UNIVERSIDADE ABERTA INSTITUTO SUPERIOR TÉCNICO

The role of Enterprise Architecture Management in Microservice Architecture adoption: The key aspects to govern the Microservice Architecture

Carlos Roberto Pinheiro

Master's Degree in Information and Enterprise Systems

(master's degree in association)

2019

UNIVERSIDADE ABERTA INSTITUTO SUPERIOR TÉCNICO

The role of Enterprise Architecture Management in Microservice Architecture adoption: The key aspects to govern the Microservice Architecture

Carlos Roberto Pinheiro

Master's Degree in Information and Enterprise Systems

(master's degree in association)

Thesis supervised by:

Professor Doutor Sergio Luís Proença Duarte Guerreiro

Professor Doutor André Vasconcelos

2019

Resumo

Arquitetura de Microserviços (MSA) é um estilo arquitetural para construção de sistemas distribuídos através de um conjunto de pequenos serviços que podem ser publicados de modo totalmente independente. Ao adotar a arquitetura de microserviços, as empresas precisam controlar alguns aspetos que possuem impacto na eficiência organizacional a fim de garantir *(i)* os benefícios estratégicos da iniciativa; *(ii)* promover o melhor uso dos recursos e *(iii)* separar as decisões essenciais da Arquitetura Corporativa e delegar outras decisões as equipas responsáveis pelos microserviços. Este trabalho investiga os fatores relevantes acerca da Arquitetura de Microserviços na perspectiva da Gestão da Arquitetura Corporativa (EAM) para propor um modelo de arquitetura em ArchiMate que pode ser utilizado como um *template* para a Arquitetura Corporativa das empresas. O modelo resultante suporta duas diferentes abordagens, um top-down e outra bottom-up, que se complementam na função de planejar e de manter o modelo de arquitetural atualizado. Por fim, (i) um modelo definindo princípios e escopo de governança, (ii) uma estrutura genérica de equipas, and (iii) uma referência arquitetural de tecnologias padrões são construídos para atender ao novo papel do gerenciamento da arquitetura corporativa através da governança descentralizada para suportar as equipas de microserviços com um caráter menos intrusivo e menos restritivo.

Keywords: Enterprise Architecture Management, Microservice Architecture, Service Oriented Architecture, Adaptive Enterprise Architecture, Adaptable Enterprise Architecture.

Abstract

Microservice Architecture (MSA) is an architectural style that aims to build a software application as a set of small services independently deployable. When adopting MSA, companies must drive some aspects that impact the organizational efficiency in order to guarantee *(i)* the strategic benefits of the initiative; *(ii)* promote the best resources usage, and *(iii)* separate the essential decisions to Enterprise Architecture Management (EAM) and postpone the other aspects to the microservice teams' decisions. This work investigates the relevant factors about MSA from the EAM perspective in order to propose an ArchiMate model architecture which could be used as a template for companies' Enterprise Architecture (EA). This model includes principles, responsibilities, team structure, a topologic view of technologies and standards, and a way to delimit system boundary and maintain the model up to date. The resulting model supports two different approaches, one top-down and another bottom-up, which complement each other in the function of planning and keeping up to date the Enterprise Architecture (EA) models. *(i)* A model defining principles and governance guidelines, *(ii)* a generic team structure and *(iii)* an architectural reference for technology standards which enable the enterprise governance of MSA, are engineered to support the new role of EAM in a decentralized governance to govern microservice teams with a less intrusive and less restrictive role as possible. In the end, the model was submitted to evaluation by teams of companies which adopted MSA, evaluating the contribution in reduce risks related to the aspects identified. The evaluation proved the usefulness of the model designed.

Keywords: Enterprise Architecture Management, Microservice Architecture, Service Oriented Architecture, Adaptive Enterprise Architecture, Adaptable Enterprise Architecture.

Dedication

To my daughter Melina Pinheiro, I dedicate this work as an effort to leave her records that will serve as an example and a memory of my passage in this world.

Acknowledgements

I thank my advisors for their guidance, availability, patience and commitment, without which it would not be possible to complete this journey. I Also, thank you to my colleagues of course for motivation and change of ideas.

Special thanks to the Portuguese people for believing in science and opening the opportunity I had to do this master's degree at Universidade Aberta. It is a pleasure to see the Portugal's effort to fulfill its historic leading role in the advancement of science. Remembering the historical contributions to humanity, such as the discovery of new worlds, my own country, prove the circumnavigation and much more, and recently becoming a reference in the field of information technology. Being part of this is something that makes me very proud, even if the contribution is quite modest.

I thank my wife Meg Ferreira for all the support, for understanding and for helping me realize this dream.

Circumstantial Specifications

This work was supported by national funds through Fundação para a Ciência e a Tecnologia (FCT) with reference UID/CEC/50021/2019 and by the European Commission program H2020 under the grant agreement 822404 (project QualiChain).

Table of Contents

List of Tables

List of Figures

Glossary of Abbreviations and Acronyms

AIS	Association for Information Systems
API	Application Program Interface
APM	Application Performance Management
BPMN	Business Process Model and Notation
BRMS	Business Rule Management Systems
COBIT	Control Objectives for Information and Related Technologies
CI/CD	Continuous Integration / Continuous Delivery
CNCF	Cloud Native Computing Foundation
CRM	Customer Relationship Management
DDD	Domain Drive Design
DevOps	Development and Operations
DS	Design Science
DSR	Design Science Research
DSRM	Design Science Research Methodology
EA	Enterprise Architecture
EAM	Enterprise Architecture Management
ERP	Enterprise Resource Planning
ESB	Enterprise Service Bus
IAM	Identity and Access Management
IS	Information System
IT	Information Technology
IoT	Internet of Things
MSA	Microservice Architecture

SME	Small and Medium-sized Enterprise
SOA	Service Oriented Architecture
SysML	Systems Modeling Language
TOGAF	The Open Group Architecture Framework
UML	Unified Modeling Language

1 Introduction

Microservice Architecture (MSA) has aroused a great interest recently (Francesco, Malavolta, & Lago, 2017; Lenarduzzi & Sievi-Korte, 2018; Soldani, Tamburri, & Van Den Heuvel, 2018; Thomas, 2018). It aims to build distributed systems based on small and independent services. In contrast to Service Oriented Architecture (SOA), which commonly is built under a strong and centralized governance, most of MSA references advocate for decentralized governance and this decentralization can hinder communication at the enterprise level (Lenarduzzi & Sievi-Korte, 2018). On the other hand, although each microservice individually presents low complexity, microservice-based systems as a whole become highly complex due to the heterogeneity of technology, volatility and high granularity of its components. Architecture is related to "the fundamental organization of a system, embodied in its components, their relationships to each other and the environment, and the principles governing its design and evolution" (Bischoff, Aier, & Winter, 2014 apud ISO/IEC/IEEE, 2011, 134). Enterprise Architecture Management (EAM) is related to the management process of Enterprise Architectures (EA) (Bischoff et al., 2014). The decentralization, volatility and high granularity of MSA demand advanced EAM approaches in Information System (IS) architecture in order to integrate a large number of small structures within an adaptive Enterprise Architecture (EA), mainly due to the lack of modelling elements and patterns that represent all the diversity of architectural descriptions to be integrated (Bogner & Zimmermann, 2016a). Nonetheless, it is important to manage and ensure the alignment and integration between the modeling of microservice-based systems and the EAM by several factors, such as planning as controlling the proliferation and dependencies of IS, maintaining IS efficiency and effectiveness, contribute to organization's business value, and coordinating the architectural development across levels and departments of the organization, among others (Weiss, Aier, & Winter, 2013). Therefore, this research assesses the impacts about EAM concerns of the scope definition, technologies heterogeneity, cost volatility, and decentralized governance of (MSA), and address them in a model based on ArchiMate (The Open Group, 2017) and TOGAF 9.2 (The Open Group, 2018), in order to serve as a template for EA modeling that pursues to ensure

the alignment between EAM needs and MSA implementation and therefore, it targets a contribution to the development of the Enterprise Architecture (EA) body of knowledge.

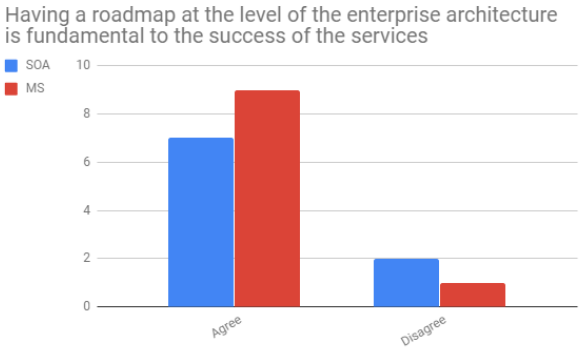
1.1 The Problem

The problem addressed by this dissertation is the lack of modelling elements to describe the microservices architecture at the enterprise level within an EA Model. Thus, we aim to identify what should be observed and documented in the EA when adopting a MSA in order to guarantee the alignment of the architectural layers and aspects sensitized in the EA by MSA, based on TOGAF (The Open Group, 2018) and how to structure it in ArchiMate (The Open Group, 2017).

In order to confirm the forehand problem, a survey was sent to specialists from several hierarchical levels involved in initiatives where SOA or MSA were adopted, including different architects' roles, developers and managers. SOA is an architectural style that focus on interoperability among services, while MSA is a style of architecture for distributed systems based on microservices (Yale Yu, Silveira, & Sundaram, 2016). The involvement of SOA practitioners with Enterprise Service Bus (ESB) would serve two purposes, first to broaden the target audience on aspects relevant to both ESB services and microservices, second to turn the SOA service group into a basis for comparison and qualitative analysis of the responses obtained of the group of practitioners of microservices. But only 18 responses in total were obtained, 9 from SOA practitioners and 9 from microservice practitioners. Unfortunately, this is a small number that did not allow us to take definitive conclusive insight. Nevertheless, it provides some partial views that seem relevant and are placed here.

84% of the experts answered that it is important to have a service roadmap at the EA level and this is reinforced by the number of respondents of the microservice group as show in Figure 1.1.

Figure 1.1 - The Importance of EA roadmap



From the experts with experience in micro services, only 33.1% stated that development is product-oriented. It implies that organizations seem to face difficulties to deploy a new framework that supports product vision and they continue to adopt a project-oriented implementation.

Regarding strategic alignment, 40% answered that it is done through departmental definitions. Although the majority seems to somehow seek alignment at the enterprise level, this percentage seems large and indicates a difficulty in enterprise communication in relation to assure that the initiatives are aligned with the company's business strategies and needs, at least in terms of enterprise IT Governance.

On the other hand, 80% agree that microservices teams should follow practices and standards defined in a centralized reference architecture

Finally, we concluded that the analysis of some of the questions answered contributed to reinforce the suspicion of the existence of difficulties at EAM level to govern MSA initiatives. These difficulties seem to be linked to the problem of having the lack of modeling elements to document, visually address EAM concerns and provide consistent architectural views on MSA. In order to solve this problem, we need answer the following research questions:

Q1 - Which aspects of MSA are relevant to EAM and should be described in an EA model in order to guarantee the management of IS architectural components dependencies, and their efficiency and effectiveness at enterprise perspective?

Q2 - How to describe these MSA aspects, which are relevant to EAM's context, in a TOGAF and ArchiMate architectural models?

1.2 Objectives

In order to help solving the lack of modeling elements that allow to architect and manage EAM concerns for MSA, the objective of this research is to investigate the relevant factors about MSA from the EAM perspective and to prescribe a model to address these factors in a reference architecture of microservice-based information systems which can serve as a template for EA modeling by any company. Therefore, we intended to create a reference model based on ArchiMate (The Open Group, 2017) and TOGAF 9.2 (The Open Group, 2018) for microservice-based information systems architecture, pursuing to ensure traceability between EAM and the identified relevant aspects of the microservices architectural components and properties.

1.3 Research Method

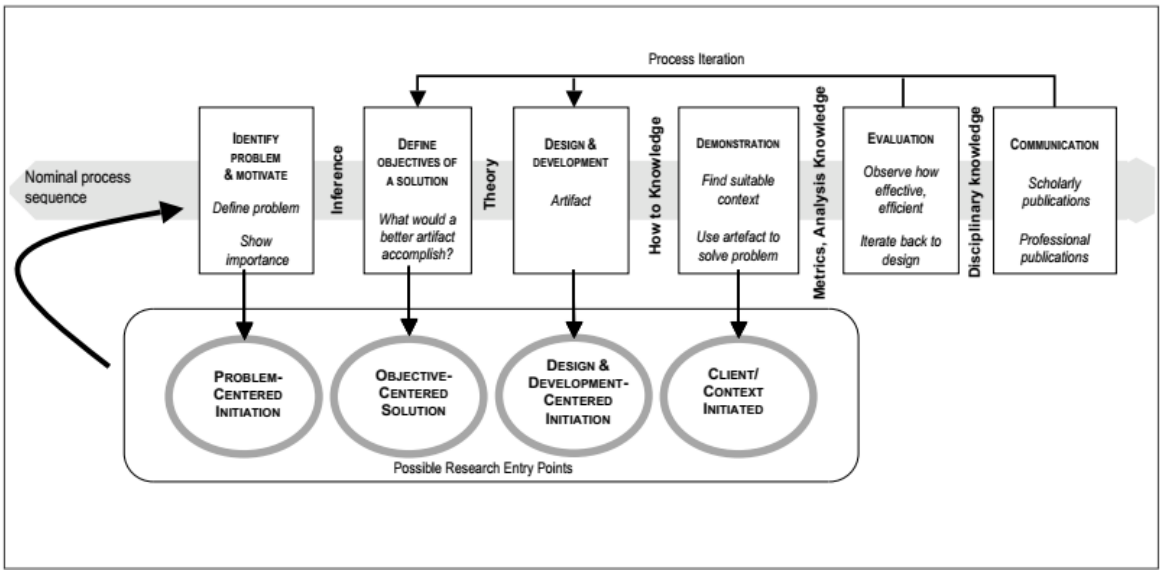
It is clear the need to develop research with theoretical and methodological rigor in order to guarantee a relevant work (Dresch, 2015). Design Science Research Methodology for Information Systems Research (DSRM) is the method chosen for this research. In order to justify the choice, the main motivations and the research process will be briefly described here.

Design Science is a research paradigm that looks for answering relevant questions for human problems through innovative artifacts, contributing to new knowledge (Hevner & Chatterjee, 2010) on how to design and how to apply it. It is therefore about designing solutions to improve existing systems, solve problems, or contribute to a better human performance in society or organizations (Dresch, 2015).

In this research we propose as an artifact a model with the representation of aspects of MSA relevant to the EAM and a method to apply these aspects in the EA modeling. So, the context is aligned with the purpose of the DSRM and its process will be followed through the structure of this document.

By this research method a viable artifact should be produced, which is something artificially built or adapted by humans to reach a goal. Figure 1.2 illustrates the general instructions for conducting and evaluating the research by this method.

Figure 1.2 - DSRM Process Model



Source (Peffers, Tuunanen, Rothenberger, & Chatterjee, 2007)

The DSRM research process prescribes principles, practices, and procedures to conduct researches, which include six sequential activities described by Hevner (Hevner & Chatterjee, 2010) as:

Activity 1: *Identify Problem and motivation.* Defines the specific problem of the research and justifies the value of a solution. In this research, subsections 1.1 and 0 of the introduction, provides an overview of the problem and the motivation that justify the research.

Activity 2: *Define Objectives of a Solution*. Infers the objectives of the solution from the problem definition and the knowledge about what is achievable. Sections 2 and 3 present the theoretical base to define the solution objectives that can be achieved.

Activity 3: *Design and development*. Creates the artifact, such as constructs, models, methods or instantiations. In this work this activity is described in section 4, which we divided in tree subsections. Subsection 4.1 that defines the requirements, 4.2 that develops a solution proposal to meet the objectives, meeting the requirements, and 4.3 which assess the feasibility of the proposal.

Activity 4: *Demonstration*. Presents the use of the artifact to solve one or more instances of the problem. This activity is realized in section 5, which demonstrates an application of the solution proposed to a hypothetical case.

Activity 5: *Evaluation*. Observes and measures how well the artifact supports a solution to the problem. Section 6 present the survey and interviews with experts involved in MESA initiatives targeting to assess the usefulness of the solution proposed.

