level of control and corresponding improvements shown in the supply chain.

3.1.4 The Service Control Tower

The Service Control Tower level, also known as the last level in Figure 3.1, enables the Service Control Tower to act as a platform that controls the supply chain on servitization principles. In other words, decision-making is based upon service offerings for (end) users in the Service Control Tower. We define the step towards servitization as the last step, in contrast to Verma et al. (2020), which states that the usage of SLA's should take place on Level 2, even before the introduction of decision support of autonomous control. In our opinion, we see servitization only as a possibility when the Service Control Tower has an actual influence on the decisions in the supply chain. Without it, one could not optimize the decision making based upon service contracts. Next to that, the use of service contracts requires a change in business philosophy, one that shifts the organizations from delivering products to delivering services. This change in business philosophy should also introduce the willingness to be paid or penalized based upon service performance.

For the Service Control Tower, the result is that the architecture evolves into a service-oriented architecture. This is in line with the earlier stated architecture by

Topan et al. (2020). SLA's are the core of decision making in the Service Control Tower, and of course, the name for the Service Control Tower was partially based upon. However, in the last step towards a fully functional Service Control Tower, supply chain partners must undertake the following steps to end up with a Service Control Tower.

1. Determine the business processes, goals and models eligible for service contracts
2. Propose, design and agree upon service contracts as a legal basis for performance-based service delivery
3. Implement the Service Level Agreements in the Service Control Tower environment
4. Reconstruct the IT architecture such that the Service Control Tower supports (automated) service delivery
5. Implement the servitization in operations
6. Learn from feedback mechanisms in the Service Control Tower to improve service delivery

First, organizations involved need to determine the processes in the supply chain they want to operate based upon service delivery. One should link this with the individual organizational strategy, which should be the basis of decisions made in this context. If partners in the supply chain agree upon service delivery, the supply chain context with the service delivery performance measurement should be constructed. These Service Level Agreements should form the basis for decision-making in the Service Control Tower. One should keep in mind that this requires a different service-oriented architecture in the system itself. Implementing the servitization in operations and learning from feedback mechanisms should be ongoing while using the Service Control Tower.

To conclude this section, we want to emphasize that the road towards a fully functioning Service Control Tower is challenging. Organizations that collaborate in the supply chain can only develop a Service Control Tower until a certain level. However, this might lead to the underutilization of the IT architecture that has been constructed. On the contrary, developing a well-equipped Service Control Tower could also generate new business opportunities for organizations. We will address the Service Control Tower as a Service in the next chapter. The following section focuses upon the earlier described dilemmas organizations face in developing a Service Control Tower.

3.2 Dilemma's in Service Control Tower development

In the previous two sections, we described supply chain partners' collaboration steps to a fruitful collaboration. We also showed the four levels that could be taken logically when developing a Service Control Tower. In sessions, discussions and workshops within the Marconi consortium, different stakeholders have raised issues and dilemmas they (expect to) face when constructing a Service Control Tower environment. This section discusses these dilemmas and relates them towards the earlier described collaboration tool and Service Control Tower levels.

Pan (2006) already recognizes in his paper that decisions makers consistently face dilemmas in software engineering projects. Organizations could be helped by making them aware of the different dilemmas that are faced in such projects. One of the dilemmas recognized in literature is on usability and security; more usable software usually has fewer hurdles in terms of accessibility because of security [75]. Solving these dilemmas in a supply-chain context demands intensive collaboration and clear communication. Balliet (2010) states that good communication enhances cooperation and might solve these dilemmas. In the first place, dilemmas like the Prisoner's dilemma might seem inefficient and a waste of resources. On the contrary, Scalet (2006) argues that Prisoner's dilemmas should be preserved, as they can create an opportunity for cooperation and guard the ethical norms of the market.

In our opinion, a Prisoner's dilemma should not be preserved in supply chain situations in which optimizations are clear cut. However, we agree on the line that they could provide an exciting opportunity for cooperation. In this chapter, we already explore such dilemmas, such that supply chain players already indicate the potential dilemmas they might face. First, we discuss in Section 3.2.1 the two main types of dilemmas, technical and business dilemmas. Next, we discuss in Section 3.2.2 the dilemmas we recognize, and last but not least link them in Section 3.2.3 towards the collaboration tool and Service Control Tower levels and give potential architectural examples for some dilemmas.

3.2.1 Technical and Business dilemmas

During workshops and discussions, we encountered multiple dilemmas that we can categorize into two categories. The first ones are technical dilemmas. Technical dilemmas we define as dilemmas that generally refer to technical aspects of an artefact of interest, the Service Control Tower in our case. Technical dilemmas usually involve architectural aspects of the hardware or software of the system of interest (e.g. on the semantic standard or data storage). The other type of dilemmas we could distillate are business dilemmas. Business dilemmas are dilemmas encoun-

Dilemma	Option A	Option B	Classification	Step in collaboration tool	SCT Level neccessary
Control Tower Development	In-house	Outsourced	Technical + Business	Partner Selection	0
Ownership	Centralized	Decentralized	Business	Governance	0
Leadership	Lead company in consortium	Independent	Business	Governance	0
Intellectual property	Shared	Partially owned	Business	Governance	0
Data Ownership	Single organization	Distributed	Technical + Business	Governance	0
Cost of Control Tower	Transaction based	Subscription based	Business	Governance	0
Cost calculation	Transparant	Black Box (Individually determined)	Business	Governance	0
Control Tower generalization	Generic	Tailor-made	Technical	Design of physical, finance and information flows	0
Sharing mechanisms	Legal determined	Algorithmic determined	Technical + Business	Value creation and gain sharing	2
API Service Access	Open API	Closed API	Technical	Entry & exit rules	1
Community Building	Open	Closed	Business	Entry & exit rules	0
SLA Execution / Collaboration	Rule-based (Smart Contracts)	Trust-based	Technical + Business	Collaboration agreement	3
Data Standards	Single	Multiple	Technical	Logistics and financial data harmonisation	0
Data Sharing/Gathering	Real-time	On-demand	Technical	Datasharing & IT integration	0
Data Security	Encryption only	Blockchain	Technical	Datasharing & IT integration	0
Data Storage	Centralized	Decentralized/Distributed	Technical	Datasharing & IT integration	0
Data Quality	Cleaned data	Raw data	Technical	Datasharing & IT integration	1
Decision Making	Centralized	Decentralized	Technical + Business	Collaborative planning & control	2
Optimization Techniques	Exact (OR)	Self-Learning (ML)	Technical	Collaborative planning & control	2

Table 3.1: Overview of all Service Control Tower dilemmas

tered in the business processes surrounding the collaboration and creation of an artefact (e.g., ownership and leadership in such a collaboration). These dilemmas are usually faced in the professional activities that supply chain partners undertake to reach an agreement with the help of discussion, co-creation and negotiation.

However, these types of dilemmas do not exist in isolation. It could be the case that a dilemma has elements of both a technical and business dilemma, such that there are three possibilities in the end. Technical, business and technical + business dilemmas. We use these three classifications to categorize the different types of dilemmas we retrieved from supply chain partners.

3.2.2 Dilemmas and architectural examples

During the discussions, workshops and meetings with supply chain partners in the MARCONI project, we retrieved multiple dilemmas (see Appendix A for the methodology), which resulted in 19 dilemmas. First, we give the dilemmas and the options organizations can take, our classification of the dilemma and the linkage with the collaboration tool and Service Control Tower levels. Then, we discuss all of the dilemmas in order as they concisely appear in the collaboration tool. For some, we give architectural examples in the decisions making between the two alternatives. Last but not least, we link all the dilemmas in a picture with the collaboration tool earlier discussed.

Control Tower Development

The first dilemma that supply chain partners encounter is deciding who will develop the Control Tower environment. Usually, this decision already occurs at the first level of the Service Control Tower but can also be revisited at higher levels. The main questions that supply chain partners should ask themselves is whether they want to develop these solutions *in-house* or want to *outsource* them. Some participants in

collaborations would see an opportunity to build software and keep the intellectual property created or gain knowledge as an advantage. On the contrary, software development is expensive and requires specialized knowledge from a business and technical perspective. Moreover, most organizations use software from specialized software companies (e.g., ERP systems, Business Intelligence solutions, et cetera) and have already invested in specific IT solutions. This so-called vendor lock-in might even reduce alternatives in software developers that could be an option for outsourcing.

In the end, organizations in the collaboration need to determine which IT solutions they currently have and which potential they see in developing a Control Tower solution themselves or outsourcing it. These discussions are strategic. Individual organizations need to determine whether developing such a solution is one of the organization's core competencies. However, this does not mean that the decisions are static; different parties can develop different environment parts.

Ownership

The second dilemma is regarding the ownership of the Service Control Tower. This is one of the many governance dilemmas faced by organizations in a supply chain collaboration. When designing the collaboration, one of the first aspects is ownership of the Service Control Tower environment. Two options are available for the ownership of the Service Control Tower; either it can be *centralized* (i.e., there is a single entity or organization that has full ownership), or it is *decentralized* (i.e., multiple organizations own certain aspects or parts of the environment). Both cases have their benefits and pitfalls.

The first option, centralized ownership, can be seen in the architectural example in Figure 3.2*. The example shows that a single organization controls the whole Service Control Tower environment. Ownership of the single central organization can be distributed between the different organizations involved, following a set of rules. The alternative is shown in Figure 3.3. We see a Service Control Tower environment, but it is owned by all the organizations involved, based on the environment's aspects. A benefit of centralized ownership is the stability; there is no discussion about legal ownership. On the contrary, it could also be a barrier for organizations to collaborate, as they want to keep control of a specific part of their systems.

Leadership

The third dilemma, and again one regarding the governance aspect of the collaboration, is regarding the leadership in a Service Control Tower environment. Leadership is an essential aspect of collaboration, as the leading organization usually sets out the tone for collaboration. Two options are available for supply chain partners, *a*

*See https://pubs.opengroup.org/architecture/archimate3-doc/ for the latest Archimate specification on how to read Archimate models

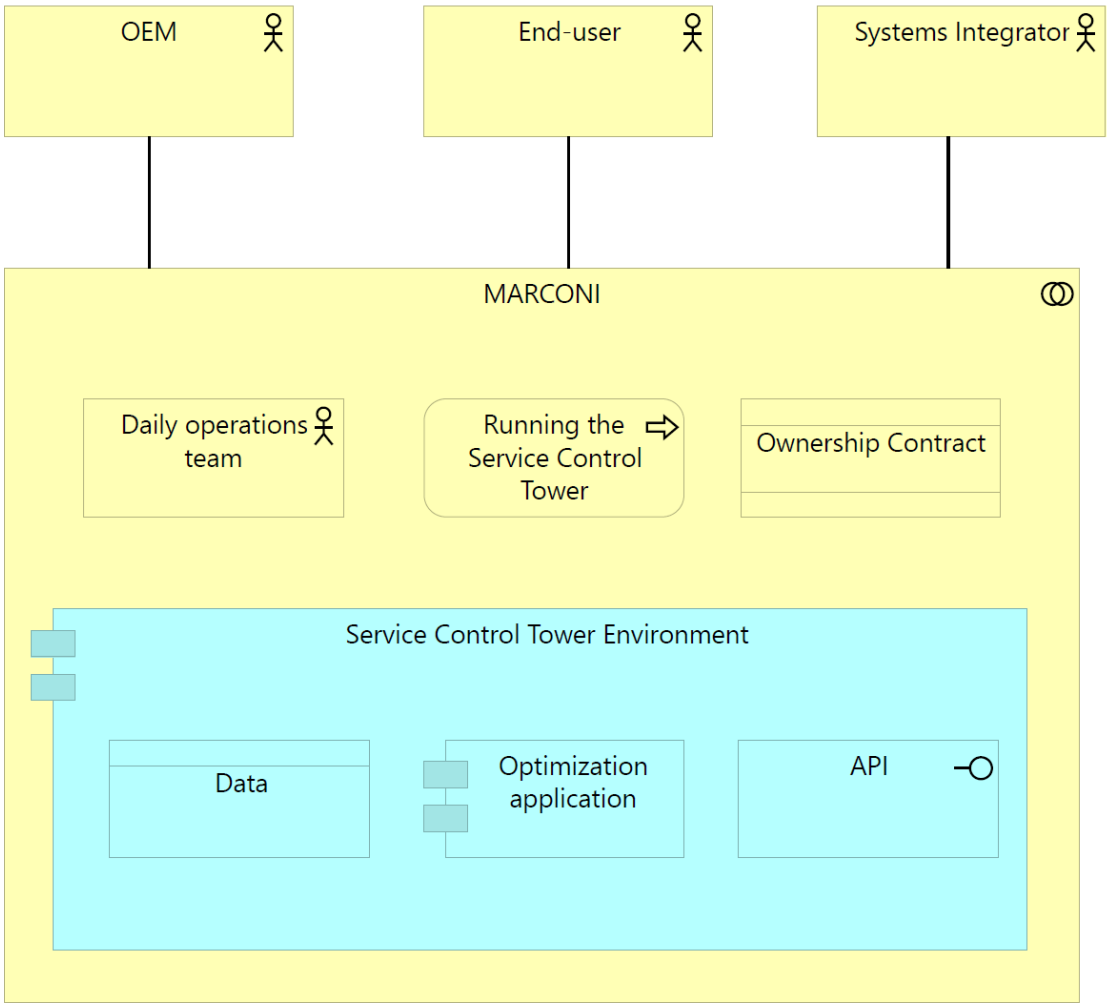


Figure 3.2: Centralized ownership example.

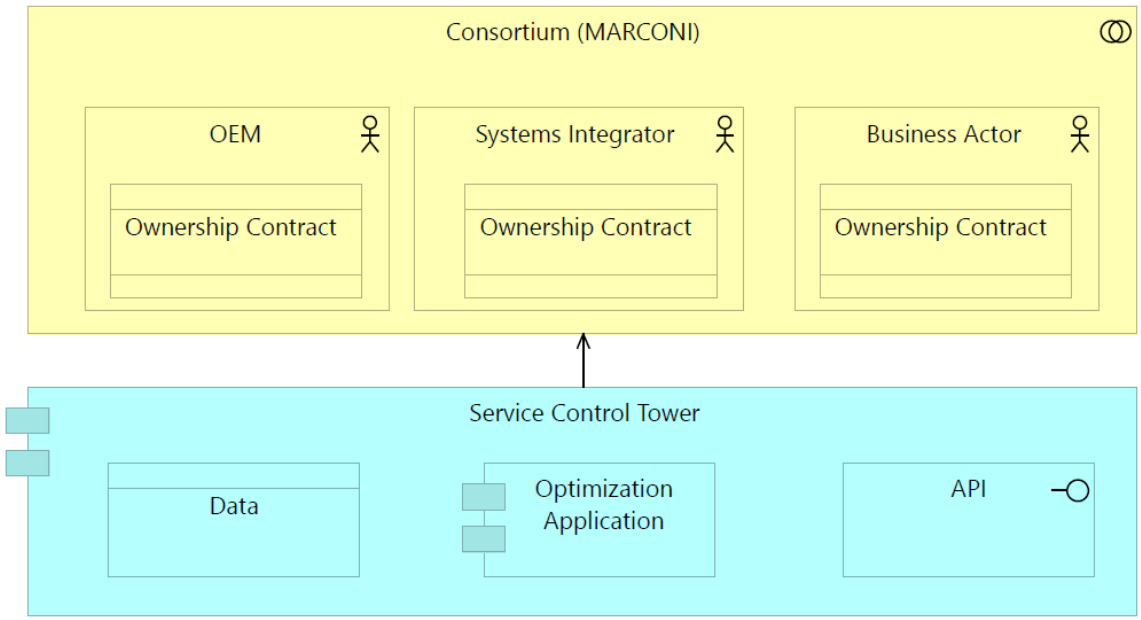


Figure 3.3: Decentralized ownership example.

leading organization is assigned, or leadership is more on an *independent* and case to case basis. Selecting a leading organization creates a clear responsibility for that same organization but could also cause a conflict of interest. Usually, organizations in a supply chain collaboration should select a lead organization that has no conflict with the consortium's interest as a whole but also not with individual organizations in the consortium.

Setting a more independent modus operandi in terms of leadership could strengthen the individual qualities of organizations involved. However, deeming the role of leadership as more organic could also cause problems in terms of responsibility. Ill-defined leadership could result in discussions about accountability and reliability. Organizations in a collaboration should get these discussions on in an early stage of the Service Control Tower design process.

Intellectual property

The dilemma on Intellectual Property is a dilemma related to the ownership dilemma described earlier. However, this dilemma is broader because knowledge is created on aspects related to the collaboration. Therefore, a consortium should define upfront which intellectual property from or brought in during the collaboration is *shared* or *partially (individually) owned*.

The first case is that intellectual property is shared within partners taking place in the collaboration. In such a situation, the sharing of intellectual property is on the liberal side, granting access to all partners' knowledge. On the contrary, partially sharing intellectual property is more defensive, with the hindsight of keeping a

knowledge advantage. However, both options are not exclusive; supply chain partners can opt for different approaches in the collaboration on a case to case basis. To conclude, before the Control Tower is constructed, the consortium needs to decide how the sharing of intellectual property is taking place.

Data Ownership

Data Ownership is a hot topic, as data are seen as a new resource for future growth and the key to the Industry 4.0 revolution. However, participants in collaboration often hesitate to share data between partners partially because they are afraid to lose ownership of their data and confidentiality. The discussion of who owns which data in a supply chain collaboration is essential for a successful collaboration. Van Alstyne et al. (1995) describe that many centralization and standardization projects have failed because of the data ownership issue. Rewarding individuals for creating and providing data is required for sharing; giving individuals data ownership is the most optimal way to provide them with an incentive to do so [95]. A potential way to distinguish the ownership of data is given by [101]. Participatory data is data for which multiple organizations have contributed in one way or another. Non-participatory data is data for which only a single organization has contributed. Participatory data should have shared ownership based upon the contribution of individual organizations. Non-participatory data should stay owned by the organization from which the data originates [101].

We see two possible data ownership decisions. One is *single ownership*, in which only a single organization is the owner of the data. Therefore, access to the data should be provided by the organization controlling the ownership. The alternative is a *distributed data ownership* in the collaboration. Different organizations own a part or parts of the data. Again, these two alternatives do not need to be exclusive, i.e., an organization can choose to own some data together (e.g., participatory data) while keeping specific data within a single organization (e.g., non-participatory data).

Cost of Control Tower

One of the two governance dilemma's related to cost is the cost of the Service Control Tower environment. Running and maintaining a Control Tower will not be without financial support. Therefore, the collaboration partners need to determine how participants and users of the Control Tower contribute towards the cost aspect. Usually, this should be decided upon before constructing the environment itself.

We see two potential options for Control Tower users to select. The first one is a *transaction-based* cost model. In this case, users are asked to compensate for particular services or usage of the Control Tower. This could also be in the case of triggering a set of rules based upon optimization. On the other side, *subscription-based* models can create a more stable source of income for the collaboration. In such a model, participants are asked to compensate for the development and

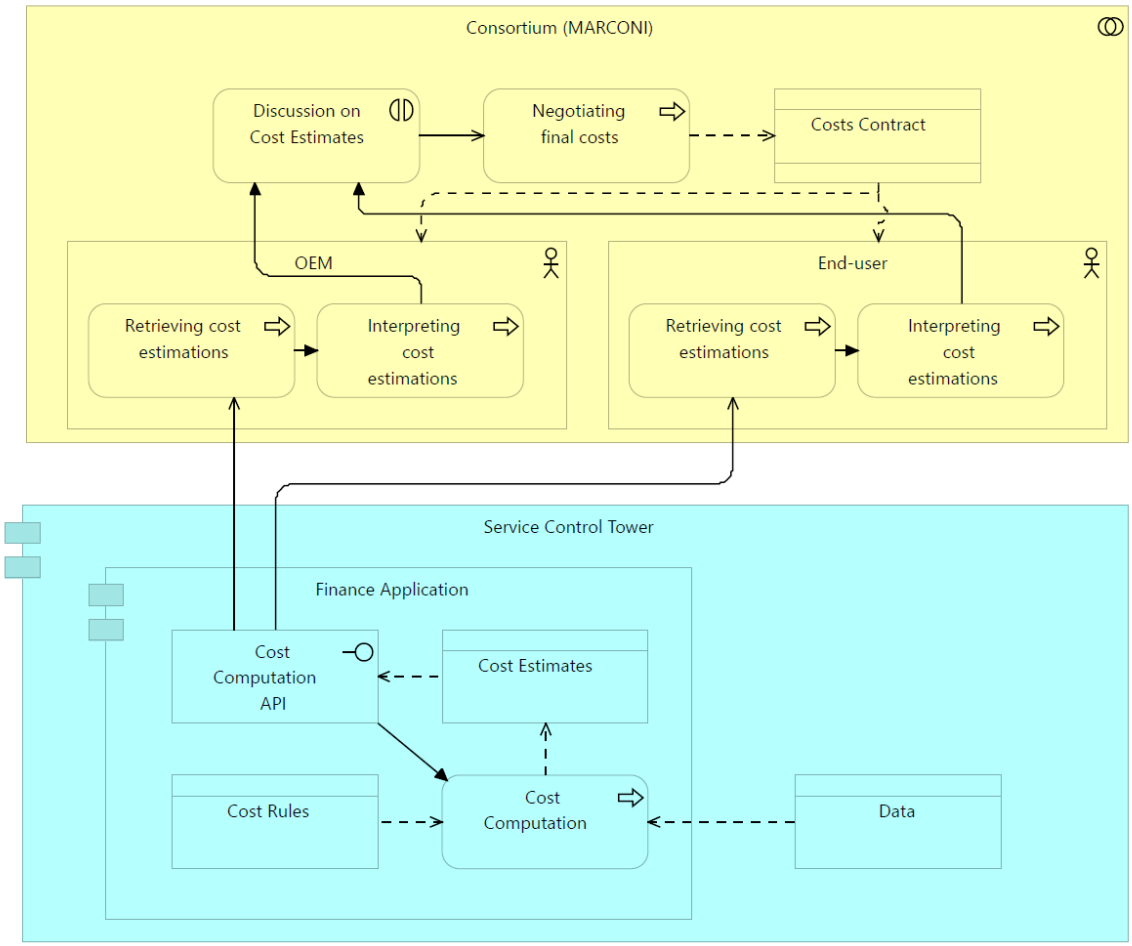


Figure 3.4: Transparent cost example.

maintenance costs of the environment based on a monthly or yearly subscription. Supply chain partners should discuss the way they are compensating for the costs in the collaboration.

Cost calculation

The last dilemma in the governance face of collaboration is the cost calculation dilemma. Running a Service Control Tower environment for without is like having free lunch; both are non-existent. However, when discussing the issue of cost calculation, we recognize two potential solutions. The first one is a *transparent* solution, displayed in Figure 3.4. Here, the rules for the computation of costs are transparent, including precise estimates and rules surrounding the computation of costs. All are accessible by the supply chain players.