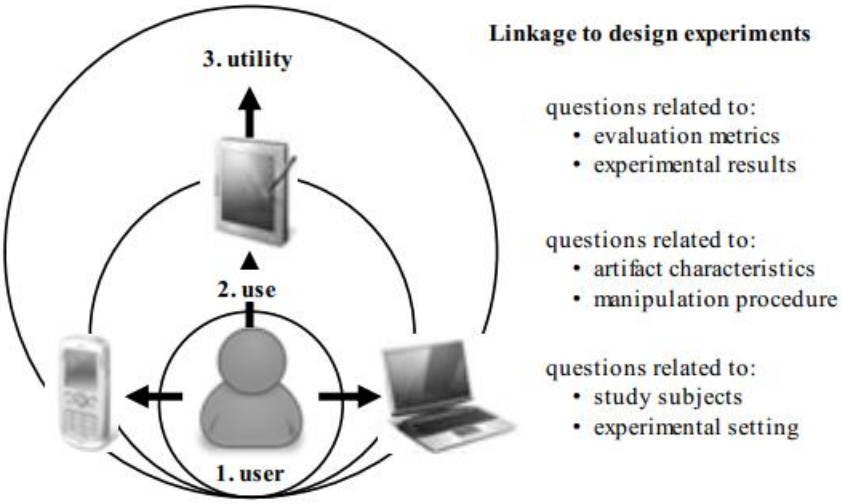
Activity 6: *Communication*. Communicates the problem and its importance, the artifact, its utility and novelty, the rigor of its design, and its effectiveness to researchers and other relevant publics. Section 6.4 describes how this research was communicate and section 8 presents the conclusions with majors' contributions, limitations e future work.

Still in the field of DSR, it is important to demonstrate the quality and effectiveness of the research in order to ensure the credibility of the research and its results. Different methods can and should be used for work validation, including case study, controlled experiments, simulations and prototypes. It is important to (Dresch, 2015):

1- Make explicit the internal, external environment and objectives.
2- Inform how the artifact can be tested.
3- Describe how the results should be generated and controlled.

Thereby, the DSRM prescribes a fundamental step of evaluation in order to prove that the results reached fulfilled their objectives (Hevner & Chatterjee, 2010; Mettler, Eurich, & Winter, 2014). To contribute to the quality of the research artifacts, Metter (Mettler, Eurich, & Winter, 2014) proposed a framework that aims to validate research experiments developed through DSR, testing the requirements and analyzing the impact of the solution in reality. The method proposed consists of seeking to answer some questions grouped in three perspectives: the user, the use, and the utility of the work, as shown in Figure 1.3, which illustrates their evaluation framework under these perspectives.

Figure 1.3 - Validation Framework



Source (Mettler et al., 2014)

1- **User**: The user determines how an artifact is used and what benefit is perceived. Being the DSR a method oriented to utility, a design experiment is successful when users deem the artifact to be of superior utility and its value is demonstrated.
2- **Use**: It is essential to provide detailed information on the use of the artifact by users. This perspective defines and describes the objectives and scope of use helps to maintain focus through parallel interactions, naturals in design experiments. The objectives may be to prove that a new artifact solves a problem or that an existing artifact works better than another existing solution.
3- **Utility**: It emerges using the artifact and it is dependent on the user and the environment. To operationalize the evaluation of the utility of an artifact it is necessary to clearly establish measurable variables (metrics) and evaluation criteria in order to allow other researchers to replicate the experiment.

We used these perspectives to build the questionnaire and the keep the focus of interview to evaluate the utility of proposal to model the EAM concerns over MSA(use) by enterprise architects (user), which is described in Section 6.

1.4 Document Structure

To answer the research questions presented in section 1.1 and achieve the research objective, the rest of this document is organized as following: **Section 2** presents the background which compiles the main references of the knowledge that support the problem definition and the proposed solution, therefore in this section, we present the main concepts, frameworks e definitions related to the EAM and MSA; **Section 3** presents related works containing a literature review that shows recent researches in the field with a brief summary of their contributions to this work. Along with background serves as basis for this research; **Section 4** presents the modelling proposed solution; **Section 5** demonstrates the application of modelling proposed; **Section 6** presents the evaluation, its results, and describe the communications of the research; and finally **Section 7** presents conclusions and point out some limitations and future work.

2 Background

This section presents a general overview of the knowledge base that is available in the field of this investigation, which compiles the main references that support the problem definition and the proposed solution. Therefore, in this section, we present the main concepts, frameworks, methods e definitions strongly recognized and used in the industry and academy related to the EAM and MSA.

2.1 Enterprise Architecture Management

Enterprise Architecture (EA) can be defined as a coherent set of principles, methods, and models that are used to design the organizational structure, containing business processes, information systems, and IT infrastructure that align business and IT initiatives with the strategic objectives and drivers of an organization. It usually deals with the overview and not with details and seeks to understand and communicate how each of the main business and IT components of a company work with the others (The Open Group, 2018).

The Zachman Framework is a structural ontological model to describe the organization, serving as the basis for the main frameworks of EA. Although already quite old, it provides a logical framework for classifying and organizing the descriptive representations of an organization that are significant for enterprise management and for the development of enterprise systems in a graphical structure that describes a design artifacts that constitute an intersection between different perspectives and about the construction of a complex product and its abstractions. This classification allows us to focus on selected aspects of an architectural object without losing the holistic view. For example, in architectural perspective may be depicted the inventory representation, whereas in engineer perspective may show an inventory specification. The same logic of representation views may be applied to responsibility, motivations and others (Zachman, 2016).

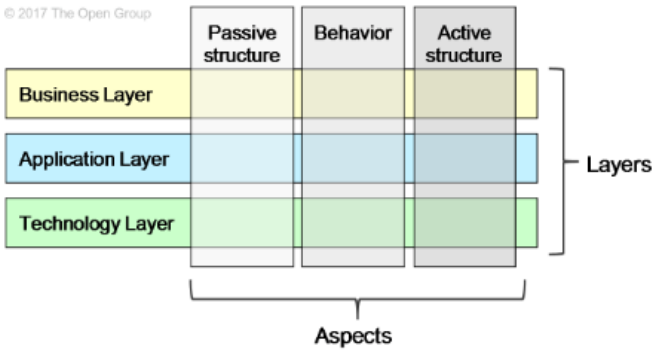
The TOGAF is a tool that allows the integration, production, use and maintenance of enterprise architectures. Maintained by The Open Group, it is very useful for describing enterprise systems, their components and their interrelationships, guiding the design,

implementation and evolution of EA over time. It defines a method of constructing enterprise architectures by identifying relevant building blocks represented in four different domains:

- **Business Architecture**: Describes business strategy, governance, organization and key business processes.
- **Data Architecture**. Describes the structure of an organization's logical and physical data assets and data management resources.
- **Application Architecture**. Provides a blueprint for applications to be deployed, their interactions, and their relations to the core business processes of the organization.
- **Technology Architecture**. Describes the logical software and hardware capabilities that are required to support the deployment of business, data, and application services, including IT infrastructure, middleware, networks, communications, processing, and standards.

The TOGAF framework defines a method to build EA, but this framework does not constitute a language, nor does it provide a notation for these building blocks. This function is fulfilled by ArchiMate, which serves as the notation bases for the description of the architectural components and contributes in the definition of the way to present the different architectural viewpoints (The Open Group, 2017). The ArchiMate core framework is shown in Figure 2.1.

Figure 2.1 - ArchiMate Core Framework



Source (The Open Group, 2017)

The figure above shows how the ArchiMate core structure allows display different viewpoints, where the position within the cells highlights concerns of the stakeholders in two dimensions.

- **Layers dimension**, which represents levels of modeling. In the core framework are addressed three levels:
  - **Business**: depicts business services offered to customers;
  - **Application**: depicts applications services and applications components that support the business,
  - **Technology**: depicts technology services needed to run the applications.
- **Aspects dimension**, that show the active structure aspects which represents structural elements, the behavior aspects which represents the behavior performed by the actors and the passive structure, which represents the object on which is performed.

Enterprise Architecture Management (EAM) is a discipline for managing increasingly complex Business/IT relationships in organizations. It is related to coordinating the architectural development across levels and departments in an organization, coordinating people to achieve and keep IS efficiency and effectiveness realized through aspects such as modeling, planning, principles and governance structures (Weiss et al., 2013). It includes the construction of EA models within large enterprises and complex information system landscapes containing a plenty of data sources with relevant information for a complete EA description. These data sources may include business process models, software documentation, service documentation, configuration management databases, such as configuration items in the IT infrastructure. (Chen, Hess, Langermeier, Stuelpnagel, & Diefenthaler, 2013).

2.2 Microservice Architecture

Microservices are stand-alone services, which can be deployed independently, which run in an individual process and communicate through light technologies, REST for example. They are built to provide business capabilities referenced as bounded context. The fundamental aspects of the MSA is that it refers to an architectural style that aims to build an application as a set of microservices that combines several elements of DevOps in order

to allow the microservices to be easily and independently deployed and scaled (Fowler & Lewis, 2014; Newman, 2015). The goal of MSA after all is to prepare the organization for agile changes. Therefore, it is important to embrace an adaptable governance in a simple and fast way that enables to track the components modeled in the EA for information systems based on MSA with all restrictions and recommendations from the enterprise level.

MSA is an implementation approach to SOA, sometimes qualified as "SOA done right" (O. Zimmermann, 2017) for distributed systems based on microservices that target scalability adding constraints to ensure the independence of services (Yale Yu et al., 2016). Balakrushnan et al. (2016) reinforces these definitions by stating that MSA aims to create systems through the use of small, independent and self-contained services strongly aligned with business activities. He also lists and describes some of the key characteristics of MSA, such as described in Table 2.1.

Table 2.1 - Characteristics of MSA

Characteristic	Description
Single Responsibility	A microservice is directly aligned with a service for a single business activity. A business activity is described by TOGAF (Balakrushnan et al., 2016) as being a unit of work performed by the organization that supports a business process or function. Thus, the service exposed by the microservice is mapped to a specific business activity fulfilling all the necessary requirements to complete the activity.
Self-Contained	A microservice must contain all the resources necessary for its execution in order to support the business activity. It means that the microservice does not share any IT resources.
Fully Decoupled	Microservices should have as much decoupling as possible. To achieve this level of decoupling, a business function must be decomposed to the level where a microservice is implemented to serve an atomic business function.
Decentralized Governance	The decentralized governance consists of the idea that a single team is responsible for autonomously managing the entire microservice life cycle, including data governance.

Scalability	Multiple instances of the microservice can be created automatically in parallel, thus allowing to increase or decrease the number of instances according to demand, optimizing the use of infrastructure and also contributing to the resilience of the microservice.

2.3 Microservice Architecture and Service Oriented Architecture

SOA is an architectural style that combines some architectural principles looking for low coupling, high reuse and interoperability among services. Microservice architecture, in turn, is a style of architecture for distributed systems based on microservices that target scalability and is one of the possible implementations of SOA (Yale Yu et al., 2016). The MSA emerged a short time ago, while SOA is a longer-term subject with a greater wealth of publications, therefore MSA is an SOA derivation. This diagnosis is useful because we can observe the lessons learned from SOA and the implications of adopting the MSA. It is noticeable that most of the time when reading about SOA, the authors are referring to the most common SOA implementation, which is using a services orchestration through an enterprise service bus (ESB) pattern. On the other hand, Microservices Architecture (MSA) generally adopts the Event Driven Architecture (EDA) pattern of choreography in contrast to the ESB pattern of service orchestration. The paper by Yale (Yale Yu et al., 2016), helps us to refine the understanding between these architectural styles. We also aggregated some contributions from The Open Group ("The SOA Source Book - Microservices Architecture," 2016). Both the paper and these contributions point out contrasts between SOA and MSA. An SOA service has the following characteristics:

- It is based on service design, being a logical representation of a repeatable business activity with a specified result running an enterprise business process.
- It is self-contained.
- It can be composed of other services.
- It is transparent to consumers (encapsulation).
- They are implemented to a specific environment, and should be described within a context, so the service representation uses business descriptions to provide context and implements services through service orchestration.

- It uses open standards to achieve interoperability and transparency of localization
- It requires strong governance.

In an overview, the MSA is aligned with SOA, sharing plenty of characteristics, but focused on the autonomy of execution of each service. However, MSA extends the SOA principles, adding new constraints to ensure the independence of services, thus implying some differentiations that are described in Table 2.2.

Table 2.2 - Comparative between SOA and MSA

SOA	**MSA**
Services may be composed by other services.	A microservice is self-contained and totally independent from other microservice, therefore, a microservice cannot be composed by other microservices.
A business process as a service encapsulate many services to complete the task.	A business task is totally fulfilled by one microservice.
The services are exposed through an ESB, imply that to scale, whole ESB has to scale with the infrastructure that supports it.	The services are scalable independently.
Seeks decoupling between consumers applications and providers applications.	The service itself should be highly decoupled.
The infrastructure cost is stable	The infrastructure cost is volatile due to auto-scalability.
The governance is strong and centralized.	The governance is autonomous and decentralized.
A service can support multiple responsibilities, including a long-term process	A service has a single and unique responsibility

3 Related Work

This section presents the related work containing a literature review that shows recent researches related to EAM issues and solutions approaches over MSA with a brief summary of their contributions to this work.

3.1 Literature Review Process

To identify the state of art about the problem domain, which basically brings together two major themes, EAM and MSA, we started by doing an exploratory research on Google Scholar and ResearchGate to identify the candidate keywords to the searching of publications that touch both themes. From this exploratory research we selected the key words, shown in the square below, for deepening the systematic research:

("Enterprise Architecture Management" OR "Adaptive Enterprise Architecture" OR "Adaptable Enterprise Architecture") AND (Microservice* OR SOA).

With these keywords, we search publications on Google Scholar and on AIS e-Library. From the results we applied successive exclusion criteria aiming to eliminate publications out of the objectives or from predatory origins as detailed below.

1. In the first round, articles in languages other than English or Portuguese were excluded. Portuguese because it is the natural language of the author and country of the university where this work has been developed, and English because it is the universally recognized and accepted international language for academic papers.
2. In the second round of exclusion, as the objective of this paper is to identify relevant enterprise-level factors of microservice architecture, papers whose title indicated technical focus, infrastructure bias or out of this objective were excluded.

3. In the third round with the articles that remained, we sought to eliminate publications from predatory origins through the sites *Beall´s List of Predatory Journals and Publishers*[1] and Scimago *Journal & Country Rank*[2].
4. At the end, a more careful evaluation of the abstracts, of the Introductions and of the conclusions of the articles was carried out in order to perceive the relevance of the contributions to this work, excluding those that did not seem to contribute significantly to this research.

A quantitative view of the process followed the research of the related work resulted in the numbers shown in Table 3.1.

Table 3.1 - Quantitative analysis of systematic literature review research

Step/Exclusion Criteria	**ResearchGate**	**Google Scholar**	**AIS**
Total of publications found	70	382	75
Language different from English or Portuguese and citation less than 2 and rank less than 100 and year before 2016 or titles out of subject.	0	197	56
Title indicating focus too technical, on infrastructure or view out of objective.	0	99	48
From Predatory Publishers	0	91	0
Abstract, Introduction and conclusion relevance.	5	25	9

After reading the content of the 39 publications and 3 books and two web sites [3,4] with good reputation and referenced by most of the authors, 19 baseline contributions for the initial references of this dissertation were selected. Later we also aggregated other papers published during the development of this research, papers which are described throughout this chapter.

[1] https://beallslist.weebly.com/
[2] https://www.scimagojr.com/index.php
[3] https://martinfowler.com/articles/microservices.html
[4] https://publications.opengroup.org/white-papers/togaf

3.1.1 Research´s Relevance

Di Francesco et al., (2017) reinforce the importance of this research, as they give a notion of the state of art with a systematic survey of trends, focus, and potential of scientific research in microservices, pointing out some characteristics of the microservices architecture style. For instance, smart endpoints and decentralized control of languages and data that poses some challenges to the enterprise. They show from the survey that the focus of the researchers is on the proposal of solutions at the technical and infrastructure level. This happens because the architectural style of microservices is still in its infancy, despite the great interest in it. Their paper serves here to reinforce the importance and interest in the MSA. In his survey, they separate the researches into two groups, introvert and extrovert, where introvert is concerned with internal aspects focusing on design-oriented activities of the microservice, whereas extrovert is concerned with aspects related to the external relationships of communication between architects and other stakeholders. It is about the extrovert aspects that we are concerned about in this work.

3.2 The need for a language for describing microservice architectures

Di Francesco (2017) investigated the key aspects of microservice architectures, and provided a comprehensive understanding of MSA style and its implications in some challenges. He pointed out an absence of an architectural language for describing microservice architectures, the need to understand what qualities that MSA can satisfy, globally and locally. He also indicated challenges in the definition of appropriate instruments for supporting the design of microservice architectures, as well as identifying factors that impact architectural decisions.

Chen et al. (2013) researched semantic architectural descriptions which should allow to integrate data sets from various relevant information sources within enterprises architecture repository. According to them, keeping the EA up to date is a time consuming and difficult task. They identified a structural conflict caused when semantically equivalent information is represented in a different way that hampers automation. This reinforces the importance of trying to describe EA through a standard language.

3.3 Decentralized Governance

Yale et al., (2016) reinforce that microservices are part of an enterprise landscape and affirm that the duplication of functions and microservices is favored by the lack of clarity in microservice scope and ownership definitions, which implies multiplying costs for all the business areas of the organization. They indicate that the autonomous choice of the technology, even though beneficial from the developer's perspective, can be problematic from the enterprise standpoint, exemplifying that tracking and managing licensing arrangements throughout this set may become impossible at the enterprise level, not being economically efficient if managed only by each team individually.

Balakrushnan et al (2016) contributes further, by demonstrating the scopes of enterprise architecture governance concerns in describing the microservice governance contexts in a federalized enterprise architecture, pointing to aspects such as:

- Security policies
- Legal policies
- Flexible technology standards
- Enterprise standards
- Knowledge management

Newman (2015) and Fowler & Lewis (2014) advocated that in microservice architecture, microservices must be product-oriented rather than project-oriented. This perspective expands the impact of decentralized governance, implying that the organizational structure of the teams of development, operation, and architecture governance should be planned with a product vision.

Buchgeher et al. (2017) reported that even with the decentralized governance, decisions concerning more than one microservice can be made using a shared governance model, where decisions are made together by members of different teams. They observed deviations from the Decentralized Governance principle in larger companies, such as Spotify, Soundcloud, and PegahTech Co. which restrict decision-making around service-team selection of implementation technologies. To them, technology selection for the microservice infrastructure is complex due to the large number of different implementation technologies available.

Engel et al. (2018) pointed out that experiences in the projects have shown that changes made to one microservice may not be limited to this microservice. Thus the impact of changes shoud be analysed in EAM. Also their experiences have shown that homogeneity of tecnology is advised for better maintainability, robustness and understandability of the overall system.

Knoche & Hasselbring (2019) pointed out that a big part of teams may have difficulties to take care of numerous cross-cutting concerns that were previously done by specialized teams, such as security, data protection and system monitoring. Another issue is related to the autonomous choice of tools by the teams when companies must ensure enough licensing and support contracts. Also, the organizational implications of adopting microservices may further be incompatible with compliance and regulations.

3.4 Scope definition of microservices

The microservice development must be closely aligned with the design of the organization's business process. The scope and responsibility of the microservice (bounded context) is obtained through the mapping and decomposition of the processes and business domains (Fowler & Lewis, 2014; Newman, 2015).

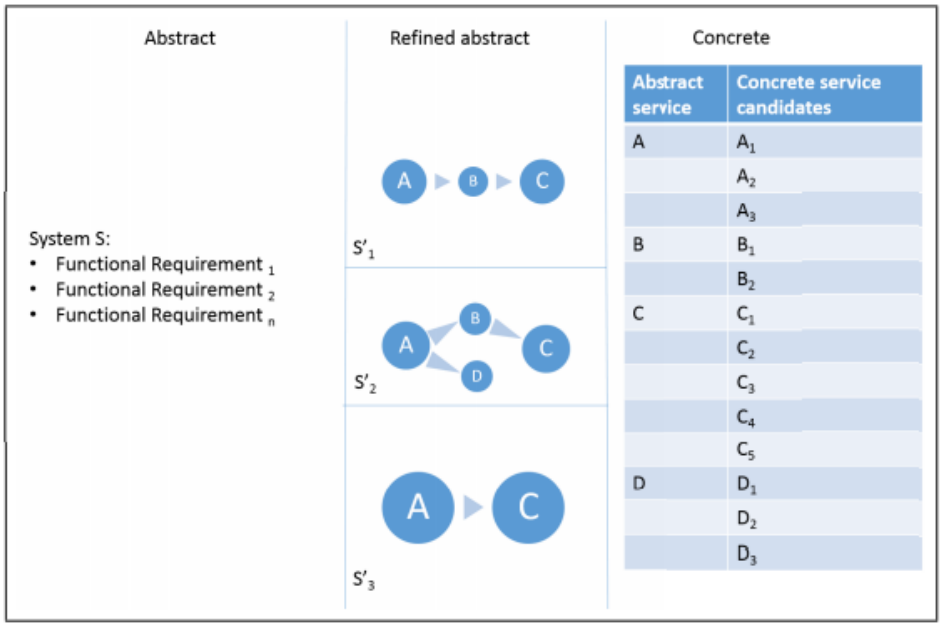
To Engel et al. (2018) the major challenge in MSA is related to a contextualized definition of microservices while keeping an overview of the systems and the effects of non-local changes.

Balalaie (2016) studied an experience of migration of an on-premise application to cloud-based microservices architecture reporting the lessons learned from this migration. They reported that as the size of the system starts growing a long-term commitment to a technology stack start to appear. This reinforce that control the technology stack at EAM level might be beneficial.

Hassan & Bahsoon (2016) investigated the impact of microservice granularity and analyzed the trade-offs from microservice sizing to reflect on the foundational context of this paradigm in order to address design problems. They mapped levels of abstraction that help to define the best granularity decomposing these levels from the most abstract to the most concrete. Figure 3.1 shows these levels, where in the first level it is abstract service which allows focus on functionalities instead of on the implementation, which is realized at

the concrete service level. Thus, it provides an abstract scope to the concrete service from a high level of abstraction.

Figure 3.1 - Different conceptual modelling levels for microservices



Source (Hassan & Bahsoon, 2016)

Soldani et al. (2018) explore pains and gains of microservices and show that at the design time the principal pain is related to the difficulty in determining the right microservice´s scope. All such studies agree that it is often difficult to identify the business capability or bounded context that should be assigned to each microservice, as the boundaries among different capabilities/contexts are usually not sharp.

3.5 Knowledge management

Buchgeher et al. (2017) pointed out that microservices currently have a high learning curve. Neither the concepts nor the principles were perceived as difficult, but their technical realization in learning and using dedicated frameworks and technologies was time-intensive and thus costly. Also, suggested that the free selection of implementation technologies may result in a loss of technology-specific knowledgement due to the small size of the development team.

Knoche & Hasselbring (2019) conducted a survey on drivers and barriers to microservice adoption among professionals in Germany's software industry. They provided relevant views over some important barriers linked to EAM. One barrier is related to insufficient skillsets of service developers and operators and observed that notions like polyglot programming and persistence require developers to be proficient in multiple programming languages and persistence solutions.

3.6 Controlling costs

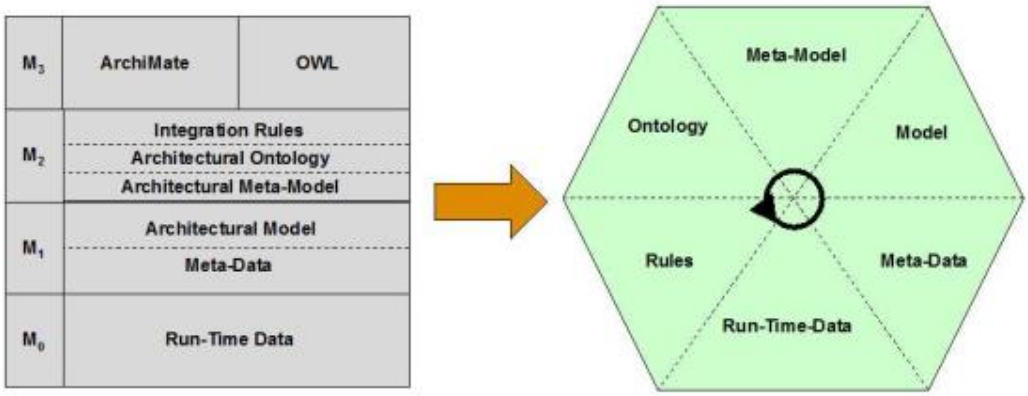
Singleton (2016) analyzed the costs of microservices and the trade-off between promised benefits and their costs. He noted that a microservices architecture requires extra machinery to publish, process, communicate among services, route to services, handle messages and queuing services, and to deploy and monitor services. These can all impose additional costs. On the other side, Knoche & Hasselbring (2019) contributed indicating that for microservice architectures, software components which are licensed per running instance, may be particularly costly.

3.7 EA-Mini-Descriptions

Although each microservice has low complexity, the complexity of the system is not reduced, but moved from internal to external architecture (Bogner & Zimmermann, 2016a, 2016b). In this context, integrating many architectural descriptions of a different variety of microservices into a consistent EA description is a challenge. Especially considering the view that the complexity of EAM can complicate the documentation and not guarantee the quality of information (Wißotzki, Wismar, & Sonnenberger, 2013). Thus, the properties of microservices require advanced EA methodologies for the integration of many micro-granular architectures within an effective and adaptive EA. In this way they proposed a model of EA-Mini-Descriptions which serves as a meta-model for microservices that can be combined into a larger model, unifying microservices into a holistic dynamically-adjustable reference architecture (Bogner & Zimmermann, 2016a, 2016b). These architectural mini descriptions are displayed in Figure 3.2, where:

- M0 and M1 capture local calls to a microservice. M0 represents operational aspects of run-time or monitoring data and M1 provides metadata such as purpose, API endpoints or monetization of the use, as well as its internal architectural model, components and communication channels.
- M2 serves as a global meta-model with information for communication between microservices, providing a view of the general properties of microservices with rules of integration.
- Finally, the M3 layer defines semantic representations and languages used to model and represent these meta-models of adaptive EA.

Figure 3.2 - Structure of EA-Mini-Descriptions



Source (Bogner & Zimmermann, 2016a)

In another related paper Zimmermann et al. (2017) emphasizes that it is important to provide viewpoints with stakeholder-relevant information within an architectural description and that organizational and semantic integration is a challenge, since that semantics in conflict with homonyms and synonyms creates misunderstandings among multiple process participants and causes weaknesses leading to organizational inefficiencies. The quick changes in the transformation process of organizations requires a high flexibility to enable the organization to quickly adapt to changes in the business environment and seize new opportunities by requiring the ability to develop and publish services in ever faster production.

Zimmermann et al. (2018) also investigate evolution approaches, models and mechanisms to support the digital transformation by applying an enterprise architecture model introducing new viewpoints combined with a service perspective of distributed

systems which defines the structure of components, their inter-relationships, together with principles and guidelines for governing their design and evolution. They identified the need for a bottom-up integration of a huge amount of dynamically growing micro-granular systems and services, which includes and refines the previously EA-Mini-Description proposal. In addition to some insights that reinforce the domain of the problem of this research, their paper contributes to the detailing of EA-Mini-Descriptions in a higher level, proposing the DEA-Mini-Models (Digital Enterprise Architecture Mini-Model) which is a mini-model that responds to the flexibility required to cope with the diversity and distribution present in the MSA.

3.8 Extreme Enterprise Architecture Planning

In addition to what was proposed by Bogner e Zimmermann (Bogner & Zimmermann, 2016b) for microservice modeling described in the previous section, we firstly intended to compose at a higher level of EA modeling with the model proposed by Ramos & Vasconcelos (2014) , which aims to provides an agile approach to EA planning in order to increase the capacity to respond to changes quickly. If the goal of MSA after all is to allow for agile changes, it is necessary an adaptable governance, to apply a simple and fast model, and able to keep track of the components modeled in EA for the information systems based on microservices with all the restrictions at the enterprise level. Therefore, this model was used as reference to assemble the higher-level architectural framework to EA, complementing the mini-descriptions of Bogner and Zimmermann, which seems to not consistently address the higher-level view of systems architecture based on MSA. However, the final result does not have a direct map, resting from it the initial insights of some artifacts.

3.9 Enterprise Communication

Lenarduzzi & Sievi-Korte (2018) indicate some negative impacts on enterprise communication regarding the adoption of microservices due to the nature of decentralized governance, which reduces the need for communication and argues that this aspect creates a problem in the synchronization of the projects or the strategies of the enterprise as a whole, correlating it to the same domain of issues observed in the failures to adopt GDS (Global

Software Development), in which independent development teams developed different parts of the same system. The authors surveyed pros and cons, thus allowing a balanced analysis between the benefit obtained in each possible team structure and the negative impacts on enterprise communication. So, the choice of this team structure and how the microservice team communicates should be analyzed at the EA level.

3.10 Bimodal Enterprise Architecture

Drews et al., (2017) describe the new role of EAM in the Bimodal Enterprise Architecture. In order to deliver services to consumers faster and faster, many companies are embracing a hybrid model where an agile team structure is created to develop and operate new products by adopting agile and DevOps practices, usually built using microservices (Fast IT), and another team structure is accountable for maintaining and operating the legacy products (Heavy IT). Despite the high autonomy of the teams in this new Fast IT environment, EAM still needs to support teams on cross issues of services / microservices, but playing a more consultative role than in traditional IT, with a focus on making recommendations instead of allowing or disallowing certain architectural decisions while still supporting cross-microservice architecture development, keeping track of permanent changes in IT architecture and providing information to enable cost transparency for microservice teams.

3.11 Microservice Architecture Monitoring

Engel et al. (2018) pointed out that experiences have shown that changes made to one microservice may not be limited to this microservice. Keeping an overview of the system is a big challenge in the projects, especially when large microservice-based systems are developed using agile development methods within different autonomous teams. Their experiences in the projects shows that the complexity of tracking of the system increases, since it becomes hard to statiscally analyse the dataflow and services dependencies like in monolithic systems.

For Haselbock & Weinreich (2017) the selection of a monitoring system is an essential part of each microservice architecture due to the high level of dynamic structure and behavior

of such a system, thus they presented a decision guidance model for microservice-based systems for generating of monitoring data and providing monitoring information to stakeholders. We used their model as a reference to complement the runtime information to the EA-Mini-Description in order to provide relevant runtime information, as well as keeping the EA model up-to-date with this information.

Buchgeher et al. (2017) investigated the impact of microservice principles to small and medium size companies (SME). They reinforced that microservices foster evolutionary design. Since they are independently deployable, can evolve independently. Another characteristic is that microservices require a comprehensive monitoring infrastructure that permits observation of the running system and all its constituent services and their interactions.

3.12 Research Discussion

Microservice has aroused a great interest recently, although it is still in its academic childhood, which is a reason for most of the scholarly papers to be characterized by questions related to technical and infrastructure solutions that Di Francesco classified as introvert aspects (Francesco et al., 2017). This reinforces the importance of investigating external characteristics related to the communication among architects and other stakeholders at the enterprise level.

We understand MSA as an extension of SOA, applicable to distributed information systems, where it is sought to develop a mesh of autonomous, independently deployable and scalable microservices (Fowler & Lewis, 2014; Balakrushnan et al., 2016; Yale Yu et al., 2016; O. Zimmermann, 2017). Information systems are covered by the Zachman and TOGAF frameworks, while SOA is already a well-discussed topic in the academic and market world. However, it is still unclear how to address these technologically heterogeneous components in order to provide relevant viewpoints for enterprise stakeholders (Bogner & Zimmermann, 2016a; Wißotzki et al., 2013).

In EAM governance of microservices we have seen challenges to clearly identify factors that impact architectural decisions and define appropriate instruments to support the design of microservices architecture (Di Francesco, 2017). If not well managed, Microservices architecture can impose additional costs to publish, process, communicate,

route and monitor (Singleton, 2016). On the other hand, companies must ensure enough licensing and support contracts and, for microservices in clouds, some architectural components are charged by running instances that can be cost-consuming (Knoche & Hasselbring, 2019), which reinforces the need to a comprehensive monitoring infrastructure in order to observe the running system in a way that should allow drilling down through the architectural components to an individual microservice (Buchgeher et al., 2017). We believe that more than allow investigating issues, this monitoring may help to predict and mitigate enterprise risks, control costs, plan better the business capability, and innovations.

The company structure must support the vision of product-orientation and a decentralized governance (Fowler & Lewis, 2014; Newman, 2015), however we observe that the decentralized governance reduces the need for communication, and this may create a problem in the synchronization of strategies of the enterprise as a whole (Lenarduzzi & Sievi-Korte, 2018). Thus, the choice of these teams' structures and how the microservice team communicates should be analyzed and planned by EAM.

From the other perspective, notions like polyglot programming and persistence require developers to be proficient in multiple programming languages and persistence solutions. A big part of teams may have difficulties to take care of numerous cross-cutting concerns previously done by specialized people or groups, such as security, data protection, monitoring, database administration, backups and others which may imply in insufficient skillsets of microservices teams (Knoche & Hasselbring, 2019). An EAM model that enables the company to control the diversity of most stack of technologies would reduce the risk of dealing with insufficient skillsets. Moreover, it would record and facilitate to handle cross-cutting concerns.

Defining the scope of microservice is hard due to the difficulty in identifying boundaries among different business capabilities or contexts (Engel et al., 2018; Soldani et al., 2018). The lack of scope clarity and difficulties do define the ownership favors duplication of functions and services, multiplying costs throughout the company (Salah, Jamal Zemerly, Chan Yeob Yeun, Al-Qutayri, & Al-Hammadi, 2016). Therefore, it is important, at least in high levels, to address the boundary and scope definition of the microservice-based system in the EAM model.

One of the major challenges is maintaining the overview of the whole system. Engel et al.(2018) advise pursueing homogeneity in order to favor the understandability of the

overall system, as well as its maintainability and robustness. For us, this challenge is also related to the absence of an architectural language to describe microservice architecture and the understanding of which qualities MSA can satisfy both globally and locally (Di Francesco, 2017). Thus, once architectural requirements are identified, EAM should be communicated, providing a holistic view of the entire system and organization in a clear architectural language, which reinforces the importance pointed by (Chen et al., 2013) of describing the architecture according to industry standards and widely recognized language, such as ArchiMate.