The alternative is to have a *black box* computation (i.e., the cost estimation is done by an individual organization which does not give any insight/transparency in the method) of the costs. This we display in Figure 3.5. Here, supply chain players can access the financial data from the Service Control Tower environment

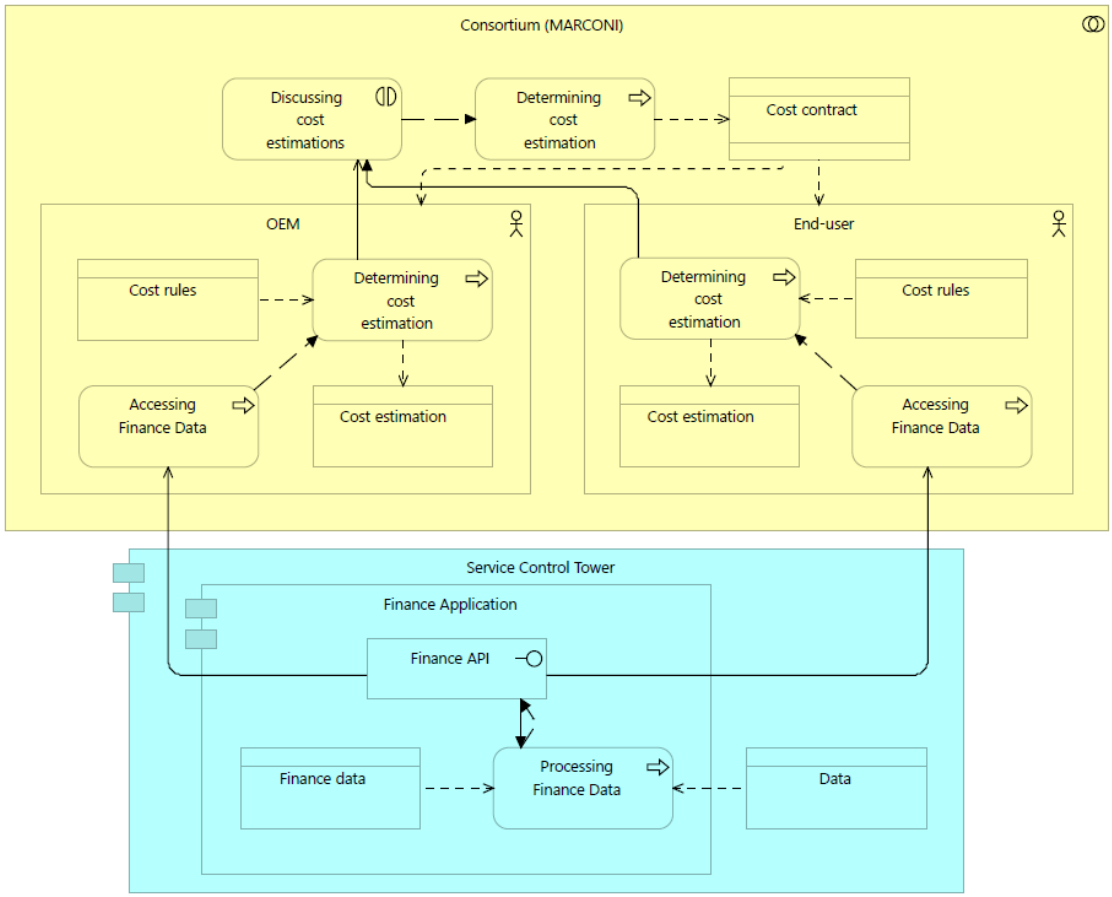


Figure 3.5: Black box cost example.

but compute based upon their own rules and discuss how to settle afterwards. Both have benefits and downsides. However, both are feasible, as long as organizations in the collaboration are clear on their wishes and goals.

Control Tower generalization

The first tactical dilemma not related to governance that we encounter in the collaboration is the generalization of the Control Tower solution. This is a technical dilemma, which is faced while designing the physical, financial and information flows. Currently, multiple IT service providers offer a ready-made environment (i.e., large IT companies that usually offer ERP systems). However, there are also IT companies that offer to build a Service Control Tower based upon the wishes of an individual customer.

Therefore we see two potential alternatives. One is a *generic* solution, which large IT companies usually offer. The benefit of these solutions is that they are tested on a larger scale and generally have good support. The only downside is that the collaboration consortium needs to fit the solution within their set of requirements. The alternative, a *tailor-made* solution, can be made solely in the context and based upon the requirements of the consortium. The downside here is that tailor-made solutions are often more expensive and need more extensive support from the IT software provider. Therefore, the consortium should decide early in the collaboration which solution has their preference.

Sharing mechanisms

If benefits are visible from the interference of the Control Tower in the supply chain, organizations that collaborate want to share these benefits in one way or another. This is usually a dilemma that will be faced in a stage of the Control Tower in which the system has a form of control over the supply chain. Therefore, when the Service Control Tower is only yet at the Tower level, it is not needed to have a sharing mechanism in place.

We see two potential sharing mechanisms; one is a *legally determined* variant, another is based on the *algorithmic determination*. The earlier one is a set of rules into a legally binding contract used when benefits are shared. The latter option is to create an algorithm based on rules to determine the specific amounts shared through the system. Sharing here is not exclusively financial but could also be intellectual property or a data set. A legal determined sharing mechanism would provide supply chain partners with a legally binding option to share benefits, while the algorithmic option is more flexible. We advise organizations in a supply chain collaboration to start discussions before achieving the Control Tower level.

API Service Access

API's have changed the way we build IT infrastructures. As described by [88], the API economy is growing and is becoming "the backbone of the Web, cloud, mobile

and machine learning applications.". There is a high probability that the Service Control Tower will contain API services within a supply chain collaboration. These API services will be available when a Tower has been built (i.e., the environment has visibility in the supply chain). Supply chain players collaborating need to decide whether they open this API for usage by external parties or keep it closed.

The first option, an open API, would offer a unique opportunity for the consortium to offer the environment as a service to external parties. They are also creating a business opportunity for additional revenue. The second option, a *closed API*, creates more stability and security in the system. They keep the services only for participants in the collaboration and reduce overall costs for the system's upkeep. Supply chain partners need to discuss whether they want to (partially) open or close the API in the collaboration tool's entry & exit rules section.

Community Building

The last dilemma faced in the design phase of the collaboration tool for the Service Control Tower is community building. The supply chain players in the collaboration need to determine how the rules of extending or leaving the consortium will occur from a business point of view. Therefore, this must be determined from a business perspective before a Service Control Tower environment is constructed.

Two options are available; there is an *open* community or a *closed* one. The first option means that the consortium has a more transparent approach to newcomers; external supply chain players are invited to join based upon loosely defined rules. Resulting in a more dynamic consortium, but one that can be less stable because of the more liberal approach. On the contrary, a closed community is an option as well. This could create more trust and build better relations between supply chain partners, leading to missed opportunities for adding potential new partners. Again, both options are not exclusive, but the consortium needs to determine upfront which strategy they opt to use for community building.

SLA Execution / Collaboration

A unique dilemma is that of the SLA Execution / Collaboration one. When servitization occurs, organizations usually create Service Level Agreements to store service delivery and compensation rules. This discussion only occurs when a Service Control Tower is at the desired level in Service Control Tower development. We align this dilemma with the eighth step in which companies need to set up a collaboration agreement in the collaboration process. We see again two alternatives that organizations can take to come to SLA execution.

The first one is visualized in Figure 3.6. In this example, *smart contracts* are utilized in the Service Control Tower. A smart contract is based upon a set of rules from a legally binding contract that is automatically executed to confirm these rules.

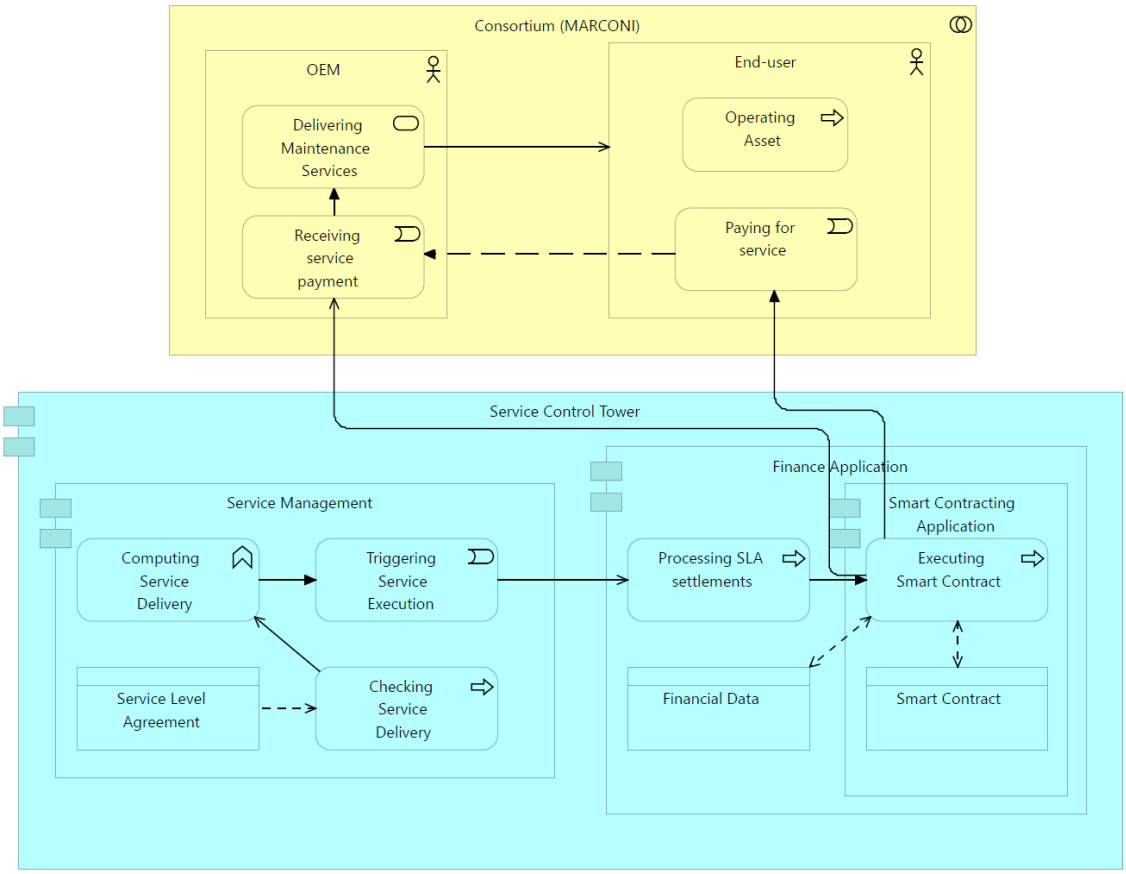


Figure 3.6: SLA Smart Contracting example.

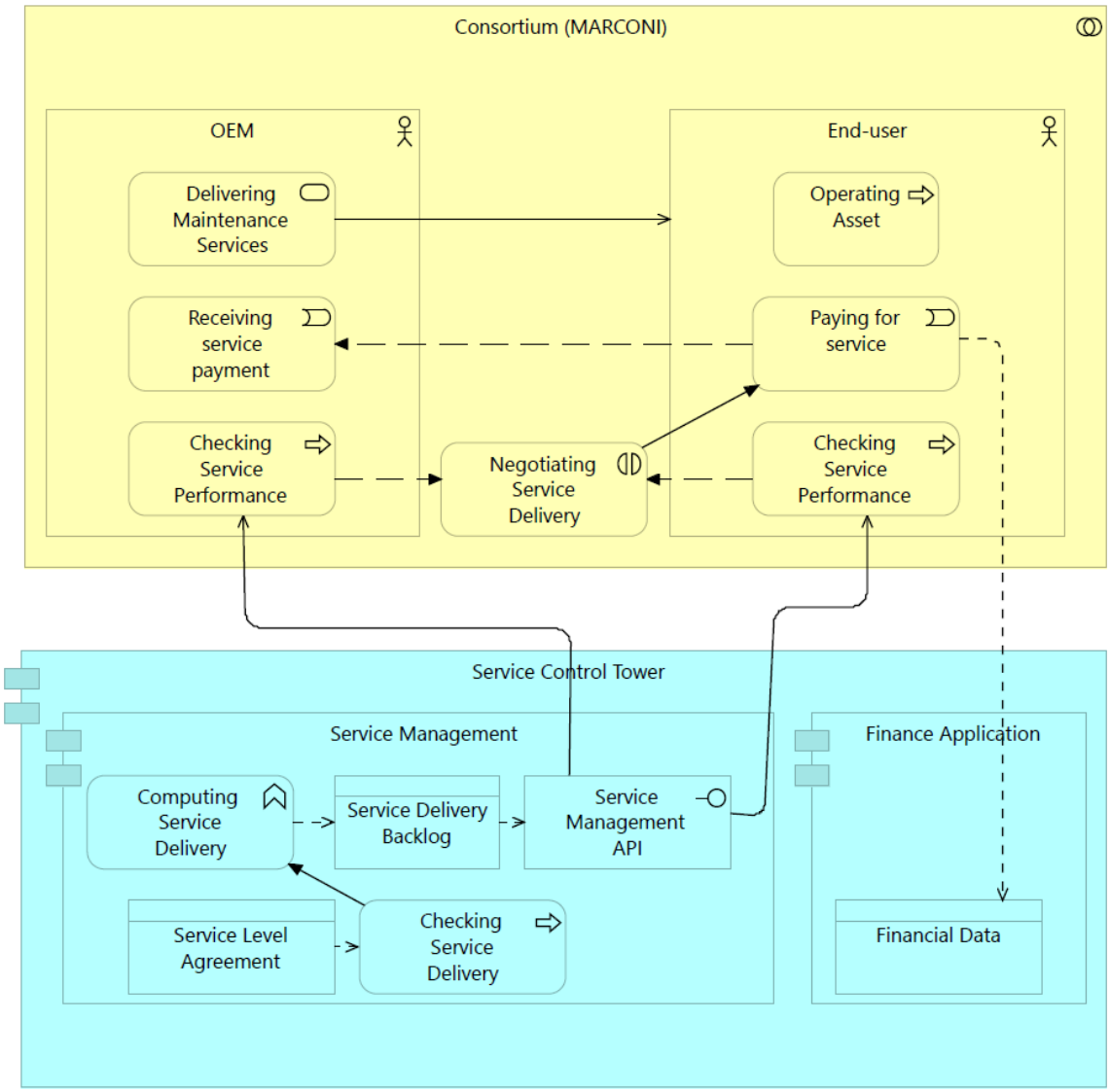


Figure 3.7: SLA Trust based example.

The benefit of such an approach is that it leaves no room for discussion and directly executes the contractual obligations. The downsides are that there is little room for flexibility and negotiations. The alternative, displayed in Figure 3.7, offers this flexibility as it is more *trust-based*. In this case, organizations check for the service performance themselves and negotiate and discuss whether this is sufficient, based upon the Service Level Agreement. The benefit is, as said, flexibility, but could also end up in endless discussions. Organizations need to decide which of these two they give the benefit of the doubt.

Data Standards

When organizations in a supply chain collaboration want to harmonize their logistics and financial data, they must decide upon the data standard(s) they apply. This is a technical dilemma usually faced in the early stage of Service Control Tower development (i.e., at the Foundation level.), which mainly concerns the linking of IT-based upon semantics. In this dilemma, data is critical, but as stated by Groth (2013): "most big data does not come fully formed in one monolithic bloc but is instead smaller pieces combined.". Thus, we see two options that supply chain partners can take. One is based on a single data standard, and the alternative is to have multiple data standards within the Service Control Tower.

Having a *single* data standard has the advantage that the consortium has a common language in data sharing. This reduces potential misconceptions surrounding data but could increase the investment needed upfront to adhere to the single data standard. Having *multiple* data standards offers flexibility, also in terms of the data standards used by individual organizations. On the contrary, they also increase data harmonization that needs to take place behind the scenes. This could, in its turn, lead to a higher cost and misunderstandings because of incorrect transformations. Supply chain partners need to decide in the early stages of the Service Control Tower development which data standard(s) (well-known data standards are open trip, EDIMAR and MTML in maritime) they will adhere to and how they occur.

Data Sharing/Gathering

Data sharing is an important dilemma. As such, we will discuss this further as part of the Data Sharing Dilemma in Chapter 4. However, the dilemma of data sharing is wider. In the phase in which Data sharing IT integration is the main topic, a decision needs to be made about how data is shared. From a theoretical perspective, issues surrounding safe sharing of data are addressed (e.g., [57], [55], [34], [59]). The dilemma faced is whether data sharing and data gathering take place in *real-time* or only *on-demand*.

Real-time data sharing and gathering have the advantage that loads of data can be retrieved instantly but can cause data processing and storage issues. Next to that, internet technology needs to be fast in order to facilitate real-time data sharing.

On the other hand, on-demand data sharing/gathering is much more interval-based and therefore lowers the technical requirements in terms of speed. Nevertheless, it could be problematic if supply chain participants want to track their supply chain in real-time. The trade-off in this dilemma is that the need for real-time data processing outweighs the costs in terms of speed and storage.

Data Security

Data security is a hot topic but also a crucial dilemma to get right. Not having sufficient protection in a supply chain collaboration system could lead to hacks, malware or breaches. Major security issues are data loss, data breaches, malicious insiders, insecure interfaces, hijacking and denial of service [2]. Therefore, data security is a must to invest in, definitely in a Service Control Tower environment.

We see two potential alternatives; one is more of a baseline option, *encryption only*. Whatever is the situation, encryption needs to be part of the system, as essential information and data need to be encrypted. The question is, how advanced does the encryption need to be? Simple encryption algorithms are usually sufficient for base protection. However, if trust in the network is an issue, one could store data securely in the *blockchain*. The blockchain alternative does need a more significant commitment of the consortium to set up but could protect data from security issues. Partners in the supply chain need to discuss which level of security is needed in which context. However, a basic level of encryption should always be the standard.

Data Storage

Data storage is the following dilemma that is faced by organizations collaborating in the supply chain. Data needs to be stored on a piece of hardware, but this piece of hardware could be located anywhere in the world. Therefore we recognize two alternatives for data storage in a Service Control Tower. The first option is *centralized* storage. In this solution, one place contains all server hardware and storage facilities to store the data. The alternative is *decentralized* (e.g. cloud storage). In such a case, data is stored at different locations for different purposes.

For the centralized option, the benefit is that data is physically available in one place, creating easier access and a simpler access mechanism. On the contrary, if the server hosting the data goes down, the whole system goes down. A decentralized option has the benefit that it is much harder to take down at once but also is more vulnerable for attacks and should be managed more thoroughly. Decentralized data storage can be in the blockchain (e.g. [80], [52]), or in the cloud (e.g., [23], [54], [56], [66], [73], [76]).

Data Quality

The last dilemma faced in terms of the Data sharing IT integration is on data quality. Data is usually raw, and the quality of the data is often mediocre. Therefore, supply chain partners need to think about the level of quality necessary, as data quality has

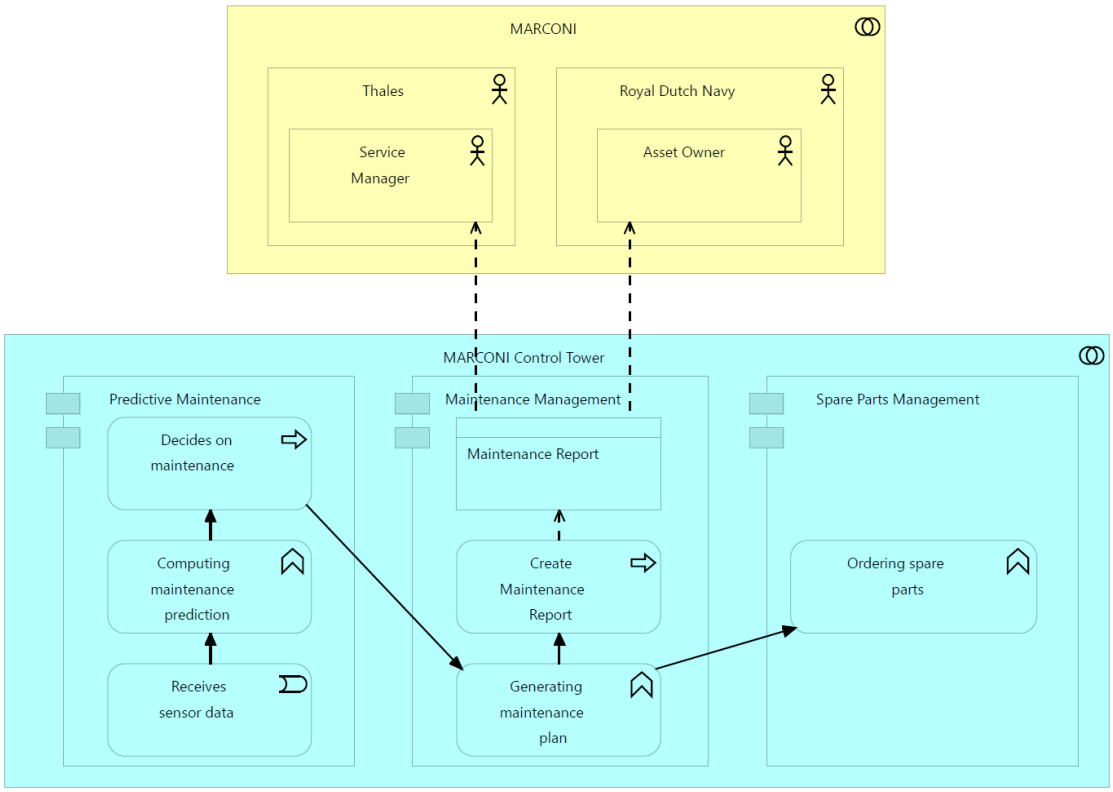


Figure 3.8: Centralized decision making example.

an impact on decision making [25]. Data quality is usually built-up of components like accuracy, completeness, redundancy, readability, accessibility, availability, consistency, etcetera [14]. We define two alternatives for data quality that supply chain partners need to choose.

The first one is *cleaned data*; in this case, organizations on the collaboration will clean their data sources for use in the Service Control Tower environment. As a benefit, this will make life easier in retrieving information from this same source of data. On the contrary, data cleaning is an intensive task, which requires specialized knowledge. The alternative is to incorporate *raw data*, which offers the possibility of looking at pure data instead of cleaned data. Finding outliers could be more accessible when having raw data available, but extracting information needs an additional cleaning step within the system. We expect supply chain organizations to discuss the necessary condition for data quality early in the collaboration.

Decision Making

One of the two dilemmas related to collaborative planning control is decision making. When operating a Service Control Tower environment, a decision needs to be made. However, who is going to make which decision under certain circum-

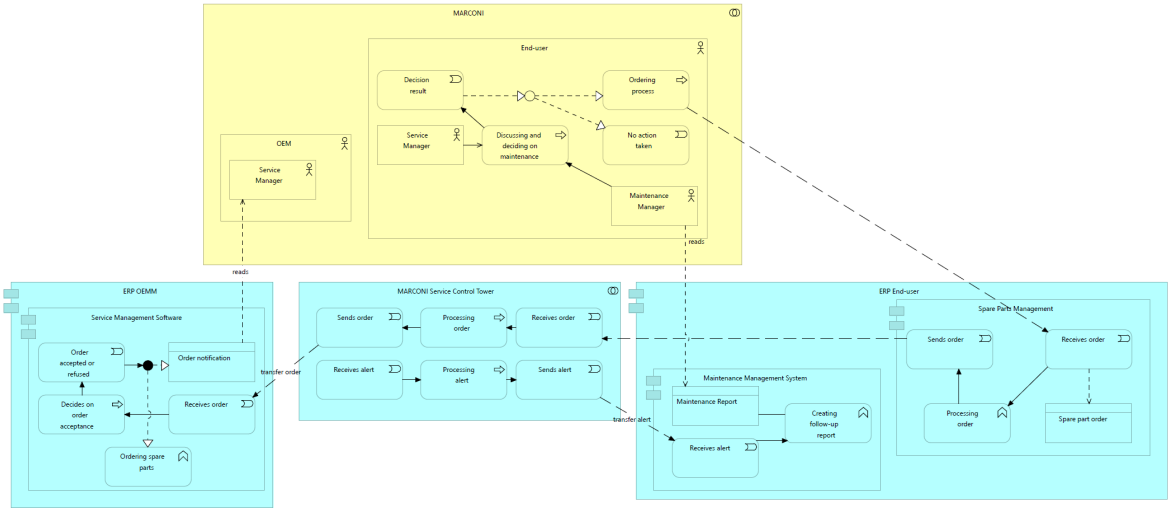


Figure 3.9: Decentralized decision making example.

stances? This could be problematic if no decision is made on this dilemma. Therefore, we recognize two possibilities, *centralized* decision making or *decentralized* decision making.

The first one is centralized decision making, displayed in Figure 3.8. In this case, organizations create a centralized structure in which a single entity, or in this case, software component, decides what is optimal in which situation. Usually, centralized decision making with complete information is optimal; however, a high level of trust and cooperation is needed to set up such a decision-making structure. The alternative is decentralized decision making, pictured in Figure 3.9. Decentralized decision making gives the individual organization more autonomy but could result in a sub-optimal solution. Therefore, organizations need to decide in a trade-off how much autonomy they are willing to give up in order to retrieve better supply chain results coordinated by a central entity.

Optimization Techniques

Last but not least, we discuss the dilemma regarding optimization techniques. When a Service Control Tower has been constructed and control is available in the system, supply chain partners need to decide on the optimization techniques regarding planning and control. In the end, an Service Control Tower environment is there to optimize the service supply chain, so optimization techniques are necessary to implement.

We recognize two options that are dominant in current practices. The first one is related to *Exact methods (Operations Research)*; these are more traditional methods that use linear programming. Operations Research is computationally hard and needs manual adjustment based upon the context applied. However, these algorithms can give exact results. The alternative we see are *Learning algorithms (Ma-*

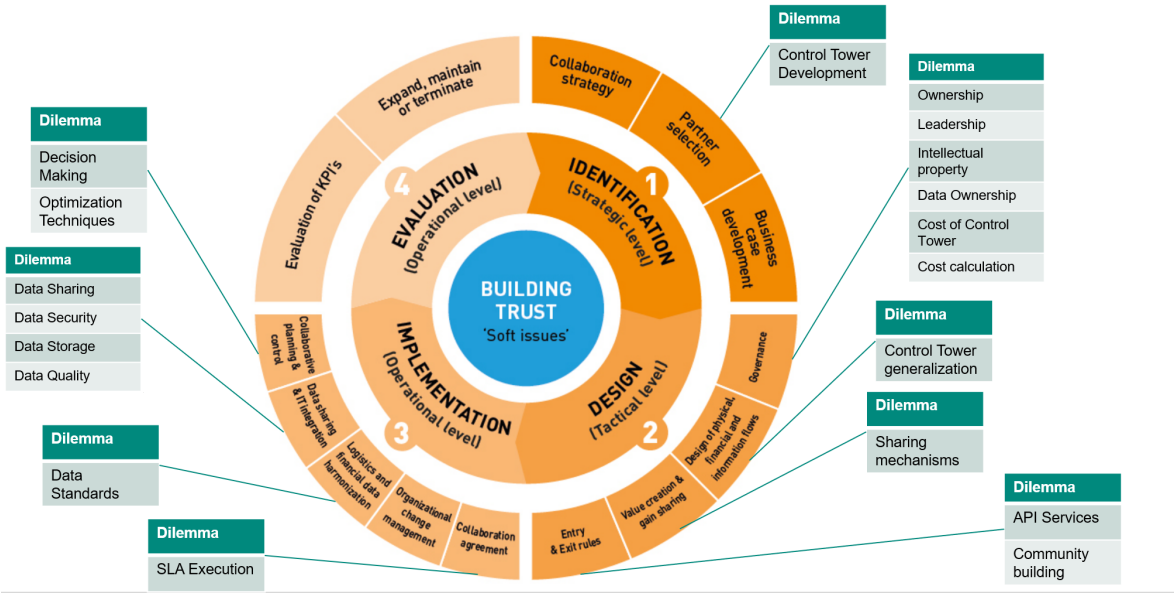


Figure 3.10: Supply Chain Collaboration Tool with aligned dilemmas.

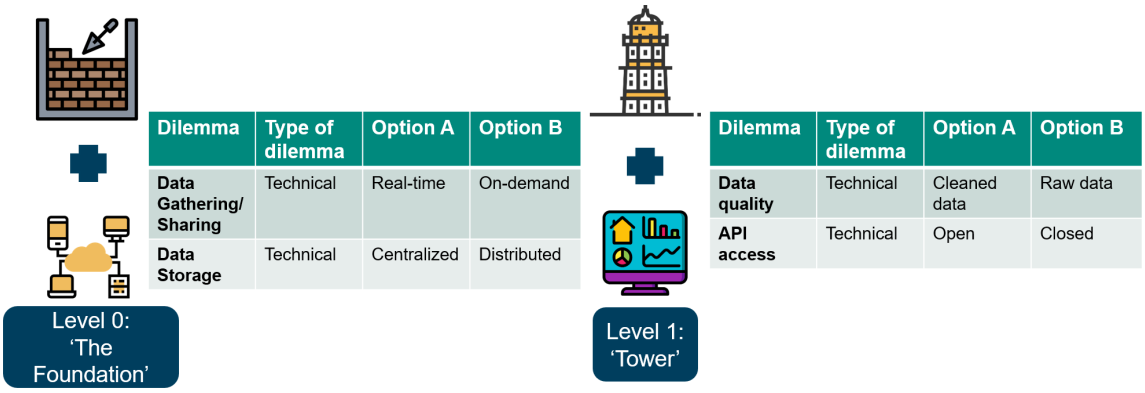
chine Learning); this approach can learn patterns and optimizations from big data sets. Benefits are that these models can learn incredibly quickly but can, on the contrary, get stuck in local optima. Therefore, supply chain partners need to decide in specific contexts which optimization techniques are optimal.

3.2.3 The Collaboration Tool and levels of Service Control Towers linked with the different dilemmas

In this last section, we discuss the link of the dilemmas described in the earlier section. Partially, we have already discussed the link with the collaboration tool provided and different levels of Service Control Towers. However, in this section, we summarize and visualize the link to the collaboration tool and different levels of Service Control Tower.

We start with the collaboration tool, as is visible in Table 3.1; we have 19 dilemmas, aligning with nine different steps of the collaboration tool. One dilemma takes place during the identification phase. Ten dilemmas occur during the collaboration design; the remaining eight dilemmas occur in the third phase, the implementation phase. We have not identified dilemmas in the evaluation phase. The step with the most dilemmas is the Governance step in the Design phase, having six dilemmas from the different steps in the collaboration tool. The second step in the collaboration tool in terms of most dilemmas is Data sharing IT integration. Finally, we visualize the dilemmas linked to the Supply Chain Collaboration Tool in Figure 3.10.

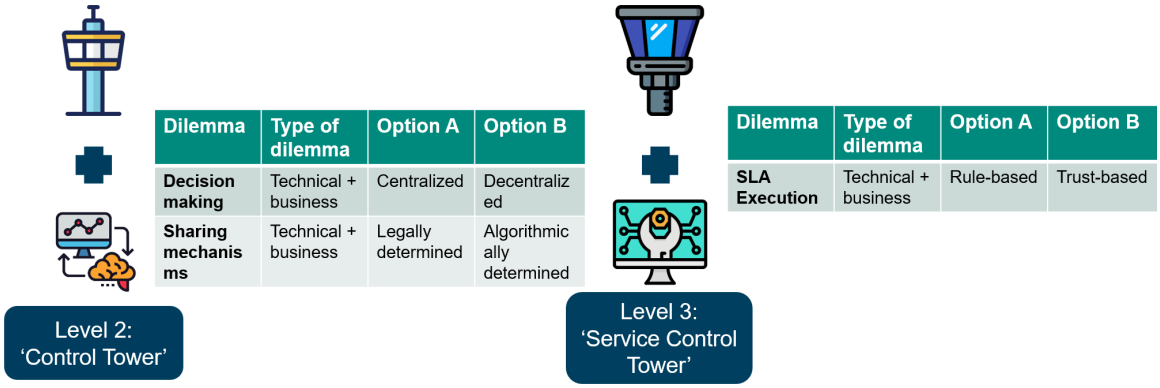
Next, we discuss the dilemmas linked to the levels of the Service Control Tower.



Dilemma	Type of dilemma	Option A	Option B
Data Gathering/ Sharing	Technical	Real-time	On-demand
Data Storage	Technical	Centralized	Distributed

Dilemma	Type of dilemma	Option A	Option B
Data quality	Technical	Cleaned data	Raw data
API access	Technical	Open	Closed

Figure 3.11: Foundation and Tower level dilemmas.



Dilemma	Type of dilemma	Option A	Option B
Decision making	Technical + business	Centralized	Decentralized
Sharing mechanisms	Technical + business	Legally determined	Algorithmically determined

Dilemma	Type of dilemma	Option A	Option B
SLA Execution	Technical + business	Rule-based	Trust-based

Figure 3.12: Control Tower and Service Control Tower level dilemmas.

From the 19 dilemmas, thirteen occur during the first stage of Control Tower development at the Foundation level. Two dilemmas occur at the first level, the Tower level. We display the first two Service Control Tower levels and a selection of their dilemmas in Figure 3.11.

The Control Tower identifies three dilemmas for level two (Sharing Mechanisms, Decision Making and Optimization Techniques). For the last level, the Service Control Tower, we only identify the dilemma of SLA Execution / Collaboration. Again, we visualize these two levels and their corresponding dilemmas in Figure 3.12.

3.3 Conclusion and discussion

To conclude, in this chapter, we have looked at the different design dilemmas that organizations encounter in developing a Service Control Tower environment. First, we discussed the supply chain tool by Dinalog, the different phases in collaboration and the stages that organizations chronologically need to tackle. Next, we stated four levels of the Service Control Tower, starting at the Foundation level and ending up at the complete Service Control Tower. They all have a unique set of activities that organizations need to undertake to construct the desired system of interest.

The central part of this chapter is the different dilemmas that we identified. In total, we recognize nineteen dilemmas that organizations encounter in the process of collaboration and Service Control Tower development. Most of these nineteen dilemmas occur in the early stages of the supply chain collaboration tool. Thirteen of the nineteen dilemmas are faced when at the Foundation level of the Service Control Tower. Therefore, organizations should emphasize the early stages of collaboration, as the most important decisions must be taken there. We discussed the different alternatives for each dilemma, and for some, we presented architectural examples.

However, our dilemmas and this chapter are merely a result of exploratory research within the MARCONI context. In other words, we have only identified dilemmas based upon discussions and a workshop within the current consortium. Next, we only gave four architectural examples on constructing the environment surrounding the dilemmas, while there are also architectural examples for the other dilemmas described. Next, we envision that the dilemmas are not set in stone, i.e., they will change over time based on changing circumstances and supply chain conditions.

Last but not least, we think that a combination of the supply chain collaboration tool, the levels of the Service Control Tower we defined, and the dilemmas explored could help organizations in collaboration structure their discussions and make decisions more balanced. Overall, we predict that if organizations fail to face these decisions early, the collaboration rate diminishes. In the end, collaborations are based upon good communication; this chapter could be the cornerstone of successful collaboration and could help in communication.

Chapter 4

The Data Sharing Dilemma: A Game Theoretical Exploration

4.1 Introduction

Data are seen as the new gold and as one of the drivers of the Fourth Industrial Revolution. However, to obtain the benefits of computational techniques like Machine Learning and Mathematical Optimization, data need to be fed into well-designed algorithms. In practice, sharing data between organizations is heavily debated. Organizations usually protect their interests and business information, but opportunities arise that offer them possibilities to benefit individually from inter-organizational cooperation.

In 1966, Kaufman [53] already envisioned a world in which organizations should use new opportunities like networking capabilities to cross organizational boundaries. Later, this led to the conceptualization of the inter-organizational information sharing system, "a general term referring to systems that involve resources shared between two or more organizations" [12]. However, "both academia and industry observers have long been concerned about the continued slow, painful process and many cases of failure to realize the performance value of IOS (inter-organizational systems) [103]".

This chapter analyses the Data Sharing Dilemma. A game-theoretical dilemma in which two partners in a supply chain may cooperate by sharing data. They need to decide on whether to (not) share data within the collaboration. Their shared data feed a Machine Learning algorithm that should improve the decision-making process and resource allocation in both companies' supply chains. We model this problem as a non-cooperative Frequency-Dependent (FD) game; the reasons for that are twofold. First, a non-cooperative game focuses on the individual rewards of a partner. Second, the Frequency-Dependent aspect can incorporate a learning function. In the end, we use this model to answer the question "Is it individually rational to share

data/knowledge between organizations?".

We start this chapter with an introduction to the relevant literature and context on inter-organizational systems, knowledge and data sharing in a game-theoretical context. We continue in Section 4.3 by introducing our game-theoretical model, which we call the Data Sharing Dilemma. Section 4.4 will focus on an analysis of the results. The last section, Section 4.5, will discuss the results of this chapter.

4.2 From the adoption of inter-organizational systems to Data Sharing

The adoption of inter-organizational systems seems to be problematic. Chen et al. (2014) describe that inter-organizational knowledge sharing is critical to maximizing operational benefits. However, knowledge sharing can only occur when there is trust by building a long-term partnership [24]. Therefore, constructing an environment with a high level of trust is crucial to creating an inter-organizational system.

Trust can be divided into two aspects, the need for trust and the level of trust. Gallivan & Depledge (2003) state that many organizations experience a gap between these two [40]. Bridging this gap is essential to adopt inter-organizational systems and to maximize operational benefits. Therefore we look at the different levels of trust that one could achieve. Panteli & Sockalingam (2005) describe three levels of trust. Starting at the calculus-based trust, going to knowledge-based trust, and finally, identification based trust [69]. Calculus-based trust can be seen as an introductory level on which both actors have computed the costs and benefits. If the benefits are larger than the costs, one could assume an incentive to explore the relation from which data sharing could result.

Partners in a supply chain should be willing to share information, but they might have strategic motives to protect data in reality. Premkumar (1995) states that trust and true partnership should be the solution to this [71]. However, in the end, the benefits should outweigh the downsides for supply chain partners when adopting an inter-organizational system. For example, Zhang et al. (2016) describe that inter-organizational ICT has a positive relation with supply chain performance while intra-organizational ICT has no direct relation with supply chain performance [103]. On the contrary, Zhu et al. (2006) state that adoption costs are a significant barrier to adoption [104].

Trust seems to be the cornerstone in the process of constructing and adopting an inter-organizational system. As Panteli & Sockalingam (2005) set, calculating the benefits and costs is necessary to start a relationship between partners in the supply chain. The Data Sharing Dilemma should be seen as the first step to providing

the calculus-based trust rationale (i.e., providing the Game Theoretical evidence). Suppose at the forehand; we can analyze this rationale from a game-theoretical perspective. In that case, we should clear the way for a trust-based relationship between two supply chain partners. The Data Sharing Dilemma is the first step in looking into individual players' rational benefits while (not) sharing data.

However, to analyze this situation, we need literature that gives an (empirical) indication (i.e., derived from a practical situation)of the game-theoretical context. Arsenyan et al. (2015) use a Nash Bargaining approach to incorporate aspects like trust, coordination, co-learning, and co-innovation dimensions in a collaborative product development environment [8]. Their models confirm that trust is a significant aspect of these projects. A cooperative approach of collaborating by sharing was taken by Lozano (2012). Their results state that not all organizations benefit equally when sharing information. Organizations that have more value to gain from cooperating usually have a lower value input in the collaboration [60]. This article confirms our earlier assumptions on the issues surrounding the adoption of inter-organizational systems and the role of trust. However, for the empirical game theoretical context, we look at other papers.

First, we see that Ho et al. (2011) model the sharing of knowledge as a simple 'share or not share' decision [43]. Shih et al. (2006) model the sharing of knowledge and information as a Prisoner's Dilemma type of game; however, empirical evidence is not provided by the authors [81]. Samieh & Wahba (2007) found empirical evidence that the game can be best characterized as a Prisoner's Dilemma in a situation of sharing knowledge. Therefore confirming the model from Shih et al. (2006). On the contrary, Chua (2003) states that based on his research, knowledge sharing should be represented by the assurance type of game (a particular version of the better known Stag Hunt type *) [26]. The results might be contradicting, but offer us the possibility to further research both effects in this chapter. Therefore we use the earlier described games (i.e., the Prisoner's Dilemma and the Stag Hunt) as a basic building block to design the game payoffs on.

4.3 The Data Sharing Dilemma

In this section, we model the Data Sharing Dilemma. This model is based upon findings in the literature described in the previous section. With the Data Sharing Dilemma model, we should be able to answer whether data sharing is rational. We build this model as a non-cooperative game, more specifically as a Frequency-Dependent (FD) game [49], we explain how this is constructed and under which

*See Figure 4.10

assumptions, first. Next, we build the function that represents a Machine Learning algorithm, which directly influences the payoffs. Finally, the game needs to be analysed, methods that we use for analyses are discussed.

4.3.1 The Data Sharing Dilemma Game

Adopting the inter-organizational system is problematic, and sharing data between supply chain players is essential to adopt such systems. However, no literature is available on sharing data in a context between supply chain players trying to optimize a Machine Learning algorithm. Hypothetically speaking, this Machine Learning algorithm is available to both players and optimizes their decision-making by learning from the shared data of the players. However, the prerequisite is that an inter-organizational system should be in place, including the Machine Learning algorithm, which both players can use. We recognize that without adopting such an environment, sharing of the data cannot occur. In other words, if the players are not using the system, they will also not share data with the system. On the other side, if we can already indicate what behaviour users of such a system in terms of sharing will depict, adoption may (not) be accelerated based upon the perceived benefits. Additionally, the trust could be created by analyzing this context, which we could link to calculus-based trust being the first step, as recognized by Pantelli & Sockalingam (2005) [69].

We model the situation as a non-cooperative game, as it allows us to analyze this situation from an individual rational perspective only. Therefore, the assumption is that even after constructing an inter-organizational system, two supply chain players will decide whether to share data. A one-shot non-cooperative game would imply that both players have exactly one moment in time at which they can take the decision. However, in reality, this decision could be taken in multiple instances in time. Hence, we formulate the game as a Repeated Game; thus, players can make decisions over an infinite time horizon.

The problem with the Repeated Game approach is that payoffs in the game are fixed, i.e. the players can only receive payoffs within a specific interval. These payoffs cannot be altered based upon (earlier) decisions by the players. In terms of sharing data with a Machine Learning algorithm, one could argue that if the data are shared with such an algorithm, decision making in the players' context is improved, resulting in more significant payoffs for both players. Similar phenomena were tackled by Brenner & Witt (2003), who came up with the concept of a Frequency-Dependent (FD) game [21]. The foundation of FD-games was further laid by Joosten et al. (2003) [49].

The topic of FD games has seen an expansion that is best described by Joosten

& Samuel (2020), in which Stochastic Games have FD-functions incorporated, also for the transition probabilities between game states [48]. However, with the sake of simplicity in mind, we keep the Data Sharing Dilemma as simple as possible. On the other side, we acknowledge that this could result in oversimplification, which cannot capture all important aspects. Nevertheless, we see this model as a first step in understanding the dynamics in the context that we sketch. According to the typology of Joosten & Samuel (2020), our game is a standard FD game, i.e. the game has Constant Transition Probabilities, and Endogenous Stage Payoffs [48]. The latter is applied to influence the payoffs with the help of a (Machine) Learning function.

To construct the basis for the game, we look at the literature. As described earlier, two dominant types of non-cooperative games are described. One has characteristics of the Prisoner's Dilemma, as defined by Chua (2003). The other could be classified as a Stag Hunt game described by Samieh & Wahba (2007). We use both to provide exciting results, but first, we set the stage for the game.

4.3.2 Mathematical basis

We build up the game with three building blocks that form the basis for the Data Sharing Dilemma. First, we introduce the basic payoffs of the game (i.e., Prisoner's Dilemma Type and Stag Hunt Type). Second, we link the game with a Frequency-Dependent function which represents the Machine Learning function. In other words, the actions that players take in the game ((not) sharing data) influence the amount of data learned by the function. Last but not least, we incorporate a benefit function, which impacts the amount of learning for an individual player based on the origin of the data. A chronological summary of the different building blocks can be found in Figure 4.1.

Building Block 1: Basic payoffs game

We start by designing the basic payoffs of the game. Two players play the Data Sharing Dilemma. Player 1 (2) is the row (column) player and plays strategy π (σ). Both players can randomize and therefore play mixed strategies. The game takes place in a single state FD-game (i.e., a game in which the players operate within one game context). For the rest of the mathematical notation, we adhere to the notations described by Harmelink & Joosten (2020)[42]. We will explain the concepts further in the chapter.