It should be possible to integrate data sets from different sources within enterprise architecture repositories, however, keeping the EA model integrated and up-to-date is difficult, time-consuming and hard to automate due to structural conflicts caused by differences in representations of semantically-equivalent information (Chen et al., 2013).

On the other hand, the technology selection for microservices is complex due to the large number of technologies available to integrate (Buchgeher et al., 2017). In this context, in addition to choosing a language that enables better semantic integration, EAM can act as a facilitator of technology choices by providing a general catalog and a rational decision model in an architecture that enables rapid integration among the architectural views of microservices.

Targeting to address the complexity of the microservice-based system and enable to keep the EA model up-to-date, Zimmermann et al. (2018) proposed a model of architectural mini-descriptions, where a holistic and enterprise vision would be obtained from the composition of these mini-descriptions. However, they do not detail what aspects should be described in the architectural layers that have been proposed. Still, they provide a useful base for modeling these aspects through the capturing of the micro-granular key elements from microservices models to compose the EA Model.

We found the modeling frameworks that served as a basis for the architecture model designed in the development of this research, joining two approaches that seem to complement each other. Although the XEAP (Ramos & Vasconcelos, 2014), did not appear in the final result, it provided insights to a top down approach to drive the adoption plan and, on the other hand, the EA-Mini-Descriptions provided detailed decisions from bottom-up capturing microservices data to refine and keep the holistic EA view up-to-date.

Furthermore, we have the contribution of several authors with indications of other aspects or concerns of MSA to be handled at EAM that will be covered in these meta-models. In Table 3.2 we summarize the most important microservice characteristics to EAM.

Table 3.2 - Main characteristics of MSA related to EA

Characteristic	**Description**
Fully Decoupled	Microservices should have as much decoupling as possible. Thus, the domain must be decomposed to the level where a microservice implements an atomic business function (Balakrushnan et al., 2016), thus we need an approach to help to define the microservice bounded context from a business view in the EA model.
Decentralized Governance	Decentralized governance consists of the idea that a single team is responsible for autonomously managing the entire microservice life cycle, including data governance (Balakrushnan et al., 2016; Fowler & Lewis, 2014; Newman, 2015). However, governance in EA level is still needed, however being not intrusive.
Scalability	Multiple instances of the microservice can be created automatically in parallel, thus allowing to increase or decrease the number of instances according to demand (Balakrushnan et al., 2016; Yale Yu et al., 2016). It Implies that infrastructure costs will be volatile, and these costs should be monitored and controlled.
Well-Defined Interface API	A microservice exposes a well-defined communication interface (API) with a published contract, which is exposed through an API gateway or proxy (Balakrushnan et al., 2016; Yale Yu et al., 2016). As the API gateway is a cross component it must be governed at enterprise level.
Product Orientation	In microservice architecture, microservices must be product-oriented rather than project-oriented (Fowler & Lewis, 2014; Newman, 2015). This perspective expands the impact of decentralized governance, implying that the organizational structure to support the development, operation, and architecture governance should be planned with vision focusing on the product.

In the EA context, information systems are already covered by TOGAF and ArchiMate. However, it is still unclear how to address enterprise architectural concerns about MSA in an ArchiMate Model in order to provide relevant viewpoints for enterprise stakeholders through a standard language. EA is also widely covered in SOA context, however the implications over microservice constraints require new views to accommodate the challenge of driving MSA implementation without blocking innovations.

Decentralized governance is an important concept in the microservice architecture, but at the same time it is equally important to maintain strategic alignment, optimize investments and control costs in IT. Therefore, for each relevant aspect the real benefits of delegating

certain decisions to the microservice teams or restricting such choices in EA must be scrutinized and mapped into an EA model.

The Open Group already developed a Microservice Reference Architecture ("The SOA Source Book - Microservices Architecture," 2016), but at a high level, not presented in ArchiMate and considering only a green field scenario where all microservice aspects can be managed without concerns about legacy. This paper, however, focuses on presenting an MSA Reference Architecture for companies that are highly regulated and will probably adopt this architecture in a brown field scenario, where it will be managed in a context of strong corporate culture, legacy structures and standards that must be followed, for instance bank and insurance companies. This work propose extend this reference by aggregating three key elements: an approach to model EA views of MSA based on TOGAF (The Open Group, 2018); a deep MSA view composed by micro-granular descriptions with a way to keep the EA up-to-date (A. Zimmermann et al., 2018); and a reference architecture modeled in ArchiMate standard notation (The Open Group, 2017).

Finally, Drews et al. (2017) clarified that in a bi-modal IT environment, common when adopting MSA, EAM continue important. However, it should play a more consultative role instead of restricting architectural decisions. In this role, EAM should act supporting microservice teams to ensure synchronization of business strategies and needs, while supporting innovation by providing accelerators for global and local decision-making about technology stack and ensures a holistic, strategic and integrated view of the entire system, enabling transparent control of costs and risks, and tracking and documenting changes in enterprise architecture.

4 Proposal

This section describes the solution proposal to address the key aspects of the MSA in order to permit EAM to govern the MSA adoption and guarantee the enterprise alignment of this adoption. We present here the implications of key aspects on EAM and a set of models addressing these aspects.

The solution is based on the joining of two approaches that appear to complement each other. A top-down view based on TOGAF to drive the MSA adoption plan, and the bottom up EA-Mini-Descriptions (Bogner & Zimmermann, 2016a) complementing with more details that enable capturing some decisions to update the holistic view of EA from the microservices implementation.

4.1 The microservices key aspects implications

In the scope of this work, some aspects that could matter to the EAM view were identified. These aspects, as well the solution proposal will be described ahead in this section. As previously presented , decentralized governance is an important point in the microservice architecture, at the same time it is also important to maintain organizational and strategic alignment by optimizing investments and controlling costs in IT. Then for each relevant aspect, the real benefits of delegating certain decisions to the microservice teams or restricting such choices in the enterprise architecture must be scrutinized. In this work we will not delve into the analysis of these choices but will assume that these aspects will be governed at the EA level and, thus, will be mapped into an architecture model using ArchiMate. We identified the key aspects through reading of related work and described in Section 3. These aspects are listed in Table 4.1 below and are discussed further ahead.

Table 4.1 - Microservices key aspects implications

Characteristics	Implications to EA model
Implementation-Agnostic	MSA allows each microservice in the system to be implemented with a different technology and language. Despite the benefits, the use of these technologies must be managed in order to avoid anarchy when effectively bringing some advantage to problem solving. Thus, the main technologies available in the company should be present in the EA Model so the microservice's team can select them. Also, their choice should be registered to allow for a better knowledge management of these technologies.
Fully Decoupled	Microservices should be totally independent from one another. For this, a microservice must implement an atomic business function, which should be identified by decomposing an enterprise business process.
Cloud-based	Usually, MSA uses cloud infrastructure that could be mapped in EAM in order to allow implement some global mechanism to monitor cloud usage and cost.
Well-Defined Interface with a Published Contract (API)	The microservice API should be represented in the EA Model, linking the microservice to other microservices and components to provide a holistic view of providers and consumers of services.
Decentralized governance	The model must represent the EA governance concerns and structures, as well as the microservices governance.
Business Domain	The business process and motivation should drive the definition of the microservice´s scope and the EA model will make this scope clear. Also, microservices should be represented in the EA Model, linked to a business function that they realize.
Organizational Structure	The microservice teams must be structured around the concepts of products. Also, the adoption of BuzDevOps practices require a view of how the teams are organized.

4.1.1 Fully Decoupled.

Microservices should have as much decoupling as possible to be independently deployable without impacting other microservices, consumers or other system´s components. To achieve this level of decoupling, the business function must be decomposed to the level where a microservice is implemented to serve an atomic business function (Balakrushnan et al., 2016). This function can eventually be used for many processes and serve multiple channels, thus in spite of it being decoupled, it should be interesting to keep the logical relationship dependence of these microservices with other external architectural components, in order to allow a better view of whole IT components and to promote a possible reuse. Sometimes, the reuse may become the management of these relationship complexes, with some effort to attend different needs. We consider that a microservice designed not to implement a specific need, but to implement a business domain need, will probably be reused as is, without new implementation or creeping its scope. Therefore, in this work we consider that the business function carried out by the microservice is a candidate to appear in the EA model.

4.1.2 Implementation-Agnostic

The microservice can be implemented across multiple platforms, languages and technologies, thus allowing the best choice to support microservice requirements to support business needs (Balakrushnan et al., 2016). However, from the enterprise's viewpoint, it is important to manage risks over business continuity, so they can protect themselves by contracting support from specialized companies in the technologies used by them. Of course, a new technology can be used, but it seems to not make sense for the cost management, to use a variety of technology to solve the same problem, so the company could get better agreements with a centralized catalog of the technology available for each context. On the other hand, using one or two technologies to solve the same need looks better for knowledge management, so the employee can be trained in the same technologies, thus they can migrate to other microservices team without a big technical effort, reducing the learning curve in the new team. Thus, the pool of possible technologies used to implement the microservice will be considered, like programing language and database technology.

4.1.3 Cloud-Based.

Usually microservices are published in a cloud infrastructure services (Newman, 2015), which implies in costs that must be monitored. Also sometimes companies have a corporate contact of the use of the cloud, so the use of the cloud infrastructure is also relevant for EAM, and the main information of the cloud usage and costs could be mapped in the EA model in order to allow implement some global mechanism, to monitor the cloud, which would be linked to the services contract.

4.1.4 Well-Defined Interface with a Published Contract (API).

A microservice exposes an API with a public contract, through which consumers communicate with the microservice (Balakrushnan et al., 2016). As a key integration element of the microservice with other services and information systems it seems natural to represent these APIs in EA with its connections.

4.1.5 Decentralized governance.

A single team has the ownership for the whole lifecycle of a microservice, so this team owns and autonomously governs every aspect of its microservices, including data (Balakrushnan et al., 2016). Newman (2015) reinforces the recommendation to decentralize all the things as possible. In this way, it is important not only to constantly look for opportunities to delegate decisions and control to the teams that own the microservice, but to recognize that sometimes overarching guidance is needed. For such cases, he suggests embracing a shared governance model.

As demonstrated by Balakrushnan et al., (2016) the scope of EA governance concerns is pointed to some aspects, listed below, which are useful to clarify what could be treated at EA to support a shared governance.

- Security policies
- Legal policies
- Flexible technology standards
- Enterprise standards
- Knowledge management

These policies and standards should be explicitly registered in the EA model.

4.1.6 Business Domain.

The microservice development must be closely aligned with the design of the organization's business process (Newman, 2015). Therefore, the scope and responsibility of the microservice (bounded context) should be driven by a business architecture that is modeled in the EA.

4.1.7 Organizational Structure.

The product orientation advocated by Newman (2015) and Fowler & Lewis (2014), imply a strong view of the organizational structure around the products instead of specialized functions or projects. This work assume that this structure should be planned at the EA level.

4.2 The Enterprise Architecture Model Proposal

In this section we will present the model solution containing a set of views addressing the aspect previously presented. A diagram containing the principles and the scope of EAM and microservice governance, a view of team structure to support the product development and operation, a model relating business process and application collaboration which helps to define an abstract scope for microservice-based systems, a model to keep a flexible catalog of standard technology solutions available, a model defining the responsibilities of EAM and the microservice owner team, and finally a proposed model to update the EA Model from microservices changes. At the end of the section, we show the feasibility of the model proposed checking its adherence with other architectural references and the gaps solved.

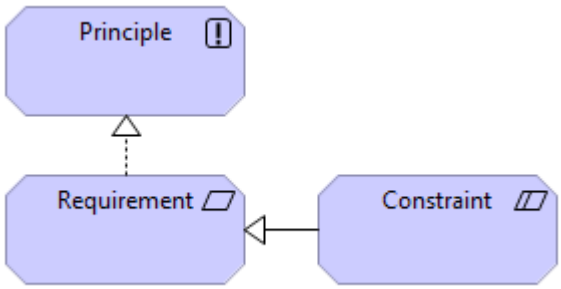
4.2.1 Defining the principles and governance scope

From the ArchiMate Specification (The Open Group, 2017) a principle represents a qualitative statement of intent that should be met by the architecture. It defines intended properties of systems and a general property that applies to any system in a certain context. Thus, principles are normative guidelines that guide the design of all possible solutions in a

given context. Requirements model the properties of these elements that are needed to achieve the “ends” that are modeled by the goals. In this respect, requirements represent the “means” to achieve goals. A constraint represents a factor that prevents or obstructs the achievement of goals and imposes a restriction on the way they may be achieved.

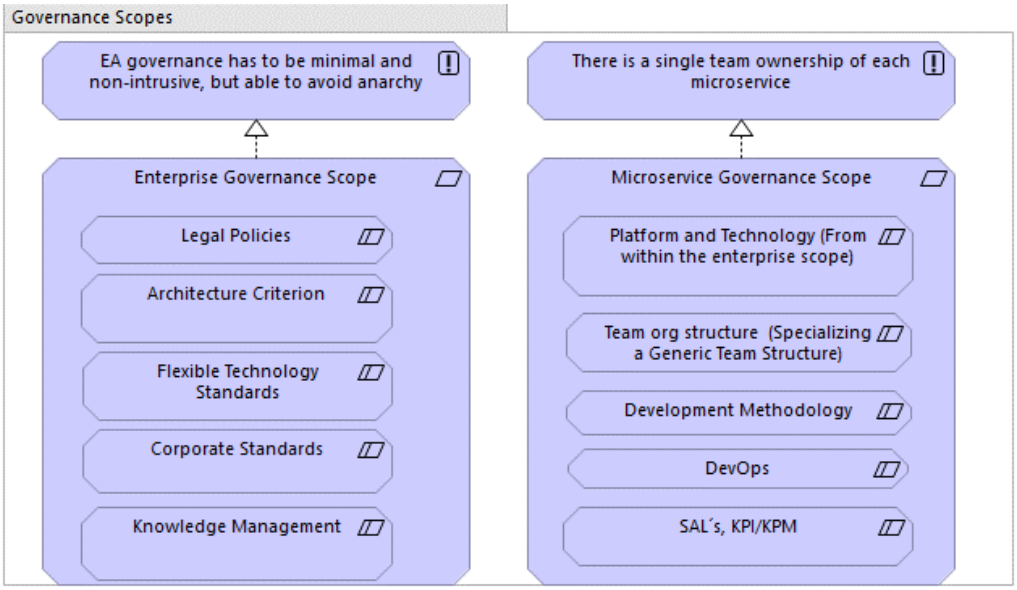
In this proposal we model the relationship expressed in Figure 4.1 to define the vision of the principles that will guide key decisions about MSA adoption governance, where principles define guidelines realized by requirements for scope governance which are specialized and expressed as constraints.

Figure 4.1 – Principle Metamodel



Based on The Open Group MSA Governance Framework (“The SOA Source Book - Microservices Architecture,” 2016), a diagram is proposed to clarify the concerns of EA and microservice governance scopes, is shown in Figure 4.2. It considers two main principles for each scope: the enterprise architecture governance must be kept minimal and non-intrusive; and a single team has full responsibility for a microservice. The idea is that any governance object that emerges should update this figure to visually guide what should be governed by EAM and what should not.

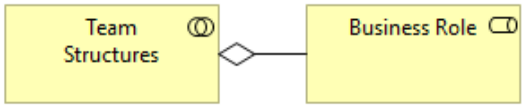
Figure 4.2 - Principles and Governance Scopes



4.2.2 The generic team structures

ArchiMate (The Open Group, 2017) defines business role as the responsibility for performing specific behavior, to which an actor can be assigned with specific responsibilities or skills. Thus, a business role is useful in a (structural) organizational sense; for instance, in the division of labor within an organization. On the other hand, a business collaboration is an aggregation of business internal active structure elements that work together to perform collective behavior. A collaboration is a collection of business roles or actors within an organization, which perform collaborative behavior. Therefore we basically choose to represent the relationship of the organizational structure based on the two elements described above, and shown in Figure 4.3, to mount the structural view of the teams that will support the governance of the identified aspects.