$R_t^k(\pi, \sigma)$, is a stochastic variable depending on the materialised strategy pair (π, σ) at stage t. As said earlier, the Data Sharing Dilemma has two variants in terms of basic payoffs. One that portrays the elements of a Prisoner's Dilemma, the other a Stag Hunt type of game. We assume that it is highly unattractive to share

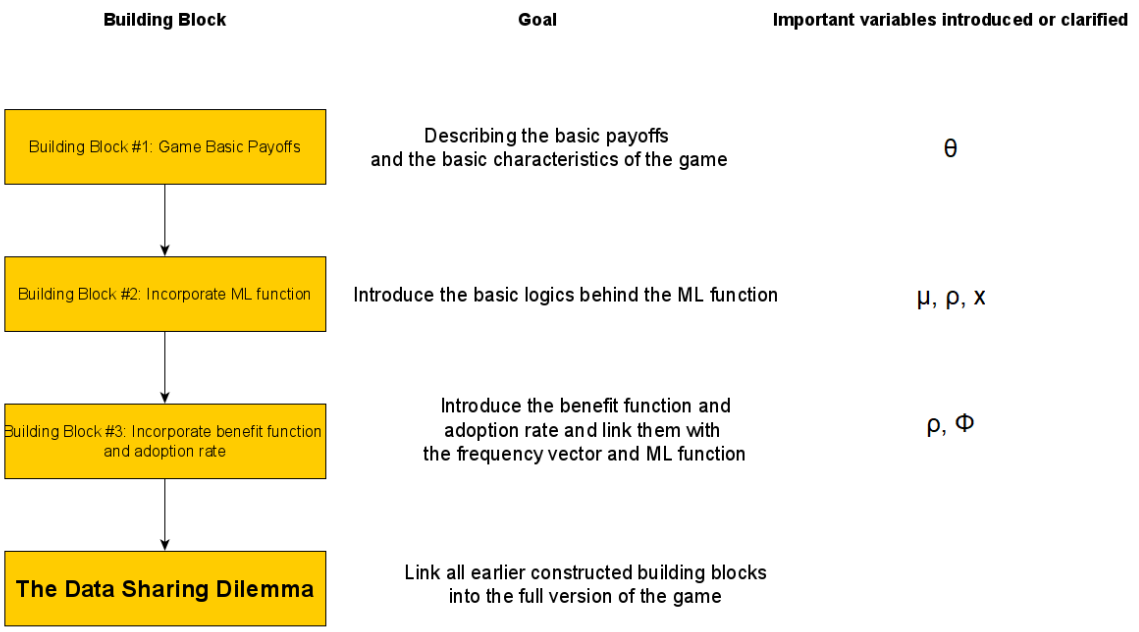


Figure 4.1: Building Blocks of the Data Sharing Dilemma.

Mathematical Letters/Symbols	Description
γ^k	Average reward for player k, $k = 1, 2$
t	Current period in time $t \in \mathbb{N}$
R_t^k	Expected payoff of player k at time t
π	Strategy played by player 1, $\pi \in \mathbb{R}^n$
σ	Strategy played by player 2, $\sigma \in \mathbb{R}^m$
P	Strategy matrix of player 1, $P \in \mathbb{R}^n \times \mathbb{R}^m$
Q	Strategy matrix of player 2, $Q \in \mathbb{R}^m \times \mathbb{R}^n$
T	Total number of periods of time $T \in \mathbb{N}$
x^t	Relative frequency vector at time t
X	Relative frequency matrix
θ_S	Payoff matrix of a stage game
$Pr_{\pi,\sigma}$	Probability under strategy pair π, σ
FD_i^t	FD function at time t with stage game type i
v	Threat point
ρ_{z_k}	The weighted rate of learning, type z for player k
ϕ	Adoption rate of the inter-organizational system
μ	The Machine Learning function accuracy

$$\theta_{S_{PD}} = \begin{array}{c} \\ \text{Share} \\ \text{Not Share} \end{array} \begin{array}{c} \begin{array}{cc} \text{Share} & \text{Not Share} \end{array} \\ \begin{bmatrix} 6,6 & 3,8 \\ 8,3 & 4,4 \end{bmatrix} \end{array}$$

Figure 4.2: Basic payoffs Prisoner's Dilemma version.

$$\theta_{S_{SH}} = \begin{array}{c} \\ \text{Share} \\ \text{Not Share} \end{array} \begin{array}{c} \begin{array}{cc} \text{Share} & \text{Not Share} \end{array} \\ \begin{bmatrix} 8,8 & 3,6 \\ 6,3 & 4,4 \end{bmatrix} \end{array}$$

Figure 4.3: Basic payoffs Stag Hunt version.

data while the other player does not share it at a certain time. Therefore the payoff for a player who shares data while the counterparty is not is the lower boundary (i.e., the lowest possible payoff) of the game, set at 3. If both players are unwilling to share data, the individual payoff will be slightly higher than the previous situation, resulting in a payoff of 4. The differences between the Prisoner's Dilemma game and the Stag Hunt game lie in the payoffs when both persons share and when the player who does not share benefits from the other player sharing. In the Prisoner's Dilemma, not sharing data while the other player does results in the highest payoff, namely 8. In the Stag Hunt, the same situation results in a payoff of 6. The payoffs for the situation in which both players share are then contrasting the earlier described situation. The following payoff matrices (see Figure 4.2 and Figure 4.3) for both game types: player 1 is the row player, and player 2 is the column player.

In game theory, the concept of the Nash Equilibrium by Nash (1951) [67] is an essential method of analysis to describe what rational decision-makers would do. A Nash Equilibrium formalizes that no player can benefit by unilaterally (independent of the opponents' strategy) changing his strategy, i.e. the Nash Equilibrium is for an individual player the best response to all the other players' strategies that are also part of the equilibrium.

Example pure Nash equilibrium Prisoner's Dilemma

In the case of the Prisoner's Dilemma, there is one pure Nash equilibrium. As the game is symmetric, i.e., the game payoffs do not depend on individual players but only on the strategies being played by the players. The result is that payoffs for the players are also symmetric under individual strategy pairs. If the row player decides to share, the column player's best response is not to share. If the row player does not share, the best response for the column player is again not sharing. As a result,

$$\theta_{S_{PD}} = \begin{array}{c} \\ \text{Share} \\ \text{Not Share} \end{array} \begin{array}{c} \begin{array}{cc} \text{Share} & \text{Not Share} \end{array} \\ \begin{bmatrix} 6,6 & 3,8\checkmark \\ \checkmark 8,3 & \checkmark 4,4\checkmark \end{bmatrix} \end{array}$$

Figure 4.4: Basic payoffs Prisoner's Dilemma version with pure Nash equilibrium.

$$\theta_{S_{SH}} = \begin{array}{c} \\ \text{Share} \\ \text{Not Share} \end{array} \begin{array}{c} \begin{array}{cc} \text{Share} & \text{Not Share} \end{array} \\ \begin{bmatrix} \checkmark 8,8\checkmark & 3,6 \\ 6,3 & \checkmark 4,4\checkmark \end{bmatrix} \end{array}$$

Figure 4.5: Basic payoffs Stag Hunt version with pure Nash equilibria.

the column player will always opt not to share, no matter what the row player does. As the game is symmetric, both players' best pure response is not to share still, resulting in a pure Nash equilibrium (Not Share, Not Share). We visualize the best responses in Figure 4.4 with a ✓.

Example pure Nash equilibria Stag Hunt: In the Stag Hunt case, we again have a symmetric game, but in this case, the payoffs are different, resulting in another interaction. When a player decides to share, the best response is also to share. When a player chooses not to share, the best response is also not to share. Therefore this game has two pure Nash equilibria (Share, Share) and (Not Share, Not Share). Having two pure Nash equilibria further complicates the game dynamics, as the Nash equilibrium conditions make both results feasible. However, the Nash equilibrium at (Not Share, Not Share) is risk dominant, i.e., if a player decides to go for share as an action, there is a possibility that the opponent chooses to not share, resulting in a lower payoff for the sharing player. A safer option would be to always share in such a case, i.e., risk dominant.

As can be derived from Figures 4.4 and 4.5, the pure Nash Equilibria (NE) differ for both types of games. For the Prisoner's Dilemma, the unique pure NE is that both players do not share. For the Stag Hunt game, there are two pure NE, one when both players share and one when both players do not share. The pure NE when both players share is Pareto efficient (i.e., no player can gain more without the direct loss of another player). However, there is also a mixed equilibrium when both players share with probability $\frac{1}{3}$ and do not share with probability $\frac{2}{3}$. Although the payoffs seem fixed, by introducing a Frequency-Dependent function later in this chapter, we turn the game into an FD-game, while in the current form, it would have been a Repeated Game. By turning the game into an FD game, the amount of sharing is captured by an FD function and influences the payoffs. The FD function

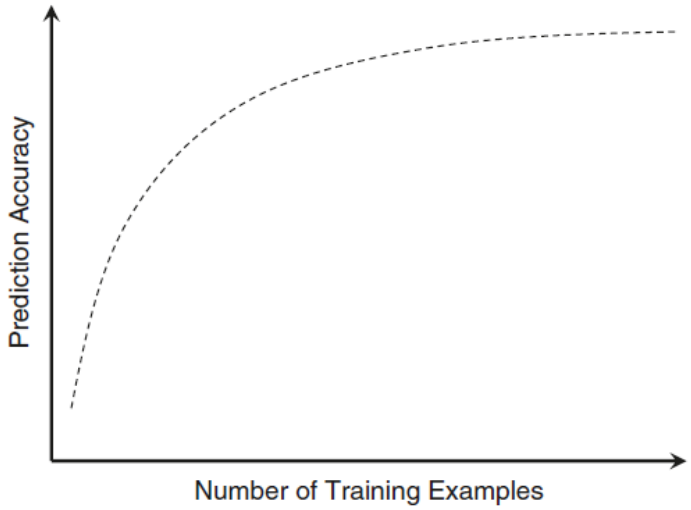


Figure 4.6: Conceptual Learning Curve in Machine Learning [70].

is, in our case, a representation of a Machine Learning function.

4.3.3 Construction of the Frequency-Dependent Functions and Parameters

Building Block 2: Incorporate Machine Learning Function

This section describes the Frequency-Dependent functions as building blocks that will alter the payoffs described previously. As conceptualized, the Data Sharing Dilemma represents the situation in which two players may share data directly into a Machine Learning algorithm that optimizes both decision-making.

Learning curves have different representations. Nevertheless, most follow a generic pattern. The learning curve was early described by Hermann Ebbinghaus in 1885 and can be at best conceptualized with the following image [100].

However, we should turn this conceptual function into one that we can represent with a mathematical function and use within the game. Multiple candidates are possible, but we opt for the following as the proposed function can change the slope. The learning function is represented by $\mu(\rho)$, in which ρ is the weighted learning rate (i.e., the amount of data from which the algorithm has learned, so more sharing of data ends up in more learning).

$$\mu(\rho) = b(1 - \frac{a}{\rho + a})$$

Under which variable a influences the curvature of the function and variable b, which gives the asymptote's height. A lower value for a increases the steepness of the curvature for low values of ρ. A higher value of a will result in a more straight

line. We envision that this function should be within the following limits for further use within the Data Sharing Dilemma.

$$0 \leq \mu(\rho) \leq 1$$

$$0 \leq \rho \leq 1$$

We develop the game and the game functions as such that these limits should be incorporated by design. Important for that are the variables a and b. Therefore, we set additional constraints on the values of these variables. The function always reflects a learning function with a similar characteristic curve as in Figure 4.6. Therefore we impose the following constraints, such that[†]:

$$b > 0$$

$$b - a = 1$$

We use the function $\mu(\rho)$ in the Data Sharing Dilemma. The following example will show Figure 4.6 represented by the function $\mu(\rho)$.

Example Learning Curve Function $\mu(\rho)$: For this example, we take $a = 0.1$ and $b = 1.1$. Resulting in the following function:

$$\mu(\rho) = 1.1(1 - \frac{0.1}{\rho + 0.1})$$

As seen in Figure 4.7, the curvature of the function is similar to that in Figure 4.6. In this case, ρ is the weighted learning rate, depending on the amount of shared data (we will explain this later). If $\rho = 0.1$, then 10% of the total data available is learned. This results in $\mu(0.1) = 0.55$, which means that the accuracy of the Machine Learning algorithm is 55%. For $\rho = 0.9$, i.e. 90% of the data is learned, results in $\mu(0.9) = 0.99$, i.e. the Machine Learning accuracy has reached 99%.

A better and more accurate Machine Learning algorithm could help supply chain players better allocate (individual) resources in the supply chain. The Learning Curve in Figure 4.7 is not directly suitable for use within the Data Sharing Dilemma. In the Data Sharing Dilemma basic game, players only have the option to share or not share. Their designs influence the amount of data being shared with the Machine Learning algorithm. However, we need to track the frequency of the number of times players choose to share or not share their data.

Therefore we introduce the relative frequency vector $x = (x_1, x_2, x_3, x_4)$. The relative frequency vector represents the relative frequency of play a game ends up

[†]In reality, achieving a perfect accuracy in real-life Machine Learning applications is rarely feasible, however, the accuracy can get close to 1 (or 100%). Therefore we opt to set 1 as the asymptote.

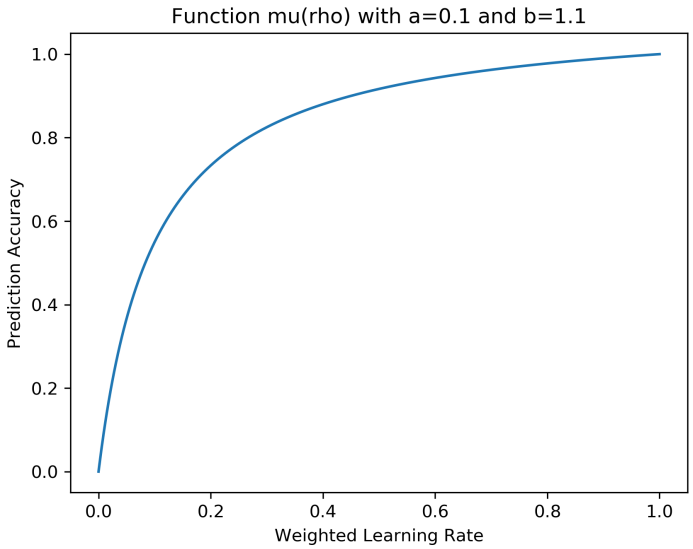


Figure 4.7: The Data Sharing Dilemma Learning Curve represented by $\mu(\rho)$ with $a = 0.1$ and $b = 1.1$.

	Share	Not Share
Share	x_1	x_2
Not Share	x_3	x_4

Figure 4.8: The relative frequency vector in the game matrix.

at a certain action combination. In case of the relative frequency vector, it should always hold that $\sum_{i=1}^{4} x_i = 1$ with $x_i = [0, 1]$. As an example, we show how the relative frequency vector relates to the payoffs of an FD-game. In Figure 4.8 we see that x_1 corresponds to the payoff for the result (Share, Share).

Example Relative Frequency Vector: By example, we can show how the relative frequency vector works. We assume the Data Sharing Dilemma has been played; thus, the play ended up for 60% of the time in (Share, Share). Action pair (Share, Not Share) occurs for 5%, action pair (Not Share, Share) for 10% and the resulting 25% the play ended up in action pair (Not Share, Not Share). We display the result in matrix form in Figure 4.9. The resulting relative frequency vector is $x = (0.6, 0.05, 0.1, 0.25)$.

Building Block 3: Incorporate benefit functions

The last step is transforming the relative frequency vector into a single variable representing the weighted rate of learning ρ. We envision two possible scenarios in which players in the Data Sharing Dilemma experience benefits. The first one we call Mutual Benefits (represented by ρ_{mb}). In Mutual Benefits, both players benefit more from sharing both than in cases where an individual player can train the algorithm

$$\begin{array}{r} \\ \text{Share} \\ \text{Not Share} \end{array} \begin{array}{c} \begin{array}{cc} \text{Share} & \text{Not Share} \end{array} \\ \begin{bmatrix} 0.6 & 0.05 \\ 0.1 & 0.25 \end{bmatrix} \end{array}$$

Figure 4.9: The relative frequency vector result.

on only his data. We could state that in this situation, the data from both players are complementary. However, they are contrasting; in the case of Singular Benefits (represented by ρ_{sb}), a player benefits more if the model learns his data more than if both players are sharing the data. In the latter case, one could argue that data has more individualistic value. The only common denominator for both situations is that the algorithm will not learn when both players decide not to share any data at all.

The logic of the Mutual Benefits function for player 1: If none of the players is sharing any data, this will not train the learning algorithm whatsoever. Resulting in the following learning rate for player 1 for the action pair (Not Share, Not Share)

$$\rho_{mb_{p1}} = 0x_4$$

However, if the counterparty decides to share his data, but player 1 refuses to, this is valued higher but still relatively low.

$$\rho_{mb_{p1}} = \frac{1}{3}x_3$$

Suppose player 1 decides to share his data, but the counterparty refuses to. In that case, this is more beneficial to player 1, as the Machine Learning algorithm learns his data, which relates more to the decision making context of player 1. Such that.

$$\rho_{mb_{p1}} = \frac{2}{3}x_2$$

Most beneficial for player 1, in the case of Mutual Benefits, would be the sharing of data by both players. As a result, this is valued maximally, resulting in.

$$\rho_{mb_{p1}} = x_1$$

The logic of the Singular Benefits function for player 2: In the other situation, the case of Singular Benefits. Player 1 always prefers the situation in which his data is learned but not the counterparty's data. For the rest, the Mutual Benefits function's values apply, which results in the following changes.

$$\rho_{sb_{p2}} = \frac{2}{3}x_1$$

$$\rho_{sb_{p2}} = x_2$$

Based upon the earlier mentioned logic and the relative frequency vector x, we can construct the benefit functions.

$$\rho : \Delta^3 \rightarrow [0, 1]$$

For player 1 the benefit functions are defined as:

$$\rho_{sb_{p1}} = \frac{2}{3}x_1 + \frac{1}{3}x_2 + x_3$$
$$\rho_{mb_{p1}} = x_1 + \frac{1}{3}x_2 + \frac{2}{3}x_3$$

And for player 2 (as the game is symmetric) as:

$$\rho_{sb_{p2}} = \frac{2}{3}x_1 + x_2 + \frac{1}{3}x_3$$
$$\rho_{mb_{p2}} = x_1 + \frac{2}{3}x_2 + \frac{1}{3}x_3$$

The results of the benefits function influences the amount of learning taking place for each player. Incorporated within the learning function, it takes the following form:

$$\mu(\rho_{m_k}) = 1.1(1 - \frac{0.1}{\rho_{m_k} + 0.1})$$

Example Singular Benefits Player 1: In this example, we show the effect of the relative frequency vector and the benefit function into convergence for the learning curve. Assume the relative frequency vector is $x = (0.6, 0.05, 0.1, 0.25)$. Then filling in $\rho_{sb_{p1}}$ results in:

$$\rho_{sb_{p1}} = \frac{2}{3}0.6 + \frac{1}{3}0.05 + 0.1$$
$$\rho_{sb_{p1}} \approx 0.51666$$
$$\mu(0.51666) = 1.1(1 - \frac{0.1}{0.5166 + 0.1})$$
$$\mu(0.51666) \approx 0.92162$$

However, we should link these functions now to the actual payoffs. They are bringing us to introducing the last (exogenous) parameter in this game, namely ϕ, representing the number of initial benefits from an inter-organizational system. One could imagine that even if an inter-organizational system is constructed, the benefits start to increase after an adoption within the organizations. We assume that adoption is an organizational process, not directly resulting from the dynamics of sharing

data itself. As an example, if no inter-organizational adoption takes place, we could say that $\phi = 0$, while if the process of adoption is half-way $\phi = 0.5$. Full adoption would be represented by $\phi = 1$. Adoption rates could be low in the early stages, resulting in lower system benefits, gradually increasing if ϕ increases. We represent this with ϕ:

$$\phi \rightarrow [0, 1]$$

The Data Sharing Dilemma Combining all the above-described functions results in the construction of the Data Sharing Dilemma. However, this chapter's Data Sharing Dilemma is not one game; it describes different possible games that contain standard building blocks. However, each combination could provide exciting results in the analysis. For now, we combine the building blocks to construct the Data Sharing Dilemma in the FD-game form.

$$(\phi + \mu(\rho_{m_k}))(\theta_{S_l} \cdot x)$$

In other words, the basic payoffs in the Data Sharing Dilemma are represented by θ_{S_l} in which S_l is the type of payoffs (e.g., Stag Hunt or Prisoner's Dilemma). The rewards can be amplified by raising the adoption rate ϕ (i.e., $\phi = 0$ means that players do not receive additional rewards, contrasting, if $\phi = 1$ this is multiplied with the rewards also depending on μ in both situations) or letting the Machine Learning algorithm improve its accuracy represented by $\mu(\rho_{m_k})$. m_k is the type of benefits function active for player k (i.e., if both players share under mutual benefits $\mu = 1$ for both players giving them higher rewards, and possibly even higher rewards when $\phi = 1$ resulting in double the payoffs).

Example Data Sharing Dilemma: Let's assume we have the following situation. The Data Sharing Dilemma has the payoffs of the Prisoner's Dilemma type, both players experience Singular Benefits and the adoption rate $\phi = 1$. We compute the payoffs for player 1 under $x = (0.6, 0.05, 0.1, 0.25)$. We can continue where we left the last example, with:

$$\mu(0.51666) \approx 0.92162$$

$$(\phi + \mu(\rho_{sb_{p1}}))(\theta_{S_{PD}} \cdot x)$$

$$(1 + 0.92162)(\theta_{S_{PD}} \cdot x)$$

$$\theta_{S_{PD}} \cdot x = (6, 3, 8, 4) \cdot (0.6, 0.05, 0.1, 0.25)$$

$$\theta_{S_{PD}} \cdot x = 5.55$$

$$1.92162 \cdot 5.55 \approx 10.664991$$

The payoff for player 1 of the Data Sharing Dilemma, in the form of Prisoner's Dilemma type with Singular Benefits and adoption rate $\phi = 1$ is 10.664991. First, however, we need to specify how the analysis will take place in the long run.

4.3.4 Method of Analysis

Two rewards methods are dominant in literature, the discounted sum of rewards and the limiting average reward [83]. For this chapter we use the latter one as a basis for our analysis, given by:

$$\gamma^k(\pi, \sigma) = \liminf_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} R_t^k(\pi, \sigma)$$

However, the average reward need not exist. Therefore it is necessary to take the $\liminf$. Finding equilibria under this criterion gives a more explicit representation of strategic decisions between players in the long run by not limiting ourselves with short-term evaluation. However, this comes at a cost. Generally limiting average zero-sum games do not guarantee a game value, making analysis of the equilibria hard. In the case of discounted sum games, equilibria's existence is assured (if, e.g.[83]). Under certain conditions, we can guarantee the finding of equilibria in limiting average games, which turns our attention towards the concept of jointly-convergent pure-strategy rewards.

The concept of jointly-convergent pure-strategy rewards was introduced by Joosten et al. (2003) [49]. In jointly-convergent pure-strategy pairs, a player plays a pure action (i.e. a single-stage move) at any stage t with probability 1. The pure-strategy pair (π, σ) is jointly-convergent for Player 1 (2) with the relative frequency vector x if and only if [49]:

$$\limsup_{t\to\infty} Pr_{\pi,\sigma}[|x_i^t - x_i| > \epsilon] = 0 \tag{4.1}$$

The strategy pair is jointly-convergent if the relative frequency vector converges to a vector containing fixed numbers with probability 1 when t advances to infinity [18]. The power of these strategy pairs is that we can compute a set of rewards conveniently (e.g., Joosten & Samuel (2020) [48]), which we also use to plot the rewards for the Data Sharing Dilemma. Additionally, two theorems rely on the concept of jointly-convergent pure-strategy pair rewards.