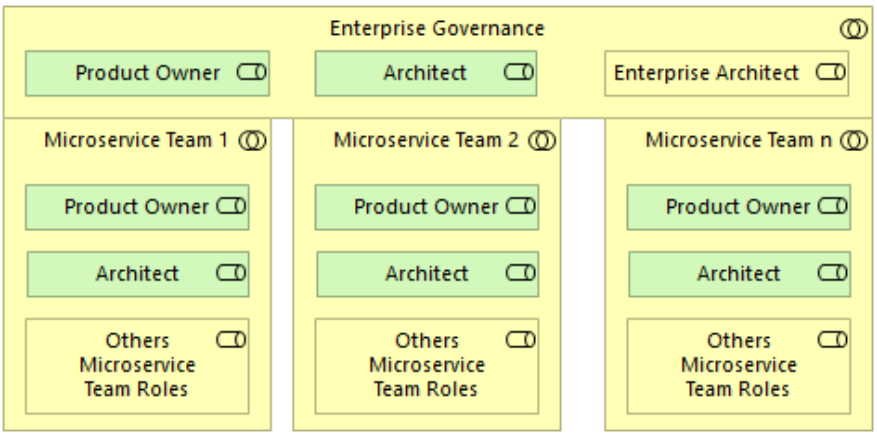
Figure 4.3 – Teams Structure Metamodel



As advocated by some references, microservice teams must be oriented to products, despite it being common for some companies to have their teams structured for projects or specialized IT functions (Fowler & Lewis, 2014; Newman, 2015). It is important to reinforce

that MSA requires an organizational structure similar to the one proposed in Figure 4.4. The main roles in this structure are the Enterprise Architect, the Product Owner and the Architect. We added the Enterprise Architect role to the structure suggested by The Open Group in order to define a specific role responsible to keep the EA models integrated and maintain the pool of EA requirements and recommendations to the microservice products. All these tree roles take part in the Enterprise Governance Board in order to keep the alignment between the EA needs and the Microservices implementation.

Figure 4.4 - Generic team structure for microservices



Adapted from (Balakrushnan et al. 2016)

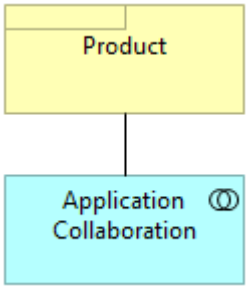
4.2.3 Business Processes and Product Applications.

A microservice is part of a system, and it may seem trivial to define a team to support the entire system. But, actually, it is not so easy, since the boundaries between the different business features of an application can be unclear. (Soldani et al., 2018).

From the ArchiMate definition (The Open Group, 2017), a product is a business element that represents a coherent collection of services and/or passive structure elements which is offered to internal or external customers. A product may aggregate or compose business services, application services, and technology services, business objects, data objects, and technology objects, as well as a contract. An application collaboration is an active structure element which represents an aggregation of applications components that work together to perform collective application behavior specifying which components cooperate to perform some task.

Thus, Figure 4.5 represents the ArchiMate elements used to link the identified products associated with an application collaboration of the subsystems involved. This collaboration will further be broken down into mini architectural descriptions.

Figure 4.5 - Product Collaboration Metamodel



Before modeling the products and their application collaborations we needed to identify the products, giving an initial scope to them. For this purpose, we propose a CRUD Matrix (Ramos & Vasconcelos, 2014)[5], shown in Figure 4.6, which helps to identify information clusters and the products, also distributing the microservices ownership based on these products, thus determining their proper scope for microservice systems, and helping do analyze the system's or product's impact. A trade-off from this view should observe the gains of scalability against the growth of complexity and costs when choosing the products architectural style.

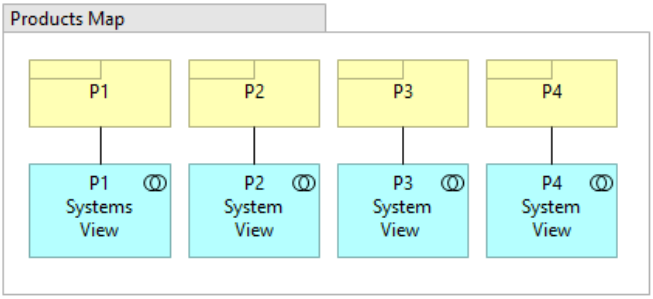
Figure 4.6 - CRUD Matrix to Determine the Products

CRUD Matrix		Entity								
		E1	E2	E3	E4	E5	E6	E7	E8	E9
Business Capability	C1	CU	CRU							
	C2		CRD	P1				R	R	
	C3			CRU	C					
	C4			CD	UR	P2		R		
	C5			R	R	CRU				
	C6						UD	CD	P3	
	C7			R					CRU	
	C8	R				R			CRD	D
	C9							P4		CRU

[5] CRUD matrices are widely used and old, so we cite recent work that provided us the idea, but the seminal work probably was (Martin, J. Information Engineer. Books I II and III, N.J., Englewood Cliffs: Prentice Hall, 1990[?])

As a result derived from the CRUD matrix above, Figure 4.7 shows the Product Map in ArchiMate notation, which in addition to providing a graphical view of product organization, helps to connect with the application collaborations that support these products.

Figure 4.7 – Products Map



4.2.4 Flexible technology standards.

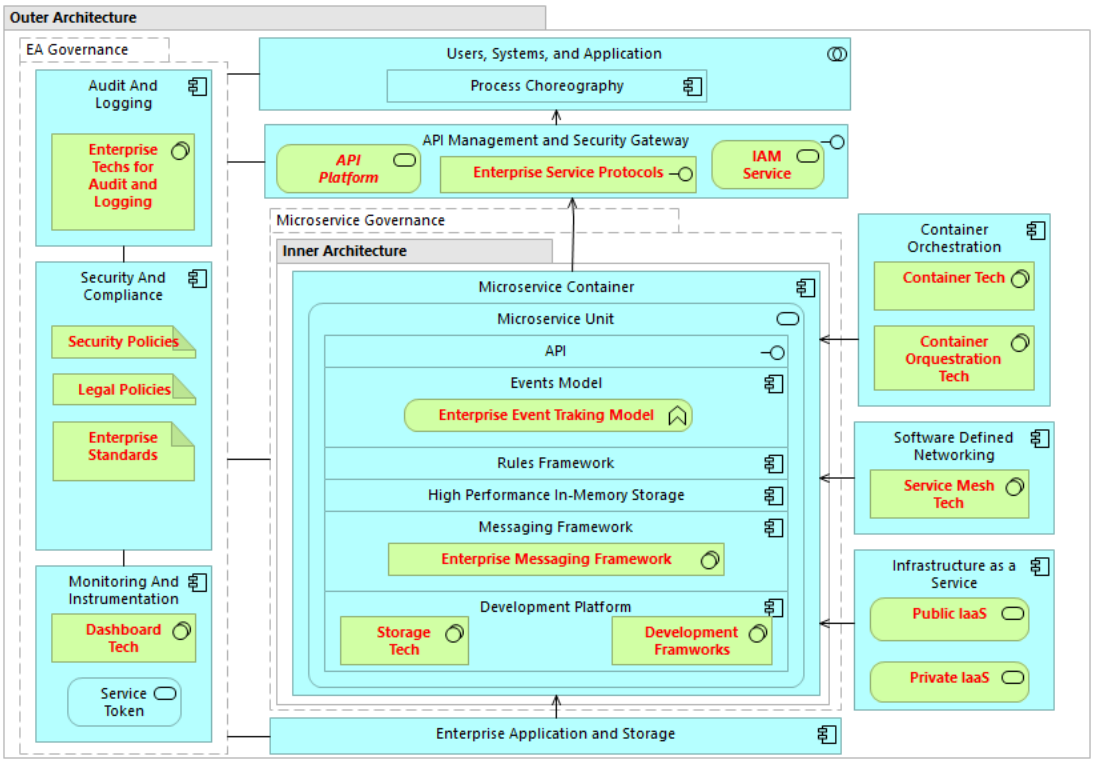
To describe the microservice architecture, it is important to perceive the separation between inner and outer architectures. This is especially relevant to realize if a concern resides internally in a microservice or at a shared-level architectural layer. To help the microservice team to choose the best technology to implement their needs, the EAM model provides a catalog of some important technologies, which considers their relevance when it comes to knowledge and cost managements. Figure 4.8 is an adaptation of the model proposed by The Open Group (“The SOA Source Book - Microservices Architecture,” 2016), but exemplifying, in green boxes, some artifacts that represent governance recommendations or requirements for microservices at the enterprise level. Thus, we represent the aspects such as enterprise security policies, legal policies, enterprise standards, architectural criteria, and available technology for each need (Balakrushnan et al., 2016). The enterprise scope could include container technologies, container orchestration and service. On the other hand, microservice containers usually will be published in a cloud which needs to be monitored in terms of cost. This model includes the available options from enterprise contracts.

Yale et al. (2016) provided some other aspects, such as an MSA should provide dashboards for instrumentation and must have the ability to monitor the services and the

instances dynamically created. They pointed out that an API platform should be used properly at enterprise level. The protocols supported by it also represent a technological restriction. A security gateway can be integrated with an IAM (Identity and Access Management) service to control access levels. Enterprise Event Tracking Model combined with an enterprise standard for an auditing and logging tool would serve as a central collection of all events, on which analytics can be performed.

It is important to note that everything inside of inner architecture is just a recommendation for the microservice team aiming to avoid the potential risks in having too many technologies, but without restricting the innovation itself. For instance, if Ruby on Rails and a Java Framework are largely used in the company, it can be a reason to recommend them, so it can benefit everyone with practices, standards and knowledge already disseminated in the corporation. These recommendations should contain criteria to support the choices.

Figure 4.8 - Microservice Enterprise Reference Architecture Restrictions



Adapted from ("The SOA Source Book - Microservices Architecture," 2016)

4.2.5 The responsibilities for architectural components.

Keeping in mind that it is desirable to delegate as many decisions as possible to the microservice team and that new technologies or business changes may give opportunities to review these responsibilities, we proposed the matrix in Figure 4.9, which defines the responsibilities of governance roles over each architectural property. In this matrix the lines represent the governance concerns identified in 4.2, and the columns, architectural components and their relations identified in section 4.2.4. The cells indicate if the principal responsibility resides in Enterprise IT Team Governance (ET), autonomously in the Microservice Team (MT), or in the Microservice team within enterprise Restrictions or Recommendation (MR).

Figure 4.9 - Governance Scope Matrix

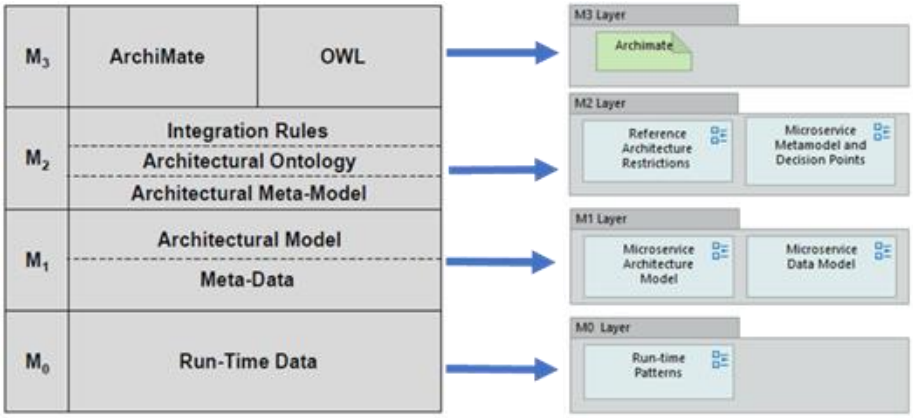
	Outer Architecture									Inner Architecture							
Governance Scope Matrix	**Users, Systems, and Application**	**API Management and Security Gateway**	**Monitoring And Instrumentation**	**Container Orchestration**	**Audit And Logging**	**Security And Compliance**	**Software Defined Networking**	**Enterprise Applications and Storage**	**Infrastructure as a Service**	**Microservice Container**	**Microservice Unit**	**Microservice API**	**Events Model**	**Rules Framework**	**High Performance In-Memory Storage**	**Messaging Framework**	**Development Platform**
Legal and Security Policies	ET	ET	ET		ET	ET		ET									
Architecture Criterion	ET	ET	ET	ET	ET	ET	ET	ET	ET	MT	MT	MT	MT	MT	MT	MT	MR
Technology Standards		ET	ET	ET	ET	ET	ET	ET	ET	MT	MT	MT	MT	MT	MT	MT	MT
Enterprise Standards	ET	ET	ET	ET	ET	ET	ET	ET	ET				MR			MR	MR
Knowledge Management										MT	MT	MT	MT	MT	MT	MT	MT
Platform and Technology										MR	MT	MT	MR	MT	MT	MR	MR
Team org structure											MR						
Development Methodology										MT	MT	MT	MT	MT	MT	MT	MT
DevOps										MT	MT	MT	MT	MT	MT	MT	MT
SLA´s, KPI/KPM										MT	MT	MT	MT	MT	MT	MT	MT

4.2.6 Keeping the Enterprise Architecture updated from microservice evolution

Once a team gets the ownership and the control over the governance of a microservice, a model is needed to help design each microservice in a language comprehensible for the whole organization in order to facilitate communication across teams, and at the same time a model that enables to integrate and maintain the EA model up to date from runtime data and capture any changes in the architecture of each single microservice. In this way, a model based on the EA-Mini-Description is used to define layers and design requirements in order

to capture the relevant information changes of microservices through the composition of these models. Figure 4.10 displays the model´s adaptation to an ArchiMate view.

Figure 4.10 - EA-Mini-Description



Adapted from (Bogner & Zimmermann, 2016b)

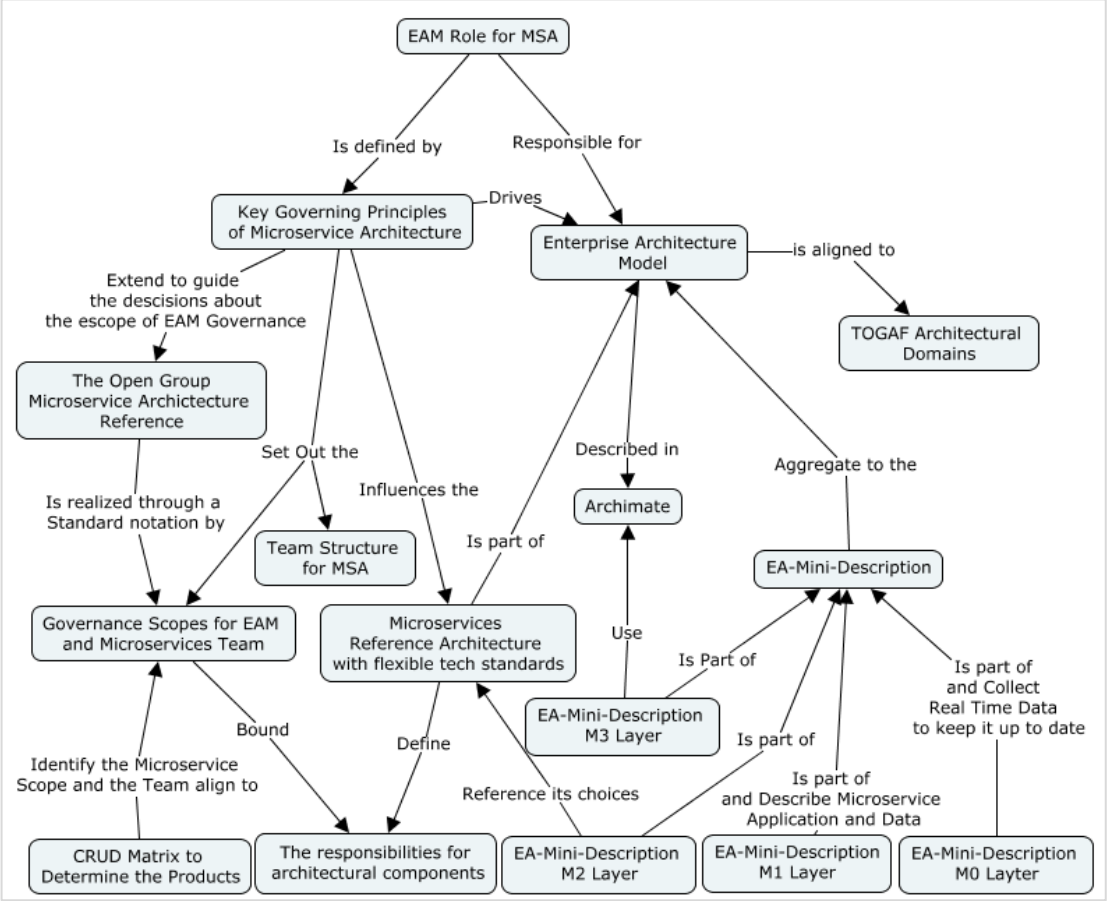
The EA-Mini-Description (Bogner & Zimmermann, 2016a) is used here to allow that evolutions in microservices automatically update the EA Model. Layer M3 describes what kind of knowledge is necessary to read the model. Here, we illustrated ArchiMate because it was used for modelling all views about the architecture proposed. However, other languages, like BPMN, UML or SysML can be used. It In layer M2 the Reference Architecture Restrictions are represented. They describe the main technologies and patterns that the microservice architecture is supposed to follow, and links to the same diagram defined in Figure 4.8 (Flexible technology standards). The M1 layer is a single microservice architecture view, which details the microservice enriched with information resulted from the microservice team choices over the options available in the EA and with more details about the implementation aspects, like rules, informational entities, event models, applications services, interfaces consumed, etc. M1 layer represents the metadata model view, containing properties such as API endpoint, usage costs or purpose.

4.3 Feasibility of the Proposed Model

We believe that the proposed model should support the pre-existing models while complementing them and keeping their utility. Thereby, we made a preliminary evaluation

to check its feasibility based on an argumentative evaluation that was built on the conceptual map shown in Figure 4.11 which summarize the main concepts covered in this work, targeting to evaluate the perspective of the feasibility of using the reference model proposed to enterprise architects in order to govern enterprise aspects of MSA. In this way, we checked the adherence of the proposal to the existing model in the literature and filled out some gaps when we joined the models to present a unified model covering all the aspects gathered.

Figure 4.11 - Conceptual Map of Solution



The role of EAM is defined by governing principles that drive the EA model, in this case, aligned with TOGAF architectural domains. The key governing principles of MSA extends The Open Group Microservice Architecture Reference, which requires a new organizational structure to support the view of product orientation. These principles are realized through delimiting the Governance scope for the EAM and microservice team. These scopes are bounded by the responsibilities for architectural components and the teams

which own each MSA product are identified by the CRUD Matrix to Determine the Products. The EA Model is described in ArchiMate and uses the EA-Mini-Description Layers to aggregate microservices information to the EA Model. Each EA-Mini-Description Layer describes a level of microservice data aiming to maintain the EA model up to date.

In Table 4.2, we evaluate the feasibility of the reference model proposed correlating it with The Open Group Microservice Architecture and with the EA-Mini-Description, which were used as architectural references baselines. Thus, we can realize exactly how we supported the pre-existing models in the unified model proposed and how we complement by filling out some gaps to cover all the aspects that the baseline models do not completely address.

Table 4.2 - Evaluation Map of Baseline Architectures Reference and Propose

The Open Group Microservice Architecture	
Key Governing Principles of Microservices Architecture	The proposal described in section 4.2.1 Defining the principles and governance scope is aligned with The Open Group Microservice Architecture reference ("The SOA Source Book - Microservices Architecture," 2016). However, we extend the key principles adding two others: *(i)* EA governance has to be minimal and non-intrusive, but able to avoid anarchy, and *(ii)* there is a single team ownership of each microservice as suggested for The Open Group in CASE STUDY: Rainyday Grocer. The objective with this extension is to establish a clear guiding principle to reinforce which concerns should be the focus of EAM or the microservice team, architecturally restricting the focus of each scope.
Governance Scope and Team Structure	The scopes of enterprise governance and microservice governance, as well the suggested team structure is almost the same as suggested by The Open Group expressed in "Microservices Architecture – CASE STUDY: Rainyday Grocer" ("The SOA Source Book - Microservices Architecture," 2016). As the open group reference is made by free drawings with no formal notations, we proposed the diagram described in the 4.2.1 defining the principles, the generic team structures, where we translated it into the ArchiMate notation in order to maintain the purpose of describing any relevant aspect in this notation and adding a role to define the responsibility to maintain the enterprise alignment and enterprise models. In addition, the section 4.2.5 the responsibility for architectural components describes clearly the responsibilities in the Governance Scope Matrix (Figure 4.9)
Microservices Reference Architecture	The components present in Flexible Technology Standards described in section 4.2.4 are completely aligned with the Microservices Reference Architecture presented by The Open Group , however we relate the respective EA governance artifacts in the same diagram for each component and discuss non-intrusive drivers and recommendations to support architectural decisions about technological choices to the

	inner architecture, such as database, message and development tools from one available in the enterprise catalog and to the outer architecture of the MSA such as the API gateway tools, enterprise security and other policies for instance.
EA-Mini-Descriptions	
M3 Layer	Although other notations for different views may be used in terms of EA Descriptions (Bogner & Zimmermann, 2016b), we use ArchiMate to accommodate the model proposed targeting to use a well-established and known standard notation in order to facilitates the communication among the teams, enterprise architects and other stakeholders.
M2 Layer	The Microservice Enterprise Reference Architecture Restrictions described in 4.2.4 the Flexible Technology Standards is incorporated in this level enabling to keep it up to date. Also, we suggested a view correlating the microservice metamodel with decision points to help choose the options available. It should follow some references such as the one proposed by Haselböck (Haselböck, Weinreich, & Buchgeher, 2017), however these elements are not deeply analyzed in this paper.
M1 Layer	In this layer should be described the specific meta data model of each unity of microservice, such as purposes, service API endpoint, usage, cost model and its inner architectural model containing its internal components. The interest of EA in this layer is to ensure that it is documented and easily found to allow a quick view on the microservices form the EA Model.
M0 Layer	The run-time patterns which allow monitoring operational and business relevant information about microservices should be described in this layer to allow collect run-time data such as cloud service or infrastructure usage and provide a real time tracking of costs, necessary to address the cost transparency.

None of these references clearly address the definition of microservices scope and the responsibility for the artifacts identified. Thus, to complement these gaps we proposed two other views. The CRUD Matrix to Determine the Products (Figure 4.6), and The Governance Scope Matrix (Figure 4.9).

Also, the models proposed contemplate the four TOGAF´s architectural domains: Business Architecture, Application Architecture, Data Architecture and Technology Architecture. The principles and governance scopes, the definitions of product and definition of team structures are driven by Business Architecture, whereas Applications and Data Architectures are driven by the Product Application Model which defines the scopes of products through the CRUD Matrix to Determine the Products (Figure 4.6). The Microservice Enterprise Reference Architecture Restrictions describing Flexible Technology Standards is essentially a view of Technology Architecture. Based on this

evaluation we prove the feasibility of these models by enterprise architects in the context of MSA at an EAM level perspective.

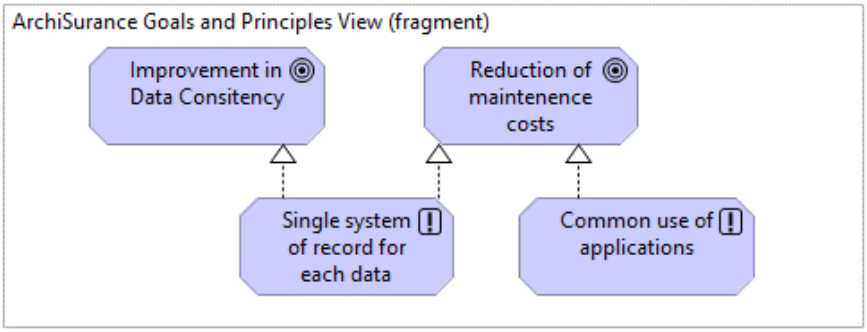
5 Demonstration

In this section we demonstrate how apply the proposed model, addressing the aspects gathered in this research, in order to provide a vision that shows the application in a concrete scenario. To do this we chose to apply a modelling based on ArchiSurance case study. This case study is a white paper published by The Open Group's, which exemplifies an Enterprise Architecture of a hypothetical insurance company (Jonkers, Band, Quartel, & Lankhorst, 2017). The ArchiSurance is a fictitious, but realistic, case study with the objective of illustrating the use of the ArchiMate 3.0.1 modeling language combined with the TOGAF framework. It concerns to describe a new company formed by a fusion of tree previously separate companies. The case describes the EA baseline of this new company which we use to illustrate the application of the model proposed. Although the ArchiSurance case mentions the use of microservices in a specific scenario to support data acquisition, MSA is not its focus, therefore it is not detailed in the case study. Thus, in this section we somehow extrapolate its context for demonstration purposes considering the premise that all systems will be built based on microservices architecture. In the next topics we will describe the use and adaptations of our solution to support the ArchSurance initiative.

5.1 Principles and governance scope for ArchiSurance

ArchiSurance defines some realistic goals that target better consistency, and reduction of maintenance costs, which most company probably are interested in. It was not our intention to subordinate the microservice governance scope principles to a higher business goal. The microservice governance principles are a guide which drive the responsibilities on microservices governance. However, we noted that the ArchiSurance goals could be supported by the architectural principles which we proposed to the governance scopes of MSA. Figure 5.1 depicts the original fragment of goals and principles defined in the paper.

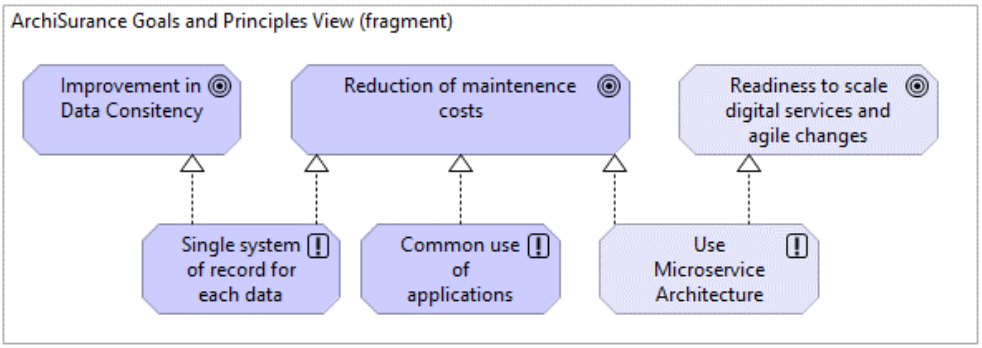
Figure 5.1 – ArchiSurance Goals and Principles



Source (Jonkers et al., 2016, p.11)

As described in the ArchiSurance case study, the resulting company from the fusion should sell its products directly to consumers through the web and other channels. The company must respond to growing competition and seize opportunities in high-growth regions. On the other hand, a flexible application architecture is required in order to make it easier to adapt to changes in business condition, at the same time, changes should not interrupt products and services. ArchiSurance intends to use data from lots of devices, most of them using Internet of Things (IoT) technologies and may invest in real-time costumer interactions. All these needs seem to be grouped into a new business goal in order to prepare the company to scale digital services and quick changes. The goal "Readiness to scale digital services and agile changes" can be realized by MSA due to its scalability and high flexibility. MSA fits into the requirement of implementing changes almost invisibly and without interrupting other services, since a microservice can be changed without affect other components of the system. Thus, we add one goal related to the scenario described to the view of ArchiSurance Goals and Principles, and the use of MSA to realize this goal as a principle. Also, we linked the use of MSA to the Reduction of maintenance costs. Due to microservices' high-decoupling and independence, maintenance on a microservice is supposed to be less costly than on a monolithic system. The resulted adaptation to the view of goals and principles is shown in Figure 5.2 below, where the new elements are highlighted in lighter color.

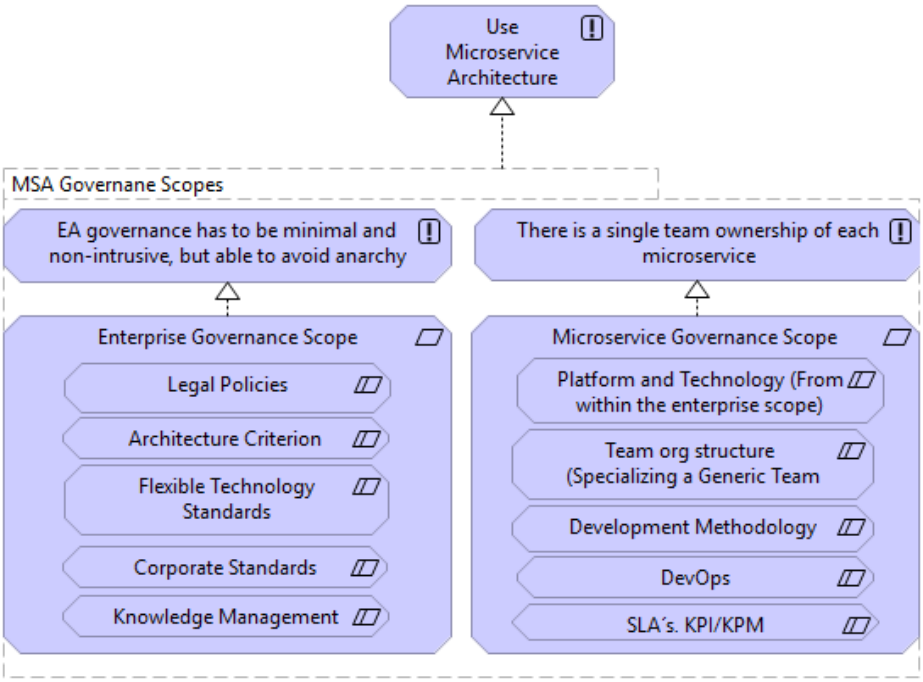
Figure 5.2 - Adaptation of ArchiSurance Goals and Principles



Adapted from (Jonkers et al., 2016, p.11)

As result of use of MSA is needed drivers for its governance making clear to ArchiSurance the scope and responsibilities of its EAM and MSA as designed in section 4.2.1. and depicted in Figure 5.3.

Figure 5.3 - ArchiSurance MSA Principles and Governance Scopes



5.2 Business Processes and Product Applications for ArchiSurance

ArchiSurance needed to improve or change several capacities to implement the strategic and operational transformation. Some of these capabilities are extracted from the Capability Map of the case study.[6] The capabilities are listed as rows of the CRUD Matrix Capabilities & Entities (Figure 5.4). In the ArchiSurance case, the capabilities are realized by resources which we adapted to Entities plotted as columns in this CRUD Matrix, like the one proposed in section 4.2.3 do identify the products scope.

Figure 5.4 - ArchiSurance CRUD Matrix Capabilities & Entities

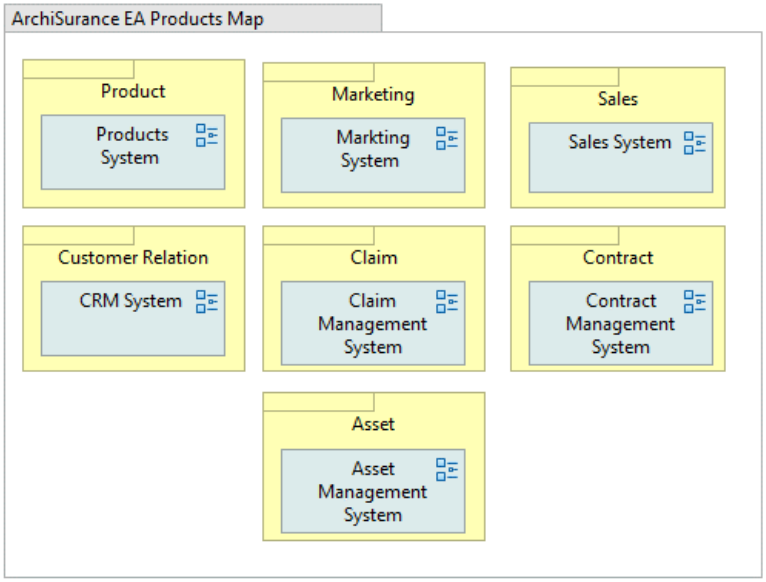
CRUD Matrix Capabilities & Entities Products Microservices Systems		Entities											
		Product	Distribution Channel	**market segments**	Marketing Campaigns	Proposal (Insurance Request)	Policy	Customer	Customer Interactions	Claim	Reserve	Contracts	Financial & Real Estate Assets
Capabilities	Product Definition	CRUD	R	R	Product								
	Product Engeneering	CRUD	R	R									
	Distribution Channel Management	RU	CRUD	R		R							
	Market Development	R		**CRUD**	**CRUD**		markting						
	Sales Execution					CRUD	CRUD	C	Sales				
	Customer Care												
	Customer Service						R	R	CRU	R			
	Service Relation Management						R	R	CRUD				
	Sevice Channel Management							R	CRUD	CRM			
	Customer Data Management						R	R	CRU				
	Digital Channel Management						R	R	CRU	R			
	Claim Management												
	Claim Settlement						R			CUD	C		
	Data Aquisition						R			RU	RU	Claim	
	Data Analysys						R			RU	RU		
	Claim Administration						R				CRUD		
	Contract Lifecycle Management						R		Contract			CRUD	
	Contract Administraion						R					CRUD	
	Asset Management												
	Investment Strategy Management												CRUD
	Investment Performance Management								Asset Management				R
	Investment Portfolio Manament												CRUD
	Asset Inventory Maintenance												CRUD

Some of these scopes will be covered by a package ERP, like CRM. However, we considered, in order to avoid too much customization in these packages, which some

[6] The ArchiSurance Capability Map is shown in Annex A.

specialized function should be built as microservices. Thus, this matrix resulted in the products view shown in Figure 5.5. In this representation the identified products are related to the application collaborations present in the view contained in each product and linked to an Archi[7] view object. This view is not part of the model proposed, and it is used here only to demonstration, because in this tool it seems to be easy drill down through the model views. Clicking in the object "Product System" for example, leads to a more detailed view. These applications collaborations will be detailed later when we describe the EA-Mini-Description.