Theorem 1: "Each pair of individually-rational jointly-convergent pure-strategy rewards can be supported by an equilibrium. Moreover, each pair of jointly-convergent pure-strategy rewards giving each player strictly more than the threat-point reward can be supported by a subgame-perfect equilibrium. [49]"

In this theorem, the threat-point reward (i.e. a tuple of minimal guaranteed rewards, indifferent of the opponents' strategies) has an important role.

Theorem 2: "Each pair of rewards in the convex hull of all individually-rational jointly-convergent pure-strategy rewards can be supported by an equilibrium. Moreover, each pair of rewards in the convex hull of all jointly-convergent pure strategy rewards giving each player strictly more than the threat-point reward can be supported by a subgame-perfect equilibrium [49]."

Therefore it is essential to find this so-called threat-point reward, which is defined as:

Definition threat-point:

$$v = (v^1, v^2) \qquad \text{with:}$$
$$v^1 = \min_{\sigma} \max_{\pi} \gamma^1(\pi, \sigma)$$
$$v^2 = \min_{\pi} \max_{\sigma} \gamma^2(\pi, \sigma)$$

The threat-point is a tuple of rewards. An individual reward for a player in the tuple is the minimal guaranteed rewards, indifferent to the opponents' strategies. Retrieving the threat-point, therefore, also indicates the complete set of (Nash) equilibria.

Finding this threat-point is possible, Harmelink & Joosten (2020) constructed an efficient algorithm that works under strict conditions surrounding the non-cooperative FD-game of interest [42]. These two conditions are the following; first, the Markov chain guarding the game's transition probabilities should be ergodic. In the case of this game, we have a single state such that the probability of ending up in the same state is 1; therefore, the game is ergodic by definition. Second, the Frequency-Dependent functions should be monotonic; if this is not the case, the threat-point algorithm's convergence to the threat-point is not guaranteed. Therefore, in the Data Sharing Dilemma, we only use monotonic Frequency-Dependent functions. Concluding, we adhere to the restrictions for the threat-point algorithm and can apply this for analysis.

Last but not least, we have a particular interest in Pareto efficient rewards. Pareto efficient rewards are reward pairs in which there is no improvement in individual rewards possible without reducing the reward of the other player. So a Pareto efficient allocation would be a rewards pair:

Definition Pareto efficient reward pairs:

$$(\gamma^1(\pi, \sigma), \gamma^2(\pi, \sigma))$$

Under the condition there is no alternative allocation:

$$(\gamma'^1(\pi, \sigma), \gamma'^2(\pi, \sigma))$$

Such that for an individual agent i:

$$\gamma'^i(\pi, \sigma) \geq \gamma^i(\pi, \sigma)$$

$\forall i \in 1, 2$ and

$$\gamma'^i(\pi, \sigma) > \gamma^i(\pi, \sigma)$$

For some i

Pareto efficient rewards are attractive as they result in minimal efficient rewards but not stating anything about the desirable outcomes socially. The set of Pareto efficient rewards, also known as the Pareto frontier, indicates trade-offs in strategic environments like the Data Sharing Dilemma.

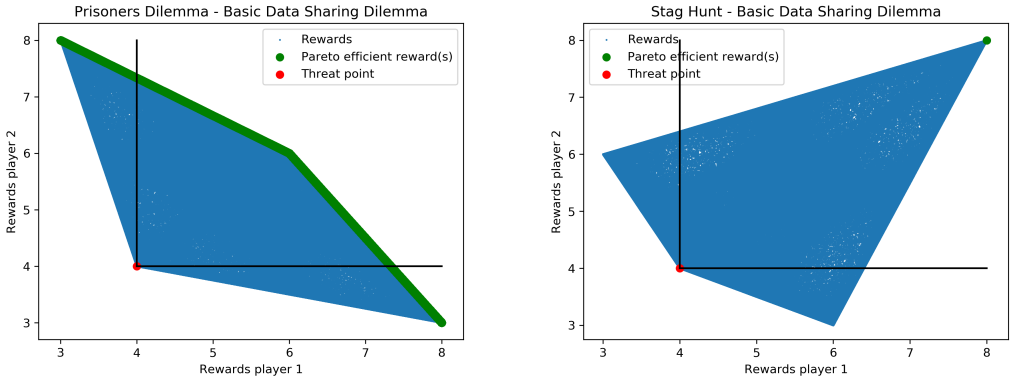
4.4 Analysis of the results

The Prisoner's Dilemma and Stag Hunt's basic games give us an excellent basis to compare with. In the Data Sharing Dilemma, we have two types of basic games (i.e. Prisoner's Dilemma and Stag Hunt), two types of benefit functions (i.e. Singular- and Mutual Benefits) and an adoption rate parameter represented by ϕ. Additionally, we could alter the Machine Learning function parameters, but for the sake of simplicity, we will keep $a = 0.1$ and $b = 1.1$. This section starts with a simple analysis of the basic games in terms of the analysis concepts defined in the previous section (e.g. threat-point and Pareto efficient rewards). Next, we analyse the different types of games with the Machine Learning function incorporated. Last but not least, we analyse the impact of the adoption (represented by ϕ) of an inter-organizational system.

4.4.1 The basic games

The two basic games offer us insights into the functions introduced in the Data Sharing Dilemma. However, first, we analyse the two games without any Frequency-Dependent influence. As stated and could be retrieved from Figures 4.4 and 4.5, the pure Nash equilibria for both games contain at least the strategy pair (Not Share, Not Share), while the Stag Hunt version also has a pure Nash equilibrium at (Share, Share).

More interestingly, when looking closely at Figure 4.10, the set of Pareto efficient rewards contains more rewards for the Prisoners Dilemma basic game when compared to the Stag Hunt basic game. This is because the Stag Hunt only has a Pareto



(a) Prisoners Dilemma Basic Game.

(b) Stag Hunt Basic Game.

Figure 4.10: Two basic games Data Sharing Dilemma.

efficient reward which is also the best obtainable pure Nash equilibrium. Therefore in Stag Hunt, there is a clear incentive to share; this is the opposite for the Prisoners Dilemma. We can see this by looking at the set of Pareto efficient rewards. The most intriguing individual Pareto efficient reward for a player in the Prisoners Dilemma requires one person to share all the data while the player receiving the highest rewards is not sharing, resulting in a reward pair which is not a Nash equilibrium. If players are patient and willing to cooperate strategically, we could obtain better rewards in the Prisoners Dilemma case. Still, they cannot be supported by a Nash equilibrium. All rewards north-east of the threat-point result (guarded by the black line) are equilibrium results by definition of the theorem of jointly-convergent pure-strategy pair rewards.

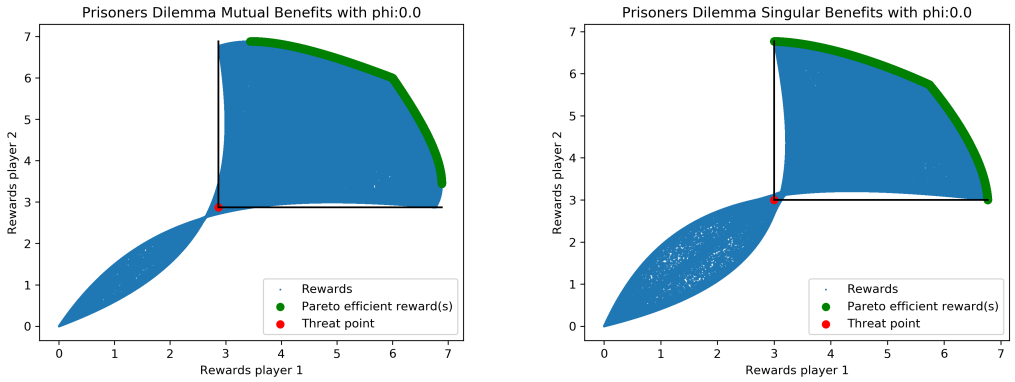
4.4.2 The effect of the Machine Learning function in the different types of games

In the following analysis phase, we introduce the inter-organizational system and assume that the system's adoption is still at the minimum level (i.e., $\phi = 0$. Players can now share their data with the Machine Learning algorithm, but they cannot reap the full potential because adoption is low. We will analyze the effect of adoption in the next session.

We display important results of the different types of games in Table 4.1 and visualize them in Figures 4.11 and 4.12. For all types of games, the minimal reward pairs are similar at $(0, 0)$ due to limited adoption of the inter-organizational system. However, players can retrieve higher rewards by sharing data. Sharing data is also an individual rational decision when looking at the threat-point result. In all types of the Data Sharing Dilemma, the threat-point result at $\phi = 0$ is that the player who is being threatened will share, even if the counter-party refuses to share.

Type of Game	Minimum Reward	Threat-point	Maximum reward
Prisoners Dilemma Singular Benefits	(0,0)	(3, 3)	(6.7692, 6.7692)
Prisoners Dilemma Mutual Benefits	(0,0)	(2.8695, 2.8695)	(6.8852, 6.8852)
Stag Hunt Singular Benefits	(0,0)	(3, 3)	(7.6521, 7.6521)
Stag Hunt Mutual Benefits	(0,0)	(2.8695, 2.8695)	(8, 8)

Table 4.1: Main results Data Sharing Dilemma $\phi = 0$.



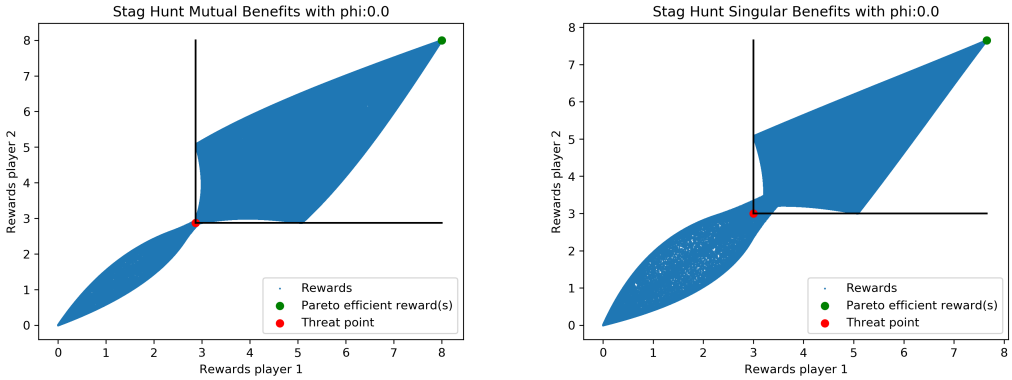
(a) Prisoners Dilemma Mutual Benefits.

(b) Prisoners Dilemma Singular Benefits.

Figure 4.11: Prisoners Dilemma types with $\phi = 0$.

This result significantly impacts the game dynamics, especially when looking at the Prisoners Dilemma version. In the Prisoners Dilemma version, sharing was never rational because of a pure Nash equilibrium when both players refused to share. However, being able to benefit from a Machine Learning function changes the game dynamics such that the equilibrium of the Prisoners Dilemma shifts to a more desirable state. Next to that, the threat-point result for the singular- and mutual types of games are numerically comparable. These results show us and confirm the logic behind the singular benefits that sharing your data is more beneficial and reaps higher rewards. Of course, the opposite is the case for the mutual benefits version, in which the threat-point rewards are slightly lower.

Furthermore, we see that the Stag Hunt type of game with mutual benefits has the highest possible rewards. In general, the mutual benefit type of game could receive higher rewards, but both players' sharing data is necessary to achieve these results. In the case of singular benefits for the Stag Hunt, some interesting behaviour is visible. In the basic Stag Hunt game, the Pareto efficient reward is also the pure



(a) Stag Hunt Mutual Benefits.

(b) Stag Hunt Singular Benefits.

Figure 4.12: Stag Hunt types with $\phi = 0$.

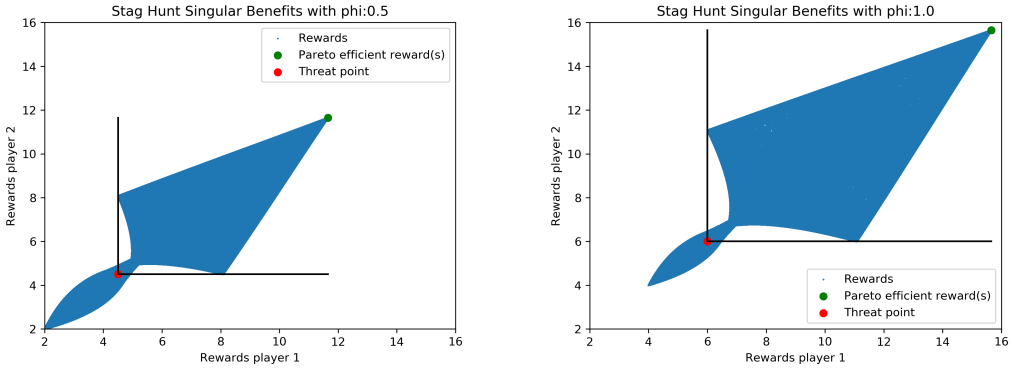
Nash equilibrium in which both players share. Contrasting this result, in singular benefits, the Pareto efficient reward is not when both players fully share their data. Still, when they carefully share their data when the counter-party is not sharing, such a Pareto efficient reward could be obtained by both players. In the opposite case of mutual benefits, the players can always retain the maximum reward by simply sharing.

Finally, we see other exciting dynamics in the set of Pareto efficient rewards for the Prisoners Dilemma. In the Prisoners Dilemma's basic game, we noticed that Pareto efficient rewards are obtainable and skewed highly into one individual's favour. However, when we introduce the inter-organizational system with Machine Learning, we end up in a more dense Pareto efficient rewards set. For each of the Pareto efficient rewards, they are worth strictly more than the threat-point result. Therefore, players' incentive in such a Data Sharing context increases substantially and could even be seen as rational.

4.4.3 The effect of the adoption of the inter-organizational system

The previous section gave us meaningful insight into the effect of the Machine Learning function, but only for $\phi = 0$. In this part, we analyse what happens when we gradually increase the adoption rate (i.e. $\phi = (0, 1]$). We observe identical behaviour between the different types of games, especially when looking at the minimum reward and threat-point reward. They all scale with ϕ in a linear fashion. For the threat-point result, this has no impact on the threat-point strategy, equal to when $\phi = 0$ the player who is being threatened shares his data while the counter-party is refusing to.

More interestingly, the set of rewards in which data sharing occurs grows in vol-



(a) Stag Hunt Singular Benefits $\phi = 0.5$. (b) Stag Hunt Singular Benefits $\phi = 1$.

Figure 4.13: Stag Hunt Singular Benefits variations.

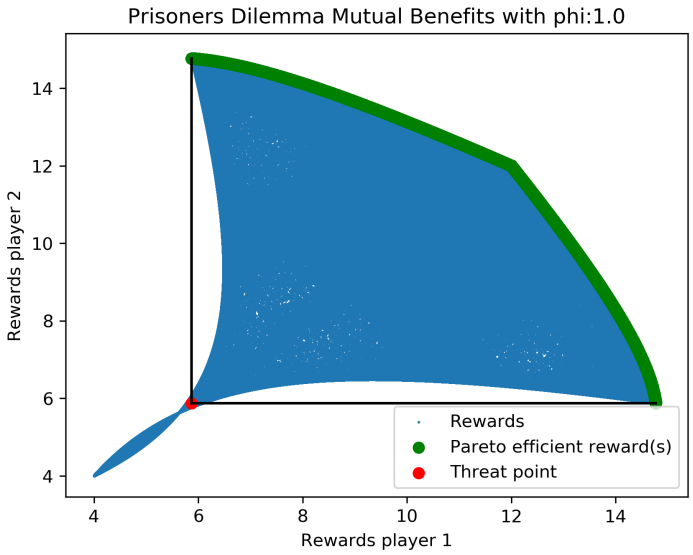


Figure 4.14: Prisoners Dilemma with Mutual Benefits $\phi = 1$.

ume compared to the set in which data sharing is only being done incidentally. This is displayed clearly in Figure 4.13. There we see that in the case when $\phi = 0.5$ that the set of rewards below (to the bottom-left of) the threat-point is larger than when $\phi = 1$. We could conclude that increasing the adoption of an inter-organizational system in such a context could lead to greater benefits, even if the counter-party refuses to share. However, we could debate whether the adoption of such a system in itself would trigger a higher frequency of sharing.

Finally, we focus on a particular case, when we have Prisoners Dilemma with Mutual Benefits and $\phi = 1$. Here we can see that the set of rewards to the bottom-left of the threat-point is tiny compared to the case in which $\phi = 0$. Partially, we can attribute this to the subject of mutual benefits, but this results from increasing the adoption rate, as stated earlier. What makes this case interesting is that the Pareto

efficient rewards set grows with ϕ. This offers players in the Data Sharing Dilemma a much greater strategic flexibility than the Stag Hunt version. On the other side, increasing the set of Pareto efficient rewards could result in disagreement in both players' strategic collaboration. In conclusion, we see the main result of adopting the inter-organizational system and therefore increasing ϕ results in greater rewards and more strategic variability.

4.5 Conclusion and discussion

In this chapter, we analysed the Data Sharing Dilemma. A problem that is faced by many supply chain players wanting to construct an inter-organizational system. As found in the literature, there are many factors influencing the adoption of an inter-organizational system (e.g. [24], [40], [69]), with trust being a common factor in that. In the literature on knowledge sharing, we find that players are often better off with not sharing knowledge (e.g. [77], [81]), or in our case, data. We combine two types of games from empirical literature to generalise a Machine Learning function into a Frequency-Dependent game. The results show that players have an individual rational incentive to share the data independent of the type of game or function, contradicting the literature on knowledge sharing. Also, we analyze multiple effects and results within the Data Sharing Dilemma, which give us insights into interesting potential strategic behaviour.

However, the results assume that an environment (i.e., an inter-organizational system) already has been constructed. One could argue that this simplifies a social complex topic. The construction of the environment is a large part of building the trust needed to create an inter-organizational system. On the other side, we argue that if a supply chain player does not see the personal benefits, he will never start constructing the environment in the first place. Furthermore, the Data Sharing Dilemma is a symmetric game, meaning that we assume that players have symmetric benefits; in reality, such a situation would be unlikely.

Further research could look into asymmetric versions of the Data Sharing Dilemma and potential effects on the willingness to share data. Last but not least, we have not discussed the sharing of potential benefits between supply chain partners. A cooperative game approach could give meaningful insights into the sharing of benefits in such a situation.

Chapter 5

A Maintenance-Oriented Service Control Tower Architecture

In the previous chapters, we investigated the different decisions and dilemmas that organizations face in developing a Service Control Tower and the Data Sharing dilemma in more detail. In this chapter, we look at the Service Control Tower itself. Developing a Service Control Tower environment is not a standardized process. However, in this chapter, we try to 'standardize' the components of a Service Control Tower based upon the MARCONI use-case. In other words, we want to provide organizations with a reference architecture for developing software aspects of the Service Control Tower and use the MARCONI use-case as a basis for this architecture.

Therefore, in this chapter, we start with an introduction to reference architectures. First, we look at definitions of reference architectures and some well-known examples. Next to that, we define the steps we take in order to construct the reference architecture. Next, we show the architecture, different software components and their functionality, and last but not least, an A3A0 architecture. Finally, this chapter discusses the Service Control Tower as a Service, the different software components, and aligning service performance indicators.

5.1 Introduction to reference architectures

The reference framework, reference model, and reference architecture are all terminology available in the literature describing models that can be used as a blueprint for software artefacts in a business context. For example, Becker (1995) describes that information models are essential for deploying and designing information systems. Reference models are also known by the name of universal models or generic models [39]. For example, a reference model is defined by Bass et al. (2003) as "a

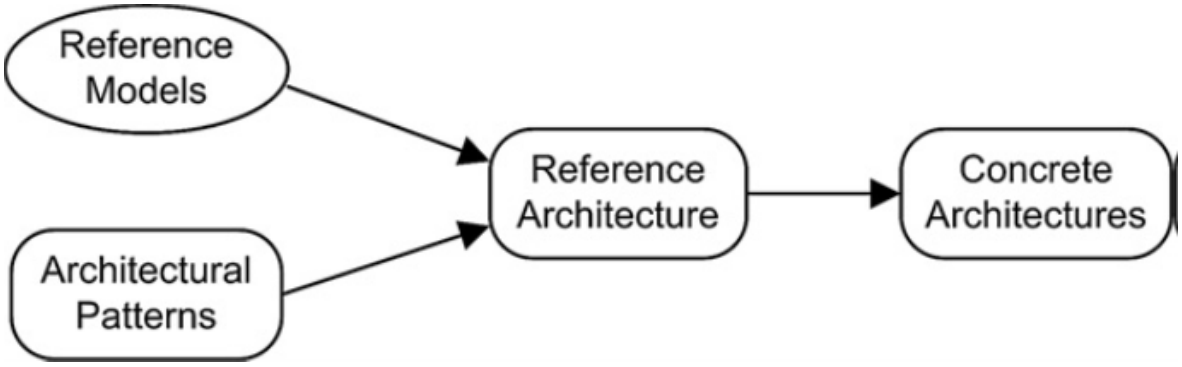


Figure 5.1: From Reference Model to Reference Architecture. [7]

division of functionality together with data flow between the pieces" while a reference architecture is defined by Bass et al. (2003) as "reference model mapped onto software elements (that cooperatively implement the functionality defined in the reference model) and the data flows between them". Specifically, reference architectures are often a collection of systems in an architecture that guides new versions of the system [65].

In Figure 5.1 we see that a reference model and architectural patterns feed into the reference architecture, which in its turn is the basis for a concrete architecture. Angelov et al. (2009) recognizes two goals for reference architectures, standardization and facilitation. Standardization is used to create interoperability between systems, while facilitation facilitates the design of a concrete architecture by acting as a guide. In total, five types of reference architectures have been determined by Angelov et al. (2009). We develop in this section a reference architecture for the Service Control Tower artefact that could be classified as a Type 5 reference architecture, i.e., the architecture is for an immature technology and is innovative in this nature, defining the components of the system.

In literature, we find a multitude of reference architecture, of which Service-Oriented Architectures is one of the nuances in reference architectures. A Service-Oriented Architecture is an architecture that links service offerings in a business context with traditional business and technology models [37]. In other words, a Service Oriented Architecture describes how a specific service offering can be realized by aligning business and IT. For example, the Service Control Tower reference architecture could be classified as a Service Oriented Architecture as it tries to offer service logistics optimization with the help of IT.

When looking at the architectures in literature, sample examples could be used as inspiration for constructing a reference architecture. We find a Service Level Agreement management reference architecture (e.g. [91]). A Service-Oriented Reference Architecture for smart cities (e.g. [27]), a service-oriented e-commerce ref-

erence architecture (e.g. [9]) and a reference architecture for microservices (e.g. [102]). We use these architectures as an inspiration for the reference architecture of the Service Control Tower.