Figure 5.5 - ArchiSurance Products Map

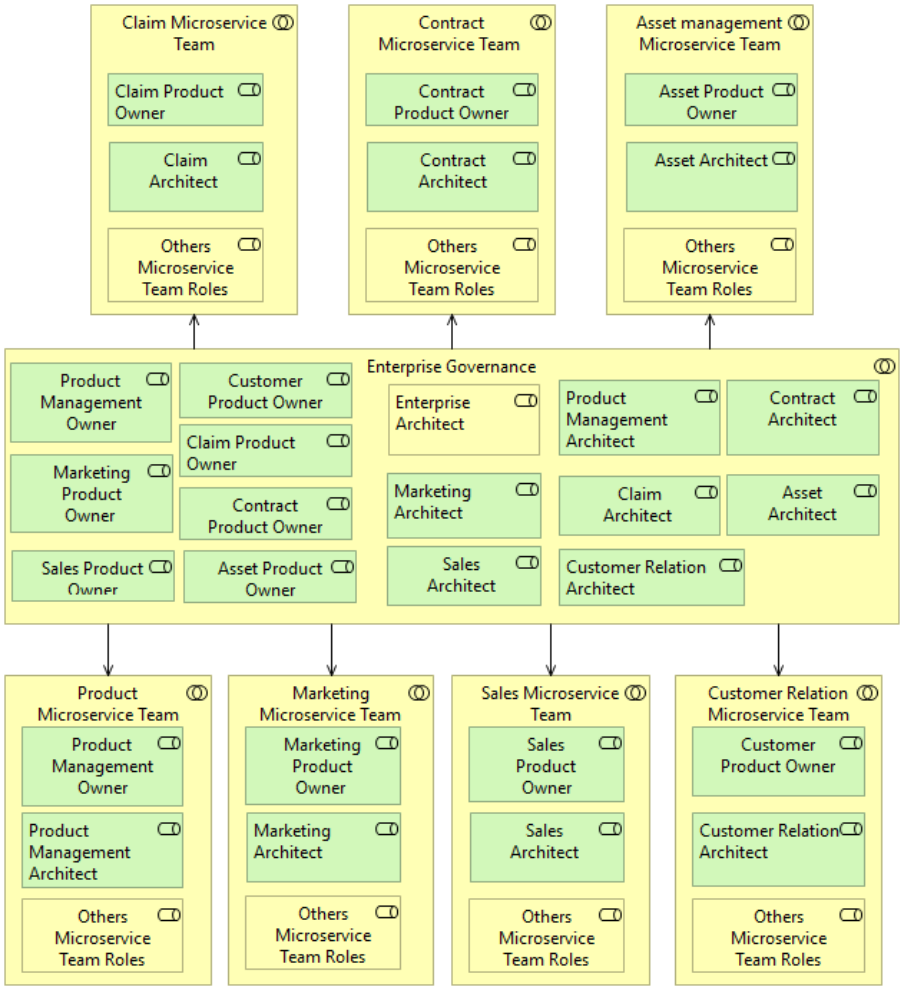


5.3 The organizational team structures for ArchiSurance

Figure 5.6 shows how teams that will support the development of microservices and their governance at the microservices level as well as at the enterprise level would be assembled. As proposed in section 4.2.2, the teams were organized around the identified products and have at least two important roles for the governance of initiatives: product owners and architects. Thus, we represent these roles in each product, which we suggest is still specialized for a named actor. In the context of this work, we just aim to make it clear who has reserved seats and should participate in the governance of the microservices architecture and showing the roles is enough.

[7] Archi is a free and widly ArchiMate modelling tool < https://www.archimatetool.com >

Figure 5.6 - ArchiSurance Organization Structure for MSA

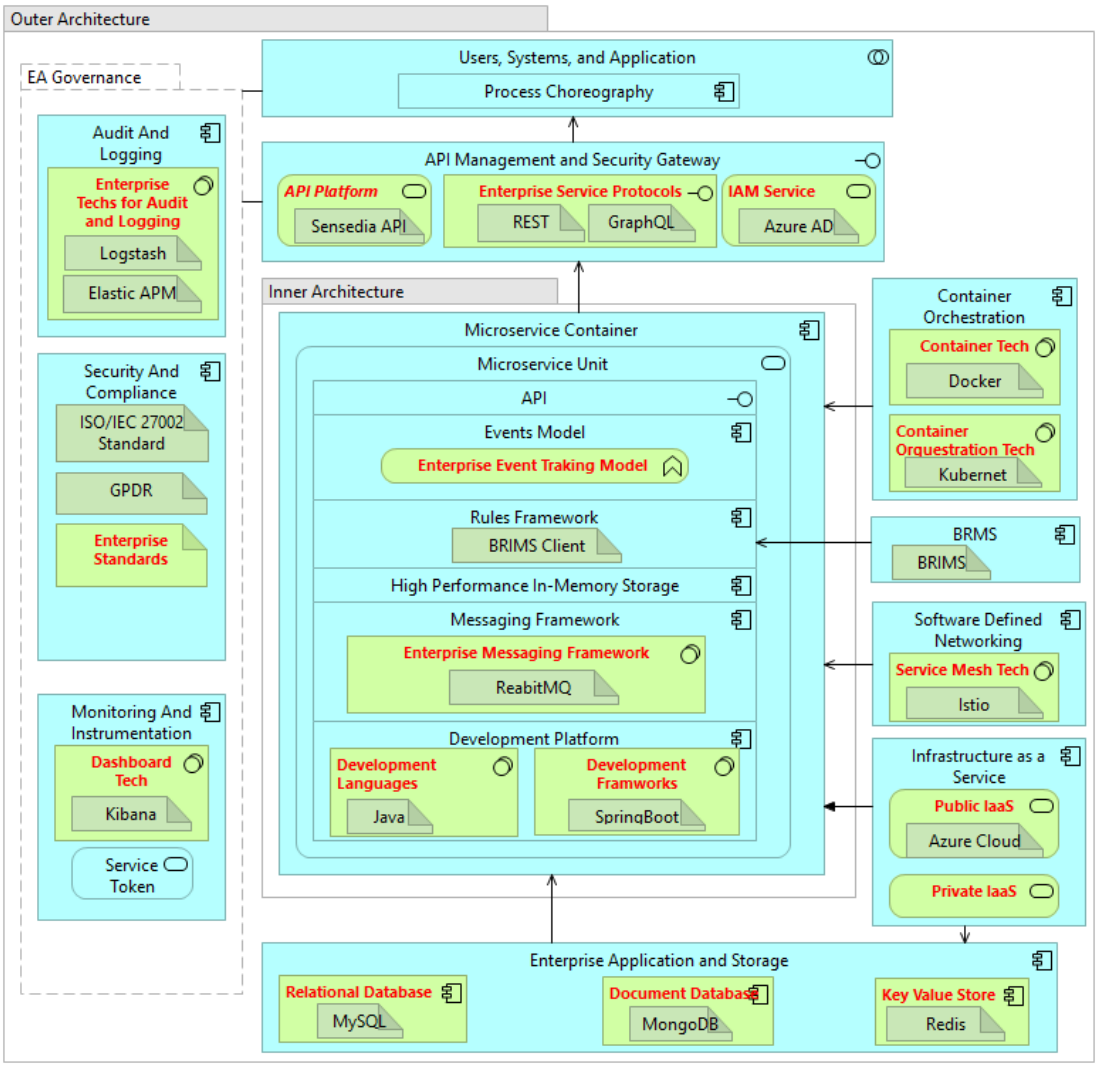


5.4 Flexible technology standards for ArchiSurance

Figure 5.7 shows the technologies chosen for ArchiSurance, which are hypothetical, aiming to demonstrate the organization of the catalog of technologies based on the proposed view in section 4.2.4, and most of these technology came from CNCF (Cloud Native Computer Foundation) landscape.[8]

[8] Cloud Native Interactive Landscape, https://landscape.cncf.io/

Figure 5.7 - ArchiSurance Microservice Enterprise Reference Architecture



5.5 The responsibilities for architectural components

Figure 5.8 shows the governance scope matrix applied to the ArhiSurance case. It is the same matrix proposed initially in the model (section 4.2.5), however we add the BRMS (Business Process Model and Notation) component under the Enterprise Team responsibility. The BRMS is described in the case study, however, was not previously documented in the proposed solution. This demonstrates that this is not a rigid view and can be adapted for each company and context.

Figure 5.8 - ArchiSurance Governance Scope Matrix

	Outer Architecture									Inner Architecture							
Governance Scope Matrix	**Users, Systems, and Application**	**API Management and Security Gateway**	**Monitoring And Instrumentation**	**Container Orchestration**	**Audit And Logging**	**Security And Compliance**	**Software Defined Networking**	**Enterprise Applications and Storage**	**Infrastructure as a Service**	**Microservice Container**	**Microservice Unit**	**Microservice API**	**Events Model**	**Rules Framework**	**High Performance In-Memory Storage**	**Messaging Framework**	**Development Platform**
Legal and Security Policies	ET	ET	ET		ET	ET		ET									
Architecture Criterion	ET	ET	ET	ET	ET	ET	ET	ET	ET	MT	MT	MT	MT	MT	MT	MT	MR
Technology Standards		ET	ET	ET	ET	ET	ET	ET	ET	MT	MT	MT	MT	MT	MT	MT	MT
Enterprise Standards	ET	ET	ET	ET	ET	ET	ET	ET	ET				MR			MR	MR
Knowledge Management										MT	MT	MT	MT	MT	MT	MT	MT
Platform and Technology										MR	MT	MT	MR	MT	MT	MR	MR
Team org structure											MR						
Development Methodology										MT	MT	MT	MT	MT	MT	MT	MT
DevOps										MT	MT	MT	MT	MT	MT	MT	MT
SLA´s, KPI/KPM										MT	MT	MT	MT	MT	MT	MT	MT

5.6 Keeping the ArchiSurance Enterprise Architecture up to date

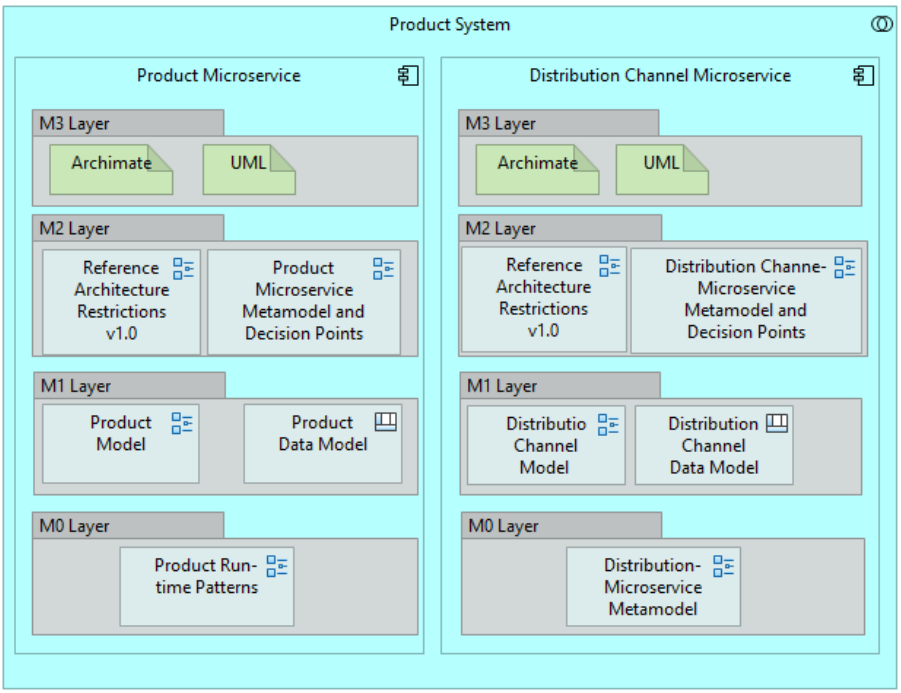
From the EA product Map, we selected the “Product Management System” to demonstrate the way to keep the EA up-to-date from microservices evolvement, as proposed in section 4.2.6.

5.6.1 ArchiSurance Product Management System

In the first view, the Product has a view which is a represetation of the EA-Mini-description of each microservice. These mini-descriptions are mantained by each respective team, but the main view is linked in the Product Map anabling to drill down from the EA high level of products into the Microservices model.

The ArchiSurance Product is realized by the Product System Application Collaboration, which in its turn is composed by two components: Product Management Microservice Subsystem and Distribution Channel Microservice Subsystem. Each of these subsystems is described through the layers from EA-Mini-Descriptions. Figure 5.9 shows the EA-Mini-Description of this product as described before in section 4.2.6.

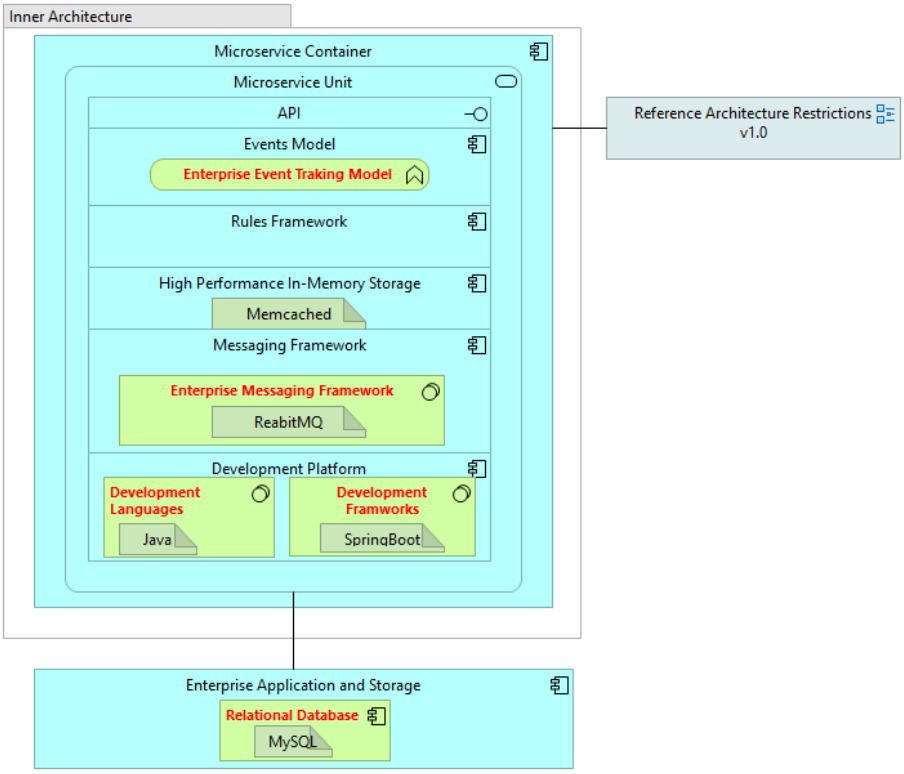
Figure 5.9 - ArchiSurance Product System



In M3 Layer of the Product Management Microservice Subsystem we just report that ArchiMate was used as the main description language, however we have described some views using UML, for instance, we will show ahead that we used the UML class diagram to present the Product Data Model View.

In M2 Layer there are two artifacts, which describe the Reference Architecture Restrictions, and the Subsystem (Product) metamodel and decision points. The first one is a reference of the EA Restriction model followed by the microservice. Considering that this model probably will change throughout the time, it is important to reference the model's version and, in this case, we described it before in section 5.4 Flexible technology standards for ArchiSurance, whereas the Product Microservice Metamodel and Decision Points (Figure 5.10) set down the choices made by the team responsible for microservices within the subsystem Product.

Figure 5.10 - ArchiSurance Product Microservice Metamodel and Decision



M1 Layer should contain important meta-data such as purpose, API endpoints or usage cost model, as well as its inner architectural model (Bogner & Zimmermann, 2016a, 2016b; A. Zimmermann et al., 2018). Thus, for this case we designed the ArchiSurance Product Model (Figure 5.11) and ArchiSurance Product Data Model (Figure 5.12). Figure 5.11 displays the internal application view of the microservice added of some physical information linked to each application component. In this case, we chose to represent the processes supported, the API exposed, the queue topic that it publishes, and the main information entity persisted by the microservice, whereas the AchiSurance Product Data Model (Figure 5.12) details the domain entities maintained within the microservice.