How to construct a reference architecture is not set in stone. In literature, we find multiple papers that describe a methodology for development of a reference architecture (e.g., [86], [31]). Nevertheless, as stated by [65], an ideal structure for a reference architecture does not exist. One needs to balance the level of abstraction and the number of details. As such, we construct our methodology to create the reference architecture for the Service Control Tower. Partially we base this upon steps described in TOGAF [50] as (parts of) our architecture is made with the help of Enterprise Architecture modelling.

The steps we take in order to construct the Service Control Tower are the following.

1. Define the context of the Service Control Tower
2. Retrieve needs and input from the user based on questionnaire and workshop
3. Transform needs into requirements and checks on quality and relevance
4. Validate the context of the Service Control Tower
5. Map requirements to functionality and software components
6. Create functional software components architecture
7. Prioritize individual components
8. Create an A3 architecture overview based upon the earlier constructed architecture

Again, one should keep in mind that we classify this architecture as a Type 5 architecture based upon [6]. Therefore, the architecture has a facilitating function, describing components in a high level of abstraction, hopefully paving the way for future research into Service Control Towers.

5.2 The architecture

In this section, we give an introductory architecture for the Service Control Tower. As stated earlier, we define a Service Control Tower as ”An (inter-) organizational system which uses IT to specifically optimize the service logistics supply chain.”. Here, we set the context for such a system. Therefore, we start by stating the current business context, the partners’ strategic goals, and main activities. Then, within this context, we retrieve requirements for a Service Control Tower, check their

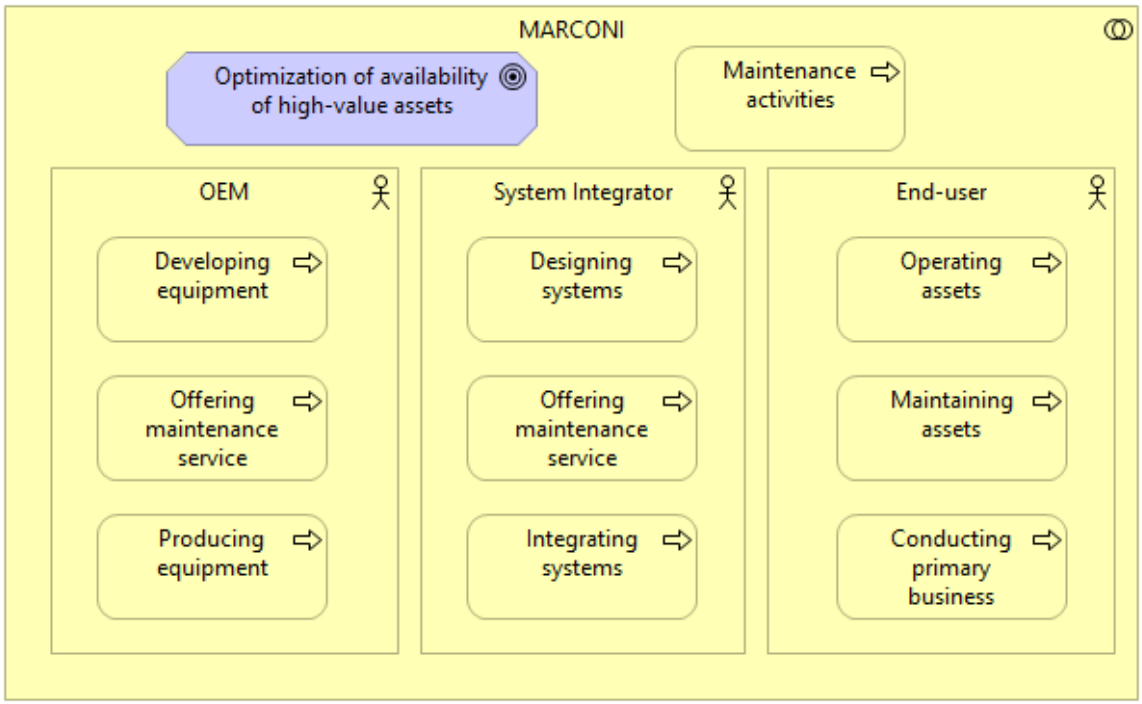


Figure 5.2: The MARCONI Business Context

quality, map them to functionality and, in the end, software components. After, we prioritize individual software components and discuss them. Finally, we end with an A3AO (A3 Architecture Overview*See A3AO.eu for an explanation) architecture that summarizes the Service Control Tower.

5.2.1 Business Context

In the current business context, the Service Control Tower is defined from the perspective of the MARCONI consortium, from which this research is a part. The MARCONI consortium consists of organizations active in the maritime industry, ranging from Original Equipment Manufacturers to End-Users. The consortium goal is to increase the availability of high-value maritime assets by optimizing maintenance supply-chain wide.

Therefore, we construct an architecture for a Service Control Tower in a maintenance-oriented environment. In other words, we expect that our architecture could be used for other organizations, as long as optimization of maintenance of assets is the goal. In Figure 5.2, we show a simplified version of the MARCONI consortium, in which we see an Original Equipment Manufacturer, a Systems Integrator (i.e., a supplier that uses hardware and software to combine subsystems into one system) and an

*http://a3ao.eu/

End-User. The End-User uses the asset for a business purpose. In real life, the roles of certain entities can change based upon changing market conditions. However, we keep context currently fixed for the sake of simplicity in Figure 5.2.

5.2.2 From needs to requirements to functionality to software entities

We set up a questionnaire (see Appendix B) and a workshop to retrieve the needs for Service Control Tower. We received 47 needs (see Appendix C), containing multiple requirements in them. Therefore, we transformed the needs into requirements with the help of [32]. In total, we have extracted 93 requirements, which we checked on guidelines from [32]. However, after checking the requirements, we removed thirteen, ending with a set of 80 requirements in total.

The next step is to link these requirements to functionalities in the Service Control Tower. In other words, which functions does the Service Control Tower need to encapsulate to meet the needs of a Service Control Tower? Again, we find inspiration from [15], where software elements are mapped towards a Transportation Control Tower. As stated, we find functionalities first, which we then map to software elements or entities as we call them. The complete selection of requirements, functionalities and entities can be found in Appendix D.

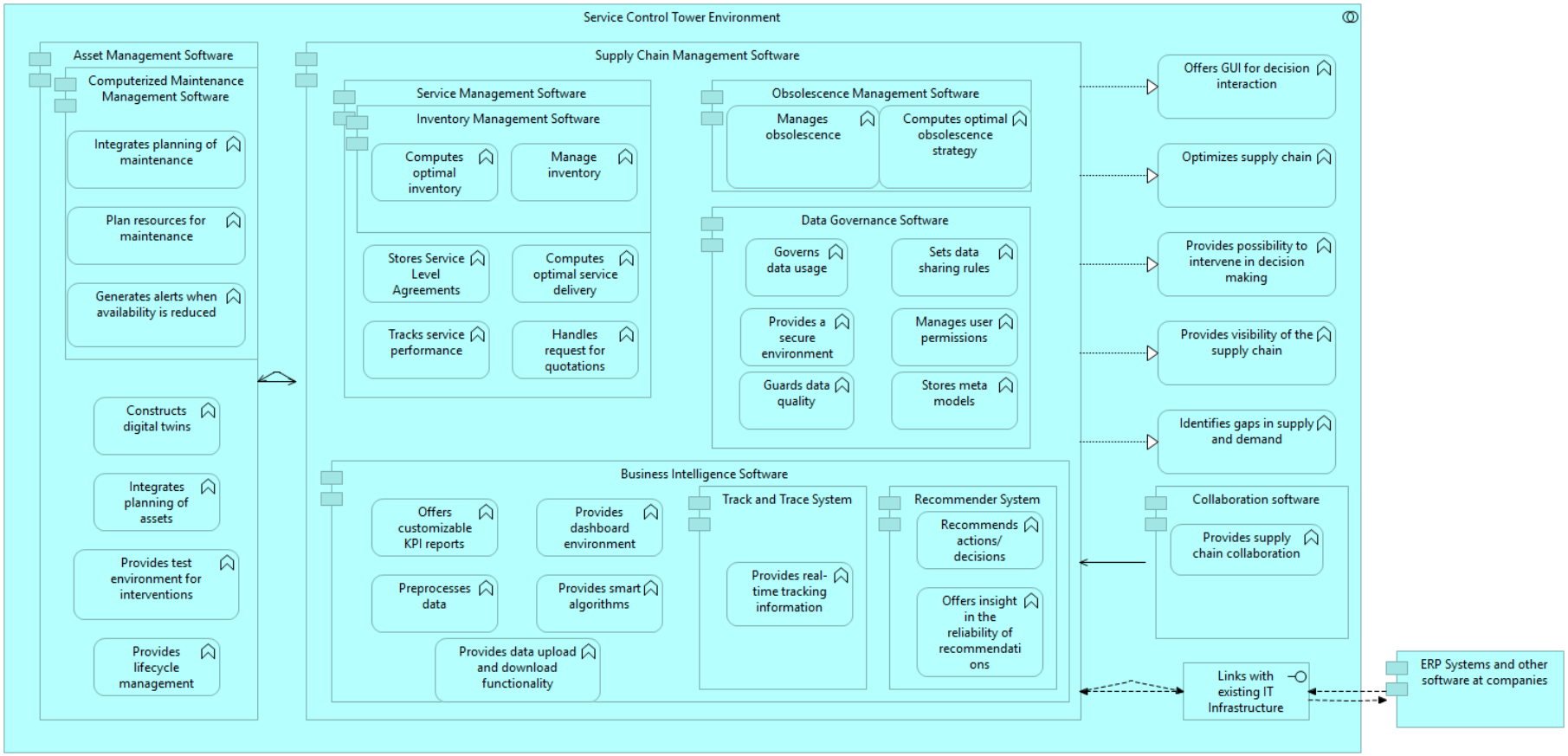


Figure 5.3: A Maintenance-Oriented Service Control Tower Application Architecture

The result of these mappings is the first architecture on an application level. We show the architecture in Figure 5.3. Here, we see the Service Control Tower, based upon the requirements that have been extracted. The main functionalities of a Service Control Tower are to optimize the service supply chain, provide visibility in the supply chain, intervene in decision-making, and identify gaps in supply and demand. All of these link to the main component of the Service Control Tower, which we define as Supply Chain Management Software. The Supply Chain Management Software consists of multiple other applications, i.e., Service Management Software, Business Intelligence Software, Obsolescence Management Software and Data Governance Software. The Supply Chain Management Software has a link and interfaces with the Asset Management Software. All software elements have their own set of functionalities and dependencies. We will discuss all of these elements one by one.

Because first, we identify which elements are most important in the development in comparison to others. Therefore, we rank the different software elements with the help of the MoSCoW method. MoSCoW stands for Must Have, Should Have, Could Have or Won't Have and prioritises the software elements. With Must-Have being the most important and Won't Have the least important. We ranked these software elements with the help of expert opinion in a workshop with eight consortium participants. Two software elements have been classified as Must Have, four as Should Have and two as Could Have. The results are available in Appendix E.

Must Have

As said, two software elements in the Service Control Tower are a Must Have. These are Business Intelligence software and Data Governance software. Both can be seen as prerequisites for the environment, as they provide software that helps operate the other elements of the Service Control Tower.

Starting with Business Intelligence, we define Business Intelligence software as "techniques used in spotting, digging-out, and analyzing business data, such as sales revenue by-products and/or departments, or by associated costs and incomes"[36]. In the Service Control Tower, Business Intelligence software allows users to transform data, build dashboards that show information, and create visibility of parts of the supply chain. Next, Business Intelligence software stores intelligent algorithms that can perform mathematical optimizations that provide decision-makers with information to improve the supply chain's decision-making. One could link Business Software to the visibility aspect of the Tower level in the previous chapter.

The second software element, Data Governance, is defined by "the framework for decision rights and accountabilities to encourage desirable behavior in the use of data"[97]. However, multiple experts have denounced that Data Governance is es-

sential, especially in a supply chain-wide collaboration. Therefore, we define Data Governance software as software that provides the set of rules regarding data governance and provides an environment that manages the security and quality of the data itself. Therefore, Data Governance can be linked towards the Foundation level of the Service Control Tower in the previous chapter as a necessity to share and link data.

Should Have

Next up are the four Should Haves; the experts have classified these as necessary but not essential for a Service Control Tower environment. Finally, starting with Collaboration Software, we define Collaboration Software as software containing tools and mechanisms that facilitate the collaboration process in a supply chain. However, we do not see the Collaboration Software as a part of the Supply Chain Management Software, as it has a more indirect effect. An example of Collaboration Software is the possibility of co-creating in an environment and communicating via video/audio and share files.

The second software element that has been classified is Asset Management Software; this software element is defined with the help of ”systematic process of maintaining, operating and upgrading physical assets cost-effectively” [61]. The software itself can construct Digital Twins, giving lifecycle management and integrating the planning of assets operation.

The following software element is Obsolescence Management Software, which helps manage obsolescence (e.g., functional or logistical obsolescence). We also define Obsolescence Management Software as a Should Have.

The following software element, Maintenance Management Software, is an element of Asset Management Software, but it is essential to discuss it independently. Maintenance Management Software regards all processes that are related to the maintenance of the assets. In our case, we see Maintenance Management as a Should Have, as it is necessary to plan resources and maintenance for the upkeep of these assets.

Last but not least is the Track and Trace System. We define Track and Trace software as software that keeps (real-time) tracking information and provides visibility about certain aspects of the supply chain (e.g., transportation, delivery, events in a business process, etcetera.). Track and Trace functionality is also a Should Have, and should be installed in the Service Control Tower in the Tower level.

To summarize, all four software elements described as a Should Have are seen as necessary. However, the focus should be on the Must-Haves first to facilitate the development of the Should Haves described here.

Could Have

The last section of software elements we describe is the Could Haves. Based upon expert opinion, two software elements have been defined as a Could Have. This means that these elements are ranked the lowest in terms of priority in the Service Control Tower. However, one should keep in mind that they are still considered valuable, as they are not ranked as Won't Haves. We discuss the individual components in this section.

The first of the Could Haves is Service Management Software. Service Management Software is defined as software that supports, executes and stores important service offerings performed in the supply chain by specific business processes. Service Management Software compares the current service performance to the service offerings in Service Level Agreements in terms of functionality. Next to that, the Service Management Software could be linked with Inventory Management Software. We advise developing the Service Management component in the last level of Service Control Tower development, so in case Service Level Agreements are core to the workings of the environment.

The other software element that has been determined as a Could Have is the Recommender System Software. We define the Recommender System as containing "tools for interacting with large and complex information spaces." [22]. In a Service Control Tower context, the Recommender System operates within the Business Intelligence software and looks especially for favourable actions/recommendations in the context it operates in. Therefore, the Recommender System can only be built if Business Intelligence Software is already in place.

Concluding, we have discussed the different software elements, their ranking and have shown their architecture individually.

5.2.3 A3 architecture overview

We end with the A3 architecture overview, which we can see as a summary of the artefact of interest, the Service Control Tower. A3 architecture overviews have been proposed by [20] as an efficient method to capture the knowledge of a complex system and its architecture. In the Service Control Tower case, we have described the business context and needs, their transformation into software components, and now we summarize all of them into the A3 architecture overview.

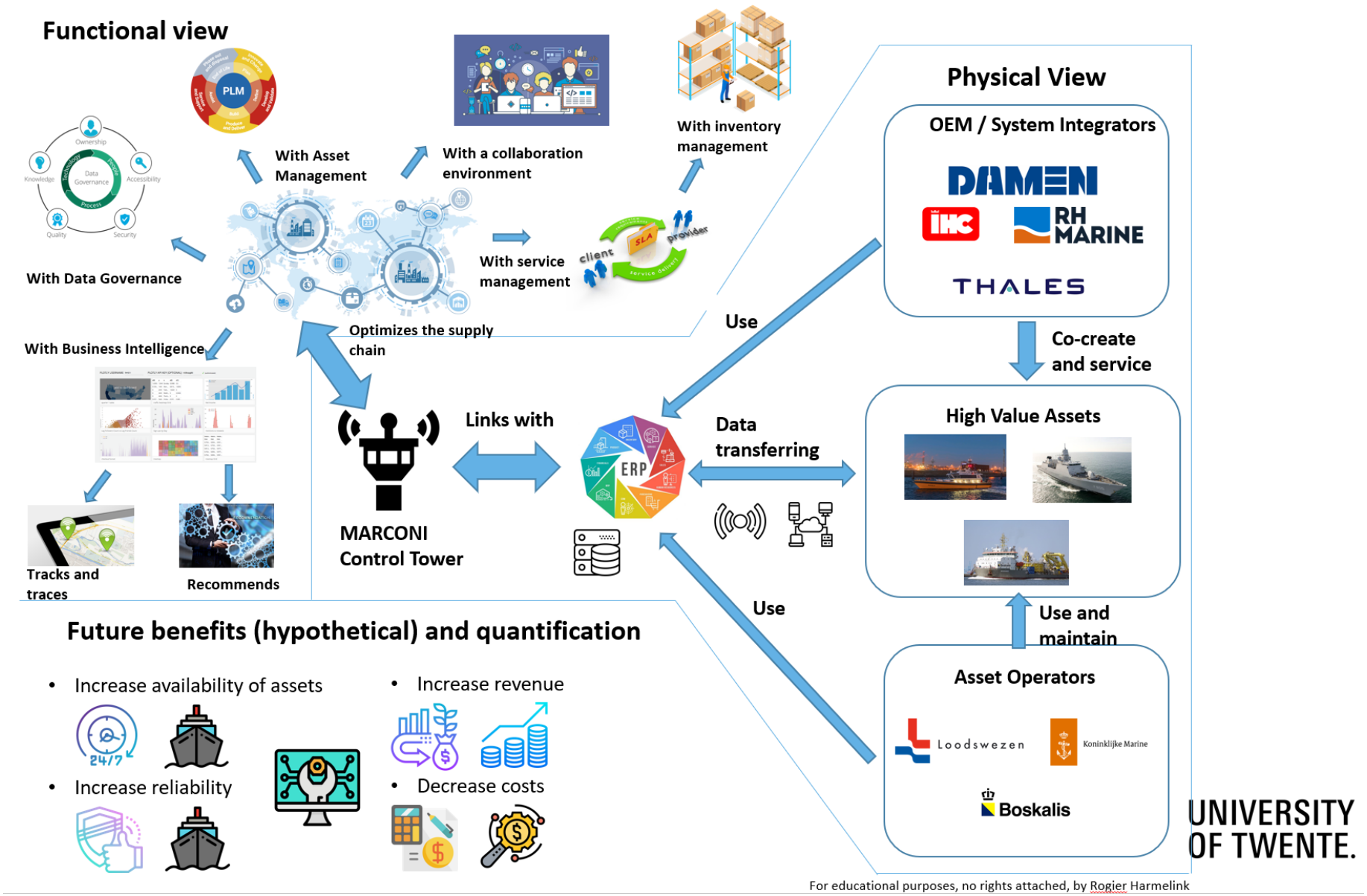


Figure 5.4: Service Control Tower A3 Architecture Overview

The A3 Architecture Overview is visible in Figure 5.4. We divide the A3 Architecture overview into three different spaces, a physical view (i.e., the tangible components of a system), a functional view (i.e., the functionality the components have) and future benefits and quantification of the system. We describe the architecture in the same order. At the Service Control Tower base is the MARCONI business context, which means that we have high-value assets (i.e., ships, vessels), which are co-created and serviced by Original Equipment Manufacturers and System Integrators. Their end-users operate these assets.

Data is captured from the processes at the individual organizations but also the assets themselves. This data is transferred to Enterprise Resource Planning systems, which are also used to maintain and operate the assets. These ERP systems should be linked with the artefact of interest, the Service Control Tower.

The Service Control Tower has the main functionality to optimize the service supply chain, with the primary goal to optimize the availability of the high-value assets in MARCONI. The Service Control Tower uses multiple software components, e.g. Business Intelligence Software, Data Governance Software and Asset Management Software. Expected benefits are an increase in availability and reliability of the assets, an increase in revenue for organizations involved and a decrease in costs.

Which concludes our construction of a first reference framework for a maintenance-oriented Service Control Tower. In the next section, we discuss the possibility to have the Service Control Tower as a Service.

5.3 The Service Control Tower as a Service

The way Information Technology is offered to customers has rapidly changed over the last few decades. Software as a Service has become a dominant concept in IT service providing. In addition, service-Oriented Architecture and Microservices have become common good in the IT services industry. In this section, we model the Service Control Tower as a Service, i.e., we look at how software components individually can be offered.'

We have discussed these possibilities within the consortium during a workshop (see Appendix F). One of the earlier questions was, who is responsible for which service in the supply chain, or could be the best supplier? The answers to these questions were mixed, so we could not generalize an answer to that. We advise organizations collaborating to discuss the same question based upon their unique business context.

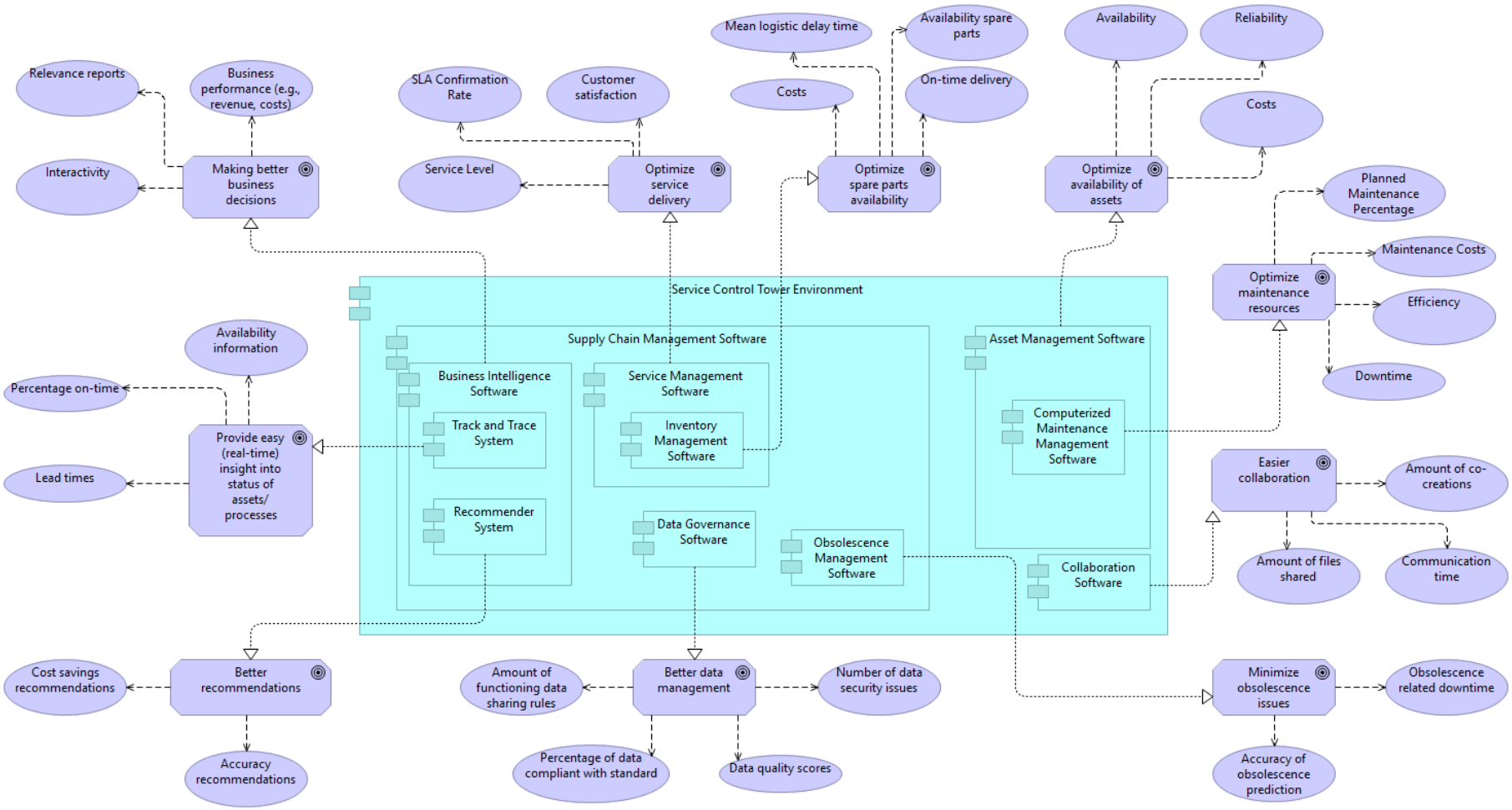


Figure 5.5: The Service Control Tower as a Service

However, we have been able to identify the goals of each software element and the different measurement methods that can help attain the goal and measure performance. Figure 5.5 shows these goals and measurements per software elements. For example, Data Governance Software should help better data management; one can measure this goal with the number of data security issues, data quality scores, the percentage of data compliant with the accompanying standard, and the amount of functioning data rules. Similarly, a Recommender System should generate better recommendations, which can be measured with the accuracy of the recommendations and the cost savings related to the recommendations.

Organizations wanting to build a Service Control Tower can use this overview as a basis for discussion; who is responsible for which service and how performance is measured should be discussed in individual cases.

5.4 Conclusion and discussion

In this chapter, we have discussed and designed the artefact of interest of this thesis, the Service Control Tower. First, we discussed available reference architectures from literature; based upon that, we defined the context of the architecture. Then, retrieved the needs from potential users of the system and transformed them into requirements and a preliminary system architecture containing software components with individual software priority. Next, we have discussed offering software elements as a service, i.e., the Service Control Tower as a Service.

As stated in this chapter, we have constructed an early reference architecture (i.e., Type 5 according to [6]). The downsides to this approach are that our architecture is limited to software components, requirements, and functionality. Exact business rules and interfacing between the different components and details of the individual components still need to be designed. Next, further research should be done into which data standard should support a Service Control Tower. However, this architecture should provide organizations with a first blueprint to guide them in developing a Service Control Tower.

Chapter 6

Conclusions and discussion

6.1 Conclusions

In the previous chapters, we have explored the world of the Service Control Tower. We introduced the context of this research at first, the MARCONI project and the maritime sector. Next, we summarized existing literature and architectures found. We see that the research surrounding (Service) Control Towers is in an early stage. Limited information is available, but interest from an academic and practical perspective is there. Finally, we define the Service Control Tower and its context and use that as a basis for the rest of the report. In the following chapters, we have three different angles.

First, we examined different dilemmas that organizations face in a supply chain collaboration. Additionally, we define the four levels of the Service Control Tower that organizations need to build to end up with a fully functioning Service Control Tower. We identified a set of dilemmas that we link towards the general process of supply chain collaborations and the four levels of the Service Control Tower. For individual dilemmas, we also provide architectures for the different solutions. The results here are exploratory but should assist organizations in the process of collaboration and successfully communicating.

Additionally to the different dilemmas in supply chain collaboration, we explore the dilemma regarding Data Sharing, which leads to the Data Sharing Dilemma, a Game Theoretical model. We show that trust is essential in adopting inter-organizational systems and finding evidence in the literature that players usually retain to share. However, in our Game Theoretical model, we show that if learning occurs on the data, players have an individual rational incentive to share data independent of the type of game or function, which contradicts current literature on knowledge sharing.

Last but not least, we construct a reference architecture in the fifth chapter. We first introduce the concept of reference architecture and classify our reference architecture. Then, we design our architecture based upon system needs (from the

MARCONI context), requirements and map them towards different software entities. The result is a preliminary system architecture, which should help organizations by acting as a first blueprint in developing the Service Control Tower.

6.2 Discussion

However, the discussed work in this thesis is not conclusive. As said, research into (Service) Control Towers is still in an immature phase. Although the subject has gained increasing attention in the last decade, academic research into the matter is still limited. This thesis contributes three-fold (i.e., by defining the Service Control Tower, analyzing dilemmas in Service Control Towers and developing a reference framework for them). However, more research should be done into certain aspects of the chapters in this thesis.

For the dilemmas we have identified in the process of supply chain collaboration, we have only done merely exploratory research. More empirical evidence should be gathered and analyzed to support organizations in a supply chain in their decision-making and collaboration. In the case of the Data Sharing Dilemma, more research needs to be done into the input and output of the model (i.e., the values taken there are abstract and arbitrary). In other words, which value does data have? Furthermore, next, how do we share this data in a supply chain collaboration? These aspects have not been investigated in the current Data Sharing Dilemma. Last but not least, our reference architecture is on an abstract level and therefore misses essential details (e.g., interfacing, data standards, etcetera.) that organizations require in the process of constructing a Service Control Tower.

To end the discussion, more (academic) research is needed into the subject of (Service) Control Towers. One should keep in mind that commercial parties have the real incentive to develop these environments, while academia should continue to extract the actual academic value of these environments.

Bibliography

[1] Henk A Akkermans, Paul Bogerd, Enver Yücesan, and Luk N Van Wassenhove. The impact of ERP on supply chain management: Exploratory findings from a European Delphi study. *European Journal of operational research*, 146 (2):284–301, 2003. ISSN 0377-2217.

[2] Sultan Aldossary and William Allen. Data security, privacy, availability and integrity in cloud computing: issues and current solutions. *International Journal of Advanced Computer Science and Applications*, 7(4):485–498, 2016.

[3] Cyril Alias, Mandar Jawale, Alexander Goudz, and Bernd Noche. Applying novel Future-Internet-based supply chain control towers to the transport and logistics domain. In *ASME 2014 12th Biennial Conference on Engineering Systems Design and Analysis*. American Society of Mechanical Engineers Digital Collection, 2014.

[4] Cyril Alias, Çagdas Özgür, Mandar Jawale, and Bernd Noche. Analyzing the potential of Future-Internet-based logistics control tower solutions in warehouses. In *Proceedings of 2014 IEEE International Conference on Service Operations and Logistics, and Informatics*, pages 452–457. IEEE, 2014. ISBN 1479960586.

[5] Cyril Alias, Alexander Goudz, Mandar Jawale, and Bernd Noche. Generating a business model canvas for future-internet-based logistics control towers. In *2015 4th International Conference on Advanced Logistics and Transport (ICALT)*, pages 257–262. IEEE, 2015. ISBN 1479984000.

[6] Samuil Angelov, Paul Grefen, and Danny Greefhorst. A classification of software reference architectures: Analyzing their success and effectiveness. In *2009 Joint Working IEEE/IFIP Conference on Software Architecture & European Conference on Software Architecture*, pages 141–150. IEEE, 2009. ISBN 1424449847.

[7] Samuil Angelov, Paul Grefen, and Danny Greefhorst. A framework for analy-

sis and design of software reference architectures. *Information and Software Technology*, 54(4):417–431, 2012. ISSN 0950-5849.

[8] Jbid Arsenyan, Gülçin Büyüközkan, and Orhan Feyzioğlu. Modeling collaboration formation with a game theory approach. *Expert Systems with Applications*, 42(4):2073–2085, 2015. ISSN 0957-4174.

[9] Fabian Aulkemeier, Milan Schramm, Maria-Eugenia Iacob, and Jos Van Hillegersberg. A service-oriented e-commerce reference architecture. *Journal of theoretical and applied electronic commerce research*, 11(1):26–45, 2016.

[10] Tim S Baines, Howard W Lightfoot, Ornella Benedettini, and John M Kay. The servitization of manufacturing: A review of literature and reflection on future challenges. *Journal of manufacturing technology management*, 2009. ISSN 1741-038X.

[11] Daniel Balliet. Communication and cooperation in social dilemmas: A meta-analytic review. *Journal of Conflict Resolution*, 54(1):39–57, 2010. ISSN 0022-0027.

[12] Stephanie Barrett and Benn Konsynski. Inter-organization information sharing systems. *MIS Quarterly*, 6:93–105, 1982. ISSN 0276-7783. doi: $10.2307/248993$.

[13] Len Bass, Paul Clements, and Rick Kazman. *Software architecture in practice*. Addison-Wesley Professional, 2003. ISBN 0321680413.

[14] Carlo Batini, Anisa Rula, Monica Scannapieco, and Gianluigi Viscusi. From data quality to big data quality. *Journal of Database Management (JDM)*, 26 (1):60–82, 2015.

[15] Anne Baumgrass, Remco Dijkman, Paul Grefen, Shaya Pourmirza, Hagen Völzer, and Mathias Weske. A software architecture for a transportation control tower. *Technische Universiteit Eindhoven, Tech. Rep. Beta Working Paper series*, 461, 2014.

[16] Jörg Becker. Strukturanalogien in Informationsmodellen. In *Wirtschaftsinformatik'95*, pages 133–150. Springer, 1995.

[17] Prashant; Griffin-Cryan Belinda; van Doesburg Rob; Sparks MarieAnne; Paton Adrian Bhosle, Gaurav; Kumar. Global Supply Chain Control Towers. Technical report, Capgemini Consulting, 2011.

[18] Patrick Billingsley. *Probability and measure*. John Wiley & Sons, 2008. ISBN 8126517719.

[19] Jose Bleda, Roddy Martin, Tushar Narsana, and Derek Jones. Prepare for Takeoff with a Supply Chain Control Tower Enabling technologies for supply chain control towers. 2014.

[20] P Daniel Borches and G Maarten Bonnema. 3.3. 1 A3 Architecture Overviews: Focusing architectural knowledge to support evolution of complex systems. In *INCOSE International Symposium*, volume 20, pages 354–369. Wiley Online Library, 2010. ISBN 2334-5837.

[21] Thomas Brenner and Ulrich Witt. Melioration learning in games with constant and frequency-dependent pay-offs. *Journal of Economic Behavior & Organization*, 50(4):429–448, 2003. ISSN 0167-2681. doi: https://doi.org/10.1016/S0167-2681(02)00034-3.

[22] Robin Burke, Alexander Felfernig, and Mehmet H Göker. Recommender systems: An overview. *Ai Magazine*, 32(3):13–18, 2011. ISSN 2371-9621.

[23] Abhishek Chandra, Jon Weissman, and Benjamin Heintz. Decentralized edge clouds. *IEEE Internet Computing*, 17(5):70–73, 2013. ISSN 1089-7801.

[24] Ying-Hueih Chen, Tzu-Pei Lin, and David C Yen. How to facilitate inter-organizational knowledge sharing: The impact of trust. *Information & Management*, 51(5):568–578, 2014. ISSN 0378-7206. doi: 10.1016/j.im.2014.03.007.

[25] InduShobha N Chengalur-Smith, Donald P Ballou, and Harold L Pazer. The impact of data quality information on decision making: an exploratory analysis. *IEEE Transactions on Knowledge and Data Engineering*, 11(6):853–864, 1999. ISSN 1041-4347.

[26] Alton Chua. Knowledge sharing: a game people play. In *Aslib Proceedings*, pages 117–129. MCB UP Ltd, 2003.

[27] Stephen J Clement, David Wesley McKee, and Jie Xu. Service-oriented reference architecture for smart cities. In *2017 IEEE symposium on service-oriented system engineering (SOSE)*, pages 81–85. IEEE, 2017. ISBN 150906320X.

[28] Morris A Cohen, Yu-Sheng Zheng, and Vipul Agrawal. Service parts logistics: a benchmark analysis. *IIE transactions*, 29(8):627–639, 1997. ISSN 0740-817X.

[29] Simon Dalmolen, Hans Moonen, and Jos van Hillegersberg. Towards an information architecture to enable cross chain control centers. *Cross-Chain*

Collaboration in the Fast Moving Consumer Goods Supply Chain, University of Eindhoven, pages 111–122, 2015.

[30] Frank W Davis and Karl B Manrodt. Service logistics: an introduction. *International Journal of Physical Distribution & Logistics Management*, 1991. ISSN 0960-0035.

[31] Jean-Marc DeBaud, Oliver Flege, and Peter Knauber. PuLSE-DSSA—a method for the development of software reference architectures. In *Proceedings of the third international workshop on Software architecture*, pages 25–28, 1998.

[32] Jeremy Dick, Michael Ryan, Lou Wheatcraft, Rick Zinni, and Katherine Baksa. INCOSE Guide for writing requirements. Technical Report April, 2017.

[33] Dinalog. Supply Chain Collaboration Tool - The Road to Horizontal Collaboration. URL `www.dinalog.nl/samenwerking`.

[34] Xinhua Dong, Ruixuan Li, Heng He, Wanwan Zhou, Zhengyuan Xue, and Hao Wu. Secure sensitive data sharing on a big data platform. *Tsinghua science and technology*, 20(1):72–80, 2015. ISSN 1007-0214.

[35] Peter F Drucker. The technological revolution: notes on the relationship of technology, science, and culture. *Technology and Culture*, 2(4):342–351, 1961. ISSN 0040-165X.

[36] Cebotarean Elena. Business intelligence. *Journal of Knowledge Management, Economics and Information Technology*, 1(2):1–12, 2011. ISSN 2069-5934.

[37] Abdelkarim Erradi, Sriram Anand, and Naveen Kulkarni. SOAF: An architectural framework for service definition and realization. In *2006 IEEE International Conference on Services Computing (SCC'06)*, pages 151–158. IEEE, 2006. ISBN 0769526705.

[38] Ayse Sena Eruguz, Tarkan Tan, and Geert-Jan van Houtum. A survey of maintenance and service logistics management: Classification and research agenda from a maritime sector perspective. *Computers & Operations Research*, 85:184–205, 2017. ISSN 0305-0548.

[39] Peter Fettke and Peter Loos. Classification of reference models: a methodology and its application. *Information systems and e-business management*, 1 (1):35–53, 2003. ISSN 1617-9846.

[40] Michael J Gallivan and Gordon Depledge. Trust, control and the role of interorganizational systems in electronic partnerships. *Information Systems Journal*, 13(2):159–190, 2003. ISSN 1350-1917. doi: 10.1046/j.1365-2575.2003.00146.x.

[41] Paul Groth. Transparency and reliability in the data supply chain. *IEEE Internet Computing*, 17(2):69–71, 2013. ISSN 1089-7801.

[42] Rogier Harmelink and Reinoud Joosten. Computing threat points in irreducible ETP-ESP games. 2020.

[43] S P Ho, Y Hsu, and E Lin. Model for knowledge-sharing strategies: A game theory analysis. *The Engineering Project Organization Journal*, 1(1):53–65, 2011. ISSN 2157-3727.

[44] Wout Hofman. Control tower architecture for multi-and synchromodal logistics with real time data. In *5th International Conference on Information Systems, Logistics and Supply Chain (ILS 2014)*, 2014.

[45] Christian Huemer, Philipp Liegl, Rainer Schuster, Hannes Werthner, and Marco Zapletal. Inter-organizational systems: From business values over business processes to deployment. In *2008 2nd IEEE International Conference on Digital Ecosystems and Technologies*, pages 294–299. IEEE, 2008. ISBN 142441489X.

[46] P K Humphreys, M K Lai, and D Sculli. An inter-organizational information system for supply chain management. *International Journal of Production Economics*, 70(3):245–255, 2001. ISSN 0925-5273.

[47] F Robert Jacobs. Enterprise resource planning (ERP)—A brief history. *Journal of operations management*, 25(2):357–363, 2007. ISSN 0272-6963.

[48] Reinoud Joosten and Llea Samuel. On Finding Large Sets of Rewards in Two-Player ETP–ESP Games. *International Game Theory Review (IGTR)*, 22 (02):1–27, 2020.

[49] Reinoud Joosten, Thomas Brenner, and Ulrich Witt. Games with frequency-dependent stage payoffs. *International Journal of Game Theory*, 31(4):609–620, 2003. ISSN 0020-7276. doi: https://doi.org/10.1007/s001820300.

[50] Andrew Josey. *TOGAF® Version 9.1-A Pocket Guide*. Van Haren, 2016. ISBN 9087539673.

[51] Marin Jovanovic, Mats Engwall, and Anna Jerbrant. Matching Service Offerings and Product Operations: A Key to Servitization Success: Existing conditions, such as product characteristics or market attributes, may determine the success of a move toward servitization. *Research-Technology Management*, 59(3):29–36, 2016. ISSN 0895-6308.

[52] Enis Karaarslan and Enis Konacaklı. Data Storage in the Decentralized World: Blockchain and Derivatives. *arXiv preprint arXiv:2012.10253*, 2020.

[53] Felix Kaufman. Data systems that cross company boundaries. *Harvard Business Review*, 44(1):141–155, 1966.

[54] Young-Jin Kim, Marina Thottan, Vladimir Kolesnikov, and Wonsuck Lee. A secure decentralized data-centric information infrastructure for smart grid. *IEEE Communications Magazine*, 48(11):58–65, 2010. ISSN 0163-6804.

[55] Gary King. An introduction to the dataverse network as an infrastructure for data sharing, 2007.

[56] A P Kryukov and A P Demichev. Decentralized data storages: Technologies of construction. *Programming and Computer Software*, 44(5):303–315, 2018. ISSN 1608-3261.

[57] Jin Li, Yinghui Zhang, Xiaofeng Chen, and Yang Xiang. Secure attribute-based data sharing for resource-limited users in cloud computing. *Computers & Security*, 72:1–12, 2018. ISSN 0167-4048.

[58] Matthew Liotine. Shaping the Next Generation Pharmaceutical Supply Chain Control Tower with Autonomous Intelligence. *Journal of Autonomous Intelligence*, 2(1):56–71, 2019. ISSN 2630-5046.

[59] Xuefeng Liu, Yuqing Zhang, Boyang Wang, and Jingbo Yan. Mona: Secure multi-owner data sharing for dynamic groups in the cloud. *IEEE transactions on parallel and distributed systems*, 24(6):1182–1191, 2012. ISSN 1045-9219.

[60] Sebastián Lozano. Information sharing in DEA: A cooperative game theory approach. *European Journal of Operational Research*, 222(3):558–565, 2012. ISSN 0377-2217.

[61] Regina S McElroy. Update on national asset management initiatives: facilitating investment decision-making. In *Innovations in Urban Infrastructure Seminar of the APWA International Public Works Congress*, pages 1–10. Citeseer, 1999.

[62] Alan Meekings and Steve Briault. The “control tower” approach to optimising complex service delivery performance. *Measuring Business Excellence*, 2013. ISSN 1368-3047.

[63] Miloš Milenković. A value case approach for improving the quality of rail freight services: control tower concept. In *8th International Symposium and 30th National Conference on Operational Research*, page 140.

[64] Nurul Azni Mohammad and Muhammad Faiz Mohd Azani. Malaysia's Experience in Integrated Coordination of Offshore Supply Vessels through Integrated Logistics Control Tower ILCT. In *SPE Asia Pacific Oil and Gas Conference and Exhibition*. Society of Petroleum Engineers, 2018. ISBN 1613995954.

[65] Gerrit Muller. A reference architecture primer. *Eindhoven Univ. of Techn., Eindhoven, White paper*, 2008.

[66] Nanjangud C Narendra, Koundinya Koorapati, and Vijayalakshmi Ujja. Towards cloud-based decentralized storage for internet of things data. In *2015 IEEE International Conference on Cloud Computing in Emerging Markets (CCEM)*, pages 160–168. IEEE, 2015. ISBN 1467385662.

[67] John Nash. Non-cooperative games. *Annals of mathematics*, 54(2):286–295, 1951. ISSN 0003-486X. doi: $10.2307/1969529$.

[68] Gary Pan. The hidden dilemmas in software development project decision-making: persist or desist? *PACIS 2006 Proceedings*, page 92, 2006.

[69] Niki Panteli and Siva Sockalingam. Trust and conflict within virtual inter-organizational alliances: a framework for facilitating knowledge sharing. *Decision Support Systems*, 39(4):599–617, 2005. ISSN 0167-9236. doi: $10.1016/\mathrm{j.dss.2004.03.003}$.

[70] Claudia Perlich. Learning Curves in Machine Learning. Technical report, IBM Research Division, New York, 2010.

[71] G Premkumar and Keshavamurthy Ramamurthy. The role of interorganizational and organizational factors on the decision mode for adoption of interorganizational systems. *Decision Sciences*, 26(3):303–336, 1995. ISSN 0011-7315.

[72] Michael R Reich. Pandemic Governance in Japan and the United States: The Control-Tower Metaphor. *Health Systems & Reform*, 6(1):e1829314, 2020. ISSN 2328-8604.

[73] Sushmita Ruj, Milos Stojmenovic, and Amiya Nayak. Decentralized access control with anonymous authentication of data stored in clouds. *IEEE transactions on parallel and distributed systems*, 25(2):384–394, 2013. ISSN 1045-9219.

[74] Jan Willem Rustenburg. Planning Services: A Control Tower Solution for Managing Spare Parts. In *Logistics and Supply Chain Innovation*, pages 239–259. Springer, 2016.

[75] Farrukh Sahar. Tradeoffs between usability and security. *IACSIT International Journal of Engineering and Technology*, 5(4), 2013.

[76] Ola Salman, Imad Elhajj, Ayman Kayssi, and Ali Chehab. An architecture for the Internet of Things with decentralized data and centralized control. In *2015 IEEE/ACS 12th International Conference of Computer Systems and Applications (AICCSA)*, pages 1–8. IEEE, 2015. ISBN 1509004785.

[77] Hanan M Samieh and Khaled Wahba. Knowledge sharing behavior from game theory and socio-psychology perspectives. In *2007 40th Annual Hawaii International Conference on System Sciences (HICSS'07)*, pages 187c–187c. IEEE, 2007. ISBN 0769527558.

[78] Willem E Saris. A technological revolution in data collection. *Quality and Quantity*, 23(3-4):333–349, 1989. ISSN 0033-5177.

[79] Steven Scalet. Prisoner's dilemmas, cooperative norms, and codes of business ethics. *Journal of Business Ethics*, 65(4):309–323, 2006. ISSN 1573-0697.

[80] Meet Shah, Mohammedhasan Shaikh, Vishwajeet Mishra, and Grinal Tuscano. Decentralized cloud storage using blockchain. In *2020 4th International Conference on Trends in Electronics and Informatics (ICOEI)(48184)*, pages 384–389. IEEE, 2020. ISBN 1728155185.

[81] Shen-Guan Shih, Tsung-Pao Hu, and Ching-Nan Chen. A game theory-based approach to the analysis of cooperative learning in design studios. *Design Studies*, 27(6):711–722, 2006. ISSN 0142-694X.

[82] Ji Shou-Wen, Tian Ying, and Gao Yang-Hua. Study on supply chain information control tower system. *Information Technology Journal*, 12(24):8488, 2013. ISSN 1812-5638.

[83] Sylvain Sorin. Classification and basic tools. In *Stochastic Games and Applications*, pages 27–36. Springer, 2003.

[84] Charalambos Spathis and Sylvia Constantinides. The usefulness of ERP systems for effective management. *Industrial Management & Data Systems*, 2003. ISSN 0263-5577.

[85] Thongchai Srivardhana and Suzanne D Pawlowski. ERP systems as an enabler of sustained business process innovation: A knowledge-based view. *The Journal of Strategic Information Systems*, 16(1):51–69, 2007. ISSN 0963-8687.

[86] Vanessa Stricker, Kim Lauenroth, Piero Corte, Frederic Gittler, Stefano De Panfilis, and Klaus Pohl. Creating a Reference Architecture for Service-Based Systems-A Pattern-Based Approach. In *Future Internet Assembly*, pages 149–160, 2010.

[87] Reima Suomi. On the concept of inter-organizational information systems. *The Journal of Strategic Information Systems*, 1(2):93–100, 1992. ISSN 0963-8687.

[88] Wei Tan, Yushun Fan, Ahmed Ghoneim, M Anwar Hossain, and Schahram Dustdar. From the service-oriented architecture to the web API economy. *IEEE Internet Computing*, 20(4):64–68, 2016. ISSN 1089-7801.

[89] J Michael Tarn, David C Yen, and Marcus Beaumont. Exploring the rationales for ERP and SCM integration. *Industrial Management & Data Systems*, 2002. ISSN 0263-5577.

[90] Marcel te Lindert. Control towers are emerging everywhere. 2013.

[91] Wolfgang Theilmann, Jens Happe, Costas Kotsokalis, Andrew Edmonds, Keven Kearney, and J Lambea. A reference architecture for multi-level sla management. *Journal of Internet Engineering*, 4(1):289–298, 2010.

[92] Engin Topan, Ayse Sena Eruguz, Weina Ma, M C Van Der Heijden, and Rommert Dekker. A review of operational spare parts service logistics in service control towers. *European journal of operational research*, 282(2):401–414, 2020. ISSN 0377-2217.

[93] Anna Trzuskawska-Grzesińska. Control towers in supply chain management–past and future. *Journal of Economics and Management*, 27(1):114–133, 2017. ISSN 17321948. doi: $10.22367/\mathrm{jem}.2017.27.07$.

[94] Osman Turan, A İ Ölçer, Iraklis Lazakis, Philippe Rigo, and Jean-David Caprace. Maintenance/repair and production-oriented life cycle cost/earning

model for ship structural optimisation during conceptual design stage. *Ships and Offshore Structures*, 4(2):107–125, 2009. ISSN 1744-5302.

[95] Marshall Van Alstyne, Erik Brynjolfsson, and Stuart Madnick. Why not one big database? Principles for data ownership. *Decision Support Systems*, 15(4): 267–284, 1995. ISSN 0167-9236.

[96] Rakesh Verma, Saroj Koul, and Gurpreet Singh. Intelligent Decision-Making: Using Control Tower at a Logistics Company. In *2020 IEEE International Conference on Computing, Power and Communication Technologies (GUCON)*, pages 550–554. IEEE, 2020. ISBN 1728150701.

[97] Kristin Weber, Boris Otto, and Hubert Österle. One size does not fit all—a contingency approach to data governance. *Journal of Data and Information Quality (JDIQ)*, 1(1):1–27, 2009. ISSN 1936-1955.

[98] Bernhard Wieder, Peter Booth, Zoltan P Matolcsy, and Maria-Luise Ossimitz. The impact of ERP systems on firm and business process performance. *Journal of Enterprise Information Management*, 2006. ISSN 1741-0398.

[99] Roel J Wieringa. *Design science methodology for information systems and software engineering*. Springer, 2014. ISBN 3662438399.

[100] Robert H Wozniak. Introduction to memory: Hermann ebbinghaus (1885/1913). *Classics in the history of psychology*, 1999.

[101] Wei Xiao, Yaqing Tu, Ping Wan, Ming Li, and Jingheng Ma. Analysis of Data Ownership Rights in the Big Data Era. In *Proceedings of the 2020 4th International Symposium on Computer Science and Intelligent Control*, pages 1–5, 2020.

[102] Yale Yu, Haydn Silveira, and Max Sundaram. A microservice based reference architecture model in the context of enterprise architecture. In *2016 IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC)*, pages 1856–1860. IEEE, 2016. ISBN 1467396133.

[103] Cheng Zhang, Ling Xue, and Jasbir Dhaliwal. Alignments between the depth and breadth of inter-organizational systems deployment and their impact on firm performance. *Information & Management*, 53(1):79–90, 2016. ISSN 0378-7206. doi: 10.1016/j.im.2015.08.004.

[104] Kevin Zhu, Kenneth L Kraemer, Vijay Gurbaxani, and Sean Xin Xu. Migration to open-standard interorganizational systems: network effects, switching

costs, and path dependency. *MIS Quarterly*, 30:515–539, 2006. ISSN 0276-7783.

Appendix A

Methodology dilemma construction

For the section 'Dilemma's in Service Control Tower development' a workshop has been held in order to retrieve information from the consortium about the dilemma's they face. Earlier inspiration for this workshop was retrieved in workshops on the Control Tower architecture itself. Participants often state difficult decisions they face in development of such a system, hence, we formulate them as a dilemma.

We organise an interactive online brainstorm session, containing two parts per topic. The main method for brainstorming is brain writing. Brain writing contains two phases, first ideas are generated by the participants based upon a certain topic or question. In a later stage, the ideas are discussed in a plenary session, thereby generating more ideas and potential solution concepts.

Nine participants of the MARCONI consortium joined the session, with different backgrounds (e.g., service design, business manager, researcher).

The following questions were asked as a beginning for the discussion:

- Which Technological Control Tower development dilemma's do you face? (Minimal three answers necessary)
- Which Business dilemma's do you face when developing a Control Tower? (Minimal three answers necessary)

The result is a list of dilemma's that have been categorized and used as input for the discussion. From the discussion, most of the dilemmas in the chapter have been derived.

Last but not least, we ask participants to choose in a dilemma between two alternatives. See the table below for the results.

As one can see in the table above, results on the presented dilemmas are mixed. Which fueled a discussion on which solution is better. To conclude, we see evidence that consortium partners are divided in their opinion about the 'optimal' solution. Therefore, the dilemmas we constructed have practical but also academic relevance.

Dilemma	Option A	Option B	Results
Decision making in a control tower should be	Centralized	Decentralized	37.5% - 62.5%
Onwership of a control tower should be	Centralized	Decentralized	37.5% - 62.5%
Decision making optimization techniques in a control tower environment should be	Exact	Self-Learning	44.4% - 55.6%
Data sharing in a control tower should be	Real-time	On-demand	44.4% - 55.6%
A control tower should be developed	In-house	Outsourced	66.7% - 33.3%
Execution of a Service Level Agreement in a control tower should be	Rule-based (Smart Contracts)	Trust-based	77.8% - 22.2%
A control tower solution should be	A generic solution	Tailormade	22.2% - 77.8%

Table A.1: Results dilemma's presented

Appendix B

MARCONI system needs questionnaire

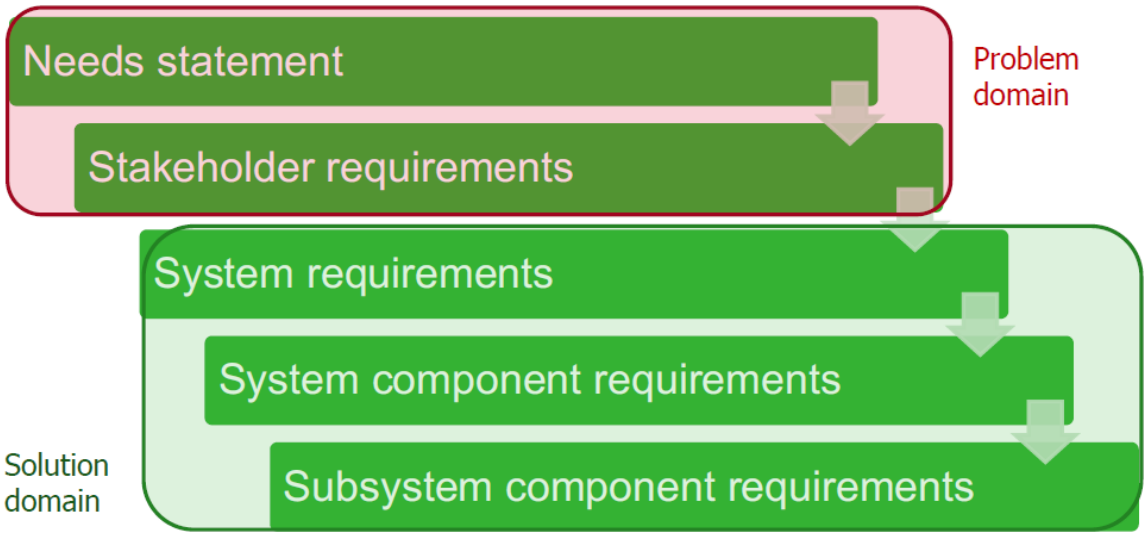
MARCONI Control Tower: From needs to requirements

Dear MARCONI participant,

Thanks a lot for taking the time to fill in this questionnaire regarding the development of the MARCONI Control Tower. In this questionnaire, we would like to ask you to define some needs from the perspective of your organization regarding the MARCONI control tower. In this questionnaire, we hope you can give us input to feed the process that is stated in the image below.

* Required

1. Email *



Needs engineering

The idea is that we look from a problem perspective in this questionnaire. The MARCONI control tower is the solution that we are going to develop, but which problem is it actually going to solve? Therefore we would like you, as an expert in the organization you are operating, to define some needs for the MARCONI control tower from the perspective of your organization.

First of all, we define a need as an agreed expectation for the entity to perform some functions.
In our case, the entity is the MARCONI control tower. We would like you to think of needs, in order to give you a headstart, please look at the following questions you can ask yourself in order to trigger the thinking process:

"What do you want from the MARCONI control tower?"
"What functions do you like to have within the MARCONI control tower?"
"What do you need from the MARCONI control tower?"
"What is the purpose of the MARCONI control tower for your organization?"
"How would you want the MARCONI control tower to work?"
"How should the performance of the MARCONI control tower be?"

2. Please define at least three needs with a maximum of five needs

Requirements engineering

In the previous step you have defined a few needs from the perspective of your organization. We would like to ask you to transform these needs into requirements. But first, we provide you a definition of a requirement (IEEE 1998):

Requirement: a statement that identifies a product or process operational, functional, or design characteristic or constraint, which is unambiguous, testable or measurable, and necessary for a product or process acceptability (by consumers or internal quality assurance guidelines).

When transforming these needs into requirements we propose to apply a technique that is called formal transformation. The formal transformation technique describes characteristics to which a good requirement is separated from a mediocre one (see: https://reqexperts.com/2017/07/15/incose-guide-for-writing-requirements-2017-update/). These characteristics are, including a short description:

- Necessary: A requirement should be necessary, which means that it is essential to be part of the control tower that we are going to develop. If it can be removed without any consequence to the system, it probably is not a necessary requirement.
- Singular: The requirement should apply to a single aspect, characteristic, capability or function of the system. The presence of an 'and' in the requirement should be avoided in order to keep the requirement singular.
- Conforming: The requirements should be defined in a uniform way of writing which should help to increase the understanding of the requirement when verifying them.
- Appropriate: The requirement should reflect the level of detail we are talking on, in this case, the requirement should be on the scope of the MARCONI control tower as a system.
- Unambiguous: The requirement should be clear and not leave room for multiple interpretations.
- Complete: The requirement should contain the necessary information required to understand the requirement.
- Correct: The requirement should be applicable to one of the needs earlier defined.
- Verifiable: The requirement should be linkable to developed solutions and proven to have been met or not.
- Feasible: The requirement should be realizable within the constraints of the project.

Last but not least, we give a short example of a well-written requirements:
- Move people from Enschede to Den Haag

Please try to keep in mind these characteristics as a reminder to write good requirements. It is hard to define them really well, but you do not have to exactly apply all of them perfectly. We will also try to further refine them based on your input.

3. Please define requirements, based on the earlier stated needs and characteristics

Thank you!

Thank you very much for taking the time to fill in the form. Your input is valuable to us, not only does it provide us a direction for development of the demonstrator enviroment and IT infrastructure. It could also feed into the work of the other workpackages.

On behalf of WP3, kind regards,
Rogier Harmelink

Appendix C

Raw system needs MARCONI

In this research we have conducted a questionnaire and also had multiple plenary (and Skype) sessions. Based on this, we derived needs. We describe these needs as an expectation in the form of ”MARCONI should“. The list here below is an unstructured list of needs derived from the sessions.

- MARCONI should optimize the supply chain planning at tactical and operational level
- MARCONI should enhance visibility and the links in the supply chain network
- MARCONI should offer the possibility for a supply chain collaboration (NVP).
- MARCONI should offer operators the possibility to intervene in short-term decision making with predefined rules
- MARCONI should measure service contract performance
- MARCONI should offer the possibility for KPI reports
- MARCONI should offer the possibility to add KPI’s
- MARCONI should provide real-time supply data
- MARCONI should provide real-time demand information
- MARCONI should identify gaps in demand and supply
- MARCONI should generate alerts to prevent reduction in availability
- MARCONI should provide a test environment for reactive and proactive interventions on short term system performance
- MARCONI should offer the possibility of integrated planning

- MARCONI should offer the possibility of obsolescence management
- MARCONI should offer a secure environment
- MARCONI should offer a connection to internal IT systems/infrastructure of participants
- MARCONI should offer a recommender system
- MARCONI should offer insight in reliability of the recommendations
- MARCONI should offer possible actions to follow up on the recommendations
- MARCONI should offer a dashboarding environment (Business Intelligence)
- MARCONI should offer the possibility for life cycle management
- MARCONI should give insight in fuel and emission metrics
- MARCONI should give the possibility to go from a stovepipe system to a more cross-functional system
- MARCONI should provide an overview of people and processes involved in the service logistics in company
- MARCONI should provide an end-to-end overview of the service logistics supply chain
- MARCONI should align logistics processes between supplier and customer
- MARCONI should give an indication of lead time on (spare) parts
- MARCONI should give information about possible suppliers of (spare) parts
- MARCONI should give information about the repair or replace process of parts of systems
- MARCONI should give information on preventive or corrective maintenance taking place
- MARCONI should give information about the potential cause of failure in case of corrective maintenance
- MARCONI should provide optimized maintenance plans including specific tasks and indication on resources needed
- MARCONI should provide decision support models (algorithms)

- MARCONI should have adaptable algorithms for optimization
- MARCONI should data download and upload functionality
- MARCONI should be able provide a GUI for system interaction with planning models
- MARCONI should support a network of organizations
- MARCONI should contain to option to construct a digital twin
- MARCONI should support machine learning algorithms
- MARCONI should reduce planning workload
- MARCONI should achieve reduction of disruption of primary activities
- MARCONI should provide a track and trace functionality
- MARCONI should give the asset-owner an overview of the core operations
- MARCONI should in real-time show the operations of the assets
- MARCONI should facilitate the addition of sensors on assets linked to the control tower
- MARCONI should offer the possibility to actually control equipment
- MARCONI should offer the owner of the asset to use pluggable intelligent tools or not

Appendix D

Service Control Tower entities, functionalities and requirements

Entity/*function/* requirement

Asset Management Software
Constructs digital twin
Connects to data source of asset
Simulates the asset
Logs the history of an asset
Visualizes the asset in the system

Provides lifecycle management
Visualizes life cycle of asset
Connect with lifecycle management software
Contains systems engineering tool

Provides test environment for interventions on assets
Draw causal relations
Contains simulation tools
Supports multi agent systems

Integrates planning of assets
Connected to operations schedules
Compute optimized planning

Obsolescence Management Software
Manages obsolescence
Shows alternative spare parts
Notifies discontinuation of parts
Link with bill of materials
Computes obsolescence risk
Links with inventory management

Computerized Maintenance Management Software
Provides specific tasks and resources needed for maintenance
Contains bill of materials
Plans use of equipment
Links with inventory management
Links with HRM

Generates alerts when availability is reduced
Sends notification if alert is not crititical
Sends alert to key user if alert is critical
Retrieves loss in availability
Send alert over SMS
Send alert over email

Integrates planning of maintenance
Connected to maintenance schedules

Business Intelligence Software
Preprocesses data
Cleanses data

Checks data requirements
Able to handle datasets of at least 1 TB
Transforms data

Offers customizable KPI reports
Select KPI's
Links data with KPI's
Able to insert KPI's
Retrieves KPI's from SLA's

Provides dashboard environment
Constructs a dashboard
Shows alerts
Creates visual representations of data
Filters data based on user input

Provides smart algorithms
Contains self learning algorithms
Connect to algorithm API's
Contains mathematical optimization

Provides data upload and download functionality
Download files
Upload files
Able to process .csv files
Able to download data only when permission is granted

Supply Chain Management Software
Optimizes supply chain
Increase availability of assets
Reduces waste
Reduces bullwhip effect

Offers GUI for decision interaction
User can influence his processes
Visualizes results of decisions
Predetermined rules in system

Provides visibility of the supply chain
Visual mapping of the supply chain(s)
Visualization of relations between processes
Visualizes business processes
Visualization of relations between users

Identifies gaps in supply and demand
Store information on supply
Compute gap in supply and demand
Store information on demand
Recognizes supply or demand based on user role

Provides possibility to intervene in decision making

Shows impact on intervention
User can overrule decision
Simulates decision impact

Service Management Software
Provides service contract performance information
Extracts information from service contracts
Computes SLA KPI's
Stores service contracts
Links with data of service processes

IT Infrastructure
Links with IT infrastructure of participants
API available to connect with ERP systems
Encrypted connection with participants
Connection with participants based on governance rules
Connects to data warehouses

Collaboration Software
Provides supply chain collaboration
Platform for knowledge sharing
File sharing
Share messages between users
Joint decision making environment

Data Governance Software
Provides a secure environment
Encrypts data
Encryption library available
Encryption is at least 128 bits

Governs data usage
Imposes rules on data access
Adjustable data governance rules
Tracks data usage

Track and Trace System
Provides real time information
Show information with a delay of at most 1 minute
Handles GPS data
Retrieves information within 30 seconds
Store information within 45 seconds

Recommender System
Offers insight in the reliability of recommendations
Show recommendations
Compute reliability of recommendations
Filter recommendations
Adjust on criteria

Offers possible actions to follow up on recommendations
Confirm recommendations
User divided actions
Action generation
Links with business processes

Appendix E

Service Control Tower MoSCoW results

Here below we show the results of the MoSCoW workshop. In case of a draw, the researcher of this work has the determining vote.

Software Element	Must Have	Should Have	Could Have	Won't Have
Collaboration Environment	3	**3**	2	0
Data Governance	**7**	1	0	0
Service Management	1	2	**4**	0
Business Intelligence	**5**	3	0	0
Track and Trace	2	**6**	0	0
Recommender System	1	2	**5**	0
Asset Management	2	**4**	1	1
Maintenance Management	3	**4**	1	0

Appendix F

Methodology Service Control Tower as a Service

In this section, we describe the methodology and workshop organized in order to retrieve input for the section on 'Service Control Tower as a Service'.

The workshop is held within the MARCONI consortium, containing input of nine participants, with a different range of backgrounds (i.e., service designer, business manager, asset manager and researcher).

First, we use discuss the reference architecture of the Service Control Tower. We talk about individual software elements and create a common understanding of the architecture. Next, we use the technique of brain writing to answer the questionnaire. After answering of the question, we discuss the results within the group in order to find new interesting results/solutions.

All software elements are discussed. The following software elements have been used from the architecture: Inventory Management, Business Intelligence, Track and Trace System, Recommender System, Service Management, Data Governance Software, Asset Management, Computerized Maintenance Management Software, Obsolescence Software and Collaboration Software.

We ask two questions regarding each software element to the participants, and discuss the results:

- Who do you think could deliver SOFTWARE ELEMENT best in the supply chain network?
- How would you measure the performance of SOFTWARE ELEMENT

We describe the answers given and discuss them of multiple software elements, here below. However, due to limited time and lengthy discussions, not all software elements were discussed in the consortium. For the remaining elements, the author brainstormed on suitable performance measurements that have been checked with literature.

Inventory Management

Question 1: Who should deliver inventory management in the network:

Answers: Asset owner, OEM or sub supplier and collaborative based upon agreed rules.

Discussion: Different participants give different reasons for their selection. Mostly also from their own point of view (i.e., a participant works at an asset owner or an OEM). Therefore it is hard to distinguish based on this discussion, which supply chain participant should take on inventory management.

Question 2: How would you measure the performance of Inventory Management?

Answers: Dashboard, just in time delivery, monthly weighed inventory, unavailability, decrease percentage, mean logistic delay time, availability, reliability, cost.

Discussion: Multiple performance indicators have been provided by the participants, which are used as input for the modelling.

Maintenance Management Question 1: Who should deliver maintenance management in the network:

Answers: Asset owner, maintainer, operations planner, maintenance facility manager

Discussion: Again, we receive multiple arguments for multiple stakeholders in the network. No clear generalised answer has been given by the participants.

Question 2: How would you measure the performance of Maintenance Management?

Answers: Uptime, live up to the expectation, asset availability, downtime, cost, amount of unexpected downtime, maintenance downtime, availability, performance

Discussion: Multiple performance indicators have been provided by the participants, which are used as input for the modelling.

Obsolescence Management Question 1: Who should deliver obsolescence management in the network:

Answers: Supply chain in all tiers, service provider, OEM, collaborative, asset owner, entity that knows the product best.

Discussion: Again, we see similar answers to the question with regards to obsolescence management and this question. Also, some answers are rather vaguely defined, again, resulting in an answer that is hard to generalize.

Question 2: How would you measure the performance of Obsolescence Management?

Answers: Uptime, absence of unavailability, accuracy of obsolescence prediction, number of unexpected failures, downtime related to obsolescence management

Discussion: Multiple performance indicators have been provided by the participants, which are used as input for the modelling.

Business Intelligence Question 1: Who should deliver business intelligence in the network:

Answers: Asset owner, 3rd party, everybody, each role should deliver his portion.

Discussion: We see again different answers, making it hard to generalise.

Question 2: How would you measure the performance of Business Intelligence?

Answers: Decrease of costs, board satisfaction, number of interventions triggered, data reliability/availability.

Discussion: Multiple performance indicators have been provided by the participants, which are used as input for the modelling.