Figure 5.11 - ArchiSurance Product Model

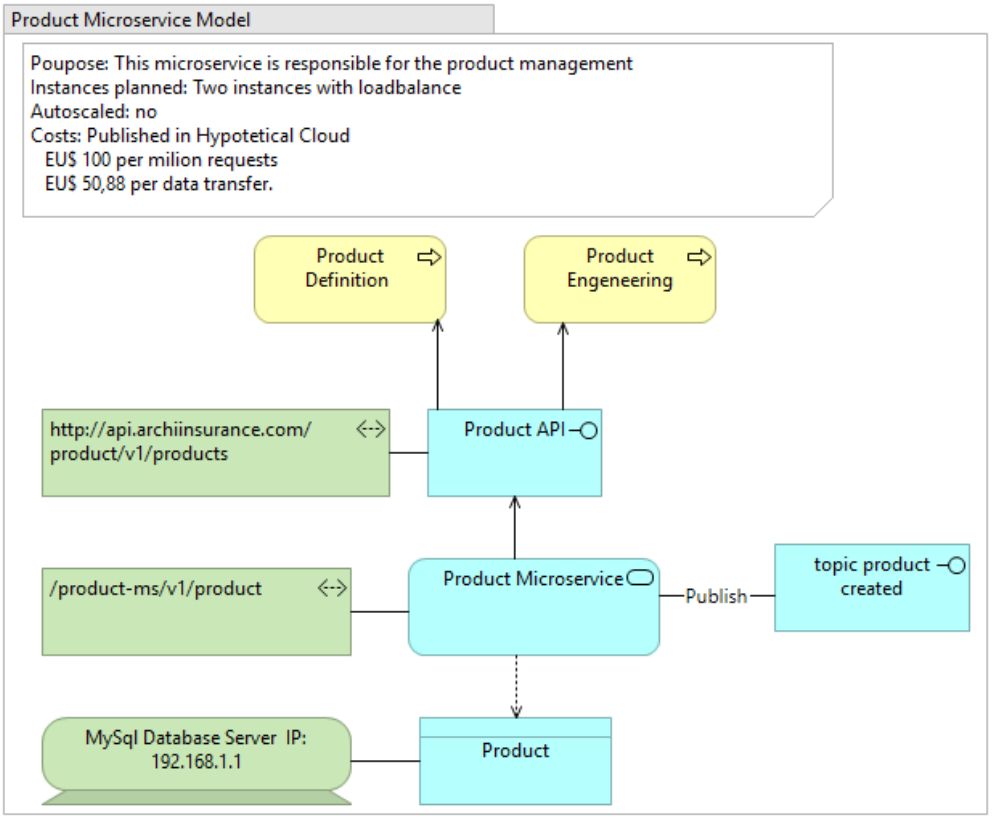
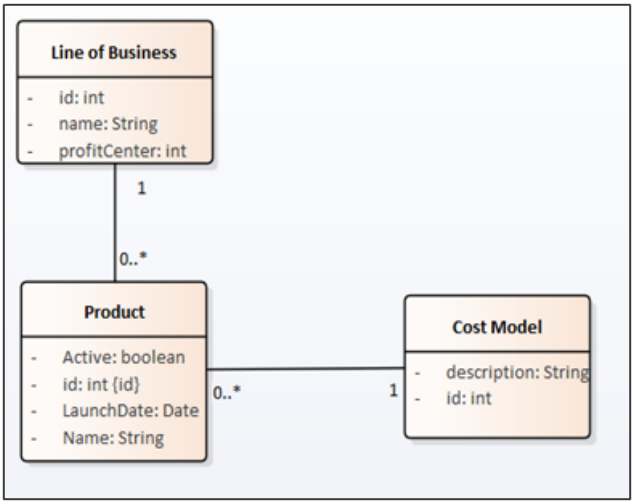


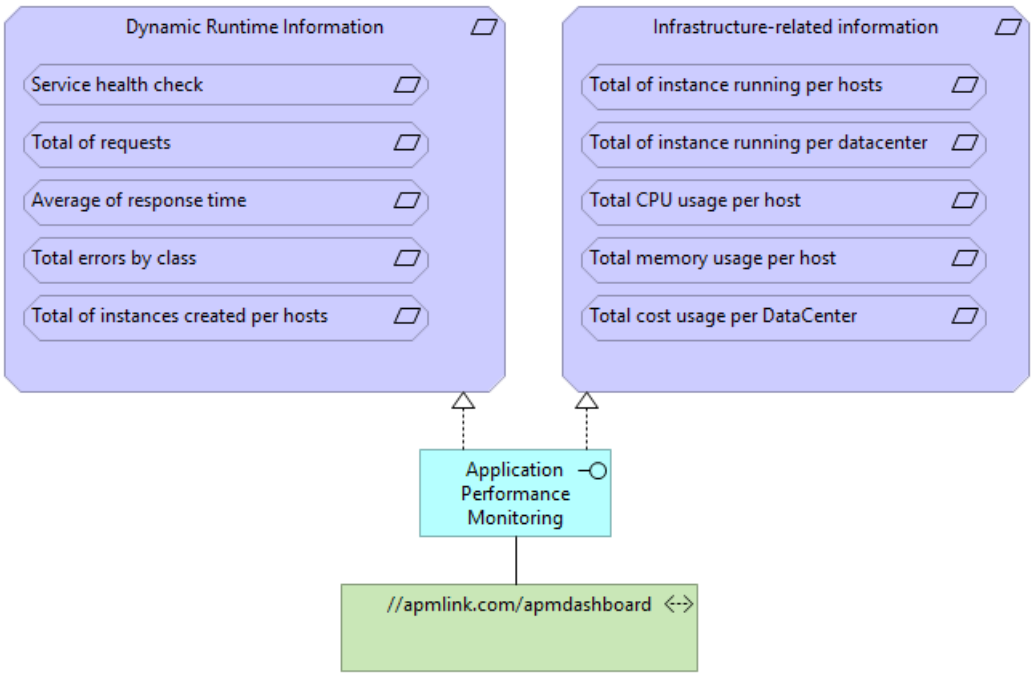
Figure 5.12 - ArchiSurance Product Data Model



M0 consists of operational run-time or monitoring data. Thus, in order to support this layer, we included a view of the requirements to drive the microservice's monitoring, which should be guaranteed by the microservice shown in Figure 5.13. This set of requirements is

an adaptation from Mayer & Weinreich (2017), just for exemplification. Our diagram also maps the source of data to support these requirements, in this case an API provided by the Application Performance Monitoring tool and linked to the dashboard address in this tool.

Figure 5.13 - ArchiSurance Runtime Monitoring Requirements



Through this adaptation of ArchiSurance case study for demonstration purposes, we have exemplified the application of the model proposed in section 4 (Proposal), providing a fluid, integrated and consistent view of how the model can be applied to planning and updating the EA Model in a scenario of intensive use of MSA, assisting in the clarity of how to apply it, as well as confirming the validity of the model in a realistic case.

6 Evaluation

This section describes the evaluation of the research as prescribed by DSRM (Peffers et al., 2007) as explained in section 1.3. The evaluation activity seeks to observe how well the artifacts created support a solution to the research problems.

In previous sections, we reinforced the need for EAM to supports adopting MSA and gathered some issues and difficulties at EAM reported in the literature, such as the problem created in strategy synchronization due to the reduction of need for communication (Lenarduzzi & Sievi-Korte, 2018), the difficulty in clearly determining the scope and the proper granularity of microservices (Soldani et al., 2018), the possible cost inefficiency (Yale Yu et al., 2016), keeping staff trained, and controlling the complexity of coordination between the different teams (Salah et al., 2016). Thus, we identified the change of the role of EAM from being restrictive to an advisory role targeting to support teams in cross concerns, focusing on tracking changes in EA, ensuring cost transparency and risk management. Thus, to support this new role of EAM, we proposed a model with some architectural views by merging other proposed architectures and complementing them, such as The Open Group Microservice Architecture Reference ("The SOA Source Book - Microservices Architecture," 2016) and Bogner and Zimmermann's EA-Mini-Descriptions (Bogner & Zimmermann, 2016a). We seek to provide an approach to delimit the scope of microservices products, which helps the choice of the team responsible for the products, defining a structure for microservice teams, a technology view that allows to visually maintain a catalog of technologies at the EAM level and a proposal to keep the EA model up to date from the evolution of each microservice. In order to evaluate the research, we conducted interviews with some architects and senior developers from organizations that use microservices to validate the solution's utility and confirm the research results.

6.1 Evaluation of ArchiInsurance Case Study

The ArchiSurance case study addresses in its description the adoption of microservice architecture, just to support some IoT data acquisition interactions, however, not as part of a system based on this architectural style. We believe that with the proposed adoption of MSA,

the company will be ready to scale its digital services and to better seize opportunities in growing markets. On the other hand, MSA promotes greater change agility and thus, the company will also be better prepared for market changes.

We demonstrated in section 5 the applicability of the proposed model and addressed points that were not covered in the original case. Therefore, we extrapolated the case, but always based on the scenario described in the case study. To support needs that did not seem to be met by other existing objectives in the case, such as flexibility and scalability to grow, we added a new business goal: Readiness to scale digital services and agile changes. This goal justifies the adoption of MSA.

In ArchiSurance's case, there is a clear domain decomposition logic, which could also identify some bounded context for microservices, however it derives only from business capabilities. The use of finer and independent granularity components is not perceived. Also, most of the systems designed would have a monolithic architecture as stated when describing their IS Architecture (Jonkers et al., 2016, p.24). We realized that this logic would still require more modeling effort to define the data boundary within each designed system. In our proposed model, however, through CRUD matrix, we quickly identified clusters of business capabilities and informational entities to explicitly determine products that would be developed based on MSA and to determine a team structure to support these products, in their detailed design, development and operation. This level of detail is not achieved in the original case study, which built the view of the organizational structure at a very high level, basically defining some functional boards and departments. However, we were able to demonstrate the potential contribution of the proposal to the ArchiInsurance scenario.

The benefits to ArchInsurance, such as agility to business changes through DevOps practices such as continuous integration and delivery (CI/CD), reducing investment costs using highly scalable cloud infrastructure, and reducing cost of maintenance due to the independence among microservices, are benefits promised by MSA in general. There is no reference in the case study to their EA model update, in this field, our proposal implements a mechanism that will follow the evolutionary changes on the architecture over time. Finally, by choosing to use MSA more intensively in its IS Architecture, ArchiSurance would also be exposing itself to the same risks deriving from MSA's choices and thus, this proposal would help the company by reducing its exposure to these risks, which are explained in the next section.

6.2 Interview and questionnaire

We interviewed practitioners targeting to evaluate the contribution of the research and the proposed model from the perspective of mitigating the enterprise risks related to the aspects identified in the literature review. In addition to the opinion on the viability and utility of the artifacts, which has also been demonstrated previously in section 5, that by correlating the proposed solution to enterprise risk we obtain more systematic and objective evidence on the contribution of solution to the EAM, proving both its effectiveness and its relevance to the enterprise environment.

In these interviews we follow a dynamic where we present to the interviewees the research questions, the problems identified in the literature and the proposed solution. Then we asked them to answer the questionnaire pointing out the risks, in order to confirm their perception if the issues of MSA presented as a whole were linked to the listed risks, how much they agree with this, and whether the proposed solution as a whole contributes to mitigating each risk listed. At the end, we concluded the interviews by asking them to comment on the points they considered most relevant, seeking open feedback in their answers to each question aiming to identify what worked better and understand what did not work so well. The risks are listed below and better described ahead in this section.

1. Risk of reduce the flexibility / agility of IT and business deliveries
2. Risk of increase the cost of training
3. Risk of increase the cost of licensing
4. Risk on control the cost of IT infrastructure
5. Risk of exposure to lack of support
6. Risk of an inappropriate use of IT Solutions
7. Risk of reduce the level of service reuse
8. Risk of non-compliance with corporate security policies
9. Risk of non-compliance with external regulatory policies

To confirm that the problems identified are related to enterprise concerns, we correlate them to risks that in turn have been mapped to the following IT related goals described in COBIT (2012) and shown in Table 6.1. In the interviews we did not discuss directly these COBIT IT Goals, once they were already linked to the risks identified. We focused on confirming risk perceptions through objective questions that were filled out during the interviews, by asking about the level of agreement to the risk and its impact.

Table 6.1 - IT Related Goals

	Information and Related Technology Goal
01	Alignment of IT and business strategy
02	IT compliance and support for business compliance with external laws and regulations
03	Commitment of executive management for making IT-related decisions
04	Managed IT-related business risk
05	Realized benefits from IT-enabled investments and services portfolio
06	Transparency of IT costs, benefits and risk
07	Delivery of IT services in line with business requirements
08	Adequate use of applications, information and technology solutions
09	IT agility
10	Security of information, processing infrastructure and applications
11	Optimization of IT assets, resources and capabilities
12	Enablement and support of business processes by integrating applications and technology into business processes
13	Delivery of programs delivering benefits, on time, on budget, and meeting requirements and quality standards
14	Availability of reliable and useful information for decision making
15	IT compliance with internal policies
16	Competent and motivated business and IT personnel
17	Knowledge, expertise and initiatives for business innovation

Source (*COBIT. A Business Framework for the Governance and Management of Enterprise IT*, 2012)

In addition to help in the validation of the proposed solution utility, the evaluation from the perspective of enterprise risks also provides a valuable discussion that helps mitigate the risks identified in planning of MSA adoption. The correlations of the risks and the COBIT's IT goals are demonstrated in Table 6.2.

Table 6.2 - Mapping Risks to IT-Related Goals

(Gray cells identify the relationship between Risk and IT Goal)

	Risks								
Cobit IT Goals	1.Reduce flexibility / agility	2.Increased cost of training	3.Increase in Licensing Cost	4.Increase in cost of IT infrastructure	5.Exposure to lack of support	6.Inappropriate use of IT solution	7.Reduction of the level of service reuse	8.Non-compliance with enterprise security policies	9.Non-compliance with regulatory policies
02 IT compliance and support for business compliance with external laws and regulations									
04 Managed IT-related business risk									
05 Realized benefits from IT-enabled investments and services portfolio									
06 Transparency of IT costs, benefits and risk									
08 Adequate use of applications, information and technology solutions									
09 IT agility									
10 Security of information, processing infrastructure and applications									
11 Optimization of IT assets, resources and capabilities									
12 Enablement and support of business processes by integrating applications and technology into business processes									
13 Delivery of programs delivering benefits, on time, on budget, and meeting requirements and quality standards									
15 IT compliance with internal policies									
16 Competent and motivated business and IT personnel									
17.Knowledge, expertise and initiatives for business innovation									

The risks identified and their relations to IT Goals are explained below.

1. Reduce flexibility / agility

Despite the benefits of the decentralized decisions about the best technology for each case, the totally uncontrolled diversity of technologies have negative impacts on IT agility since it can reduce flexibility and mobility of human resources, causing loss of agility and time to market. Also, the lack of mobility can impact the motivation of business and IT people. It can also block the optimization of IT assets and resources. Other important role of IT is supporting business innovations, however, keeping the knowledge and the expertise in these technologies have become harder and by this perspective it can impact the initiatives for business innovations. In addition, it allows choices to use specific solutions to solve local problems, however it may be inadequate from an enterprise standpoint.

2. Increased cost of training

Uncontrolled diversity can increase the cost of training. The risk of increasing the cost of training people in our view is related to transparency of cost, the benefits, and risks of each technology. Choosing a new technology to solve a problem which has already been solved using existing technology in the enterprise implies the loss of agility for teams to mature the new technology. This aspect is also related to not optimizing IT assets and resources. Moreover, we can see that it can negatively impact the knowledge and expertise to support the business innovations and to keep the competency and motivation of business and IT personnel.

3. Increase in Licensing Cost

Diversity of technologies for the same purpose can increase the cost of licenses, the cost of using clouds, and the difficulty of managing them. The diversity and duplication of technologies may not realize holistically the actual benefits intended by the investments in IT and may complicate the management of services portfolio. Although this diversity enables

a cost rationalization of cloud usage, the effective cost of managing it may be hidden, thus implying a lack of IT cost transparency and a non-optimized use of IT assets and resources.

4. Increase in cost of IT infrastructure

Cloud diversity can cause an increase or a lack of control over the cost of using infrastructure resources in the cloud. The cost increase of IT infrastructure may not completely realize the benefits of IT investments and may interfere in risks transparency, if not clearly controlled. It may impact the processing infrastructure in case of a lack of enough budget planned to support infrastructural increments and may affect the optimization of IT capabilities.

5. Exposure to lack of support.

Completely autonomous choices can lead to the choice of technologies which are not robust in terms of guaranteeing continuity, such as technologies that may be discontinued, abandoned by the community. It may be harder to recruit people with knowledge of technology, or interested in learning it, implying in difficulties to manage IT people and, consequently, in a risk related to business continuity. The choices of deprecated technologies, or of one with a small community, may hide the cost spent internally to maintain the support at the IT level, as well as the actual balance between benefits and risks of such technologies. This kind of situations sounds like an inadequate use of technological solution, as well as affecting IT agility, because the staff will probably have to investigate the problem without the help of a specialized partner. With no one to call upon, no expert and community support, the organization's professionals can come across a sense of powerlessness to solve IT problems and this may be a factor impacting overall motivation of these professionals.

6. Inappropriate use of IT solution.

The autonomous choice of technology can lead to inappropriate choices from the point of view of integration with other enterprise applications. For example, the use of a communication protocol which other enterprise applications and partners are not ready to

integrate. Clearly, this risk is strongly related to the adequate use of technological solutions. However, it also implies in a not optimized use of IT assets and resources and in the end reducing IT agility. Some choices may not best support the integration of applications, making it harder to support business processes.

7. Reduction of the level of service reuse.

Reuse is an important principle and benefit of SOA. Poor definition of the microservices scope can lead to duplicate services, contributing to inefficiencies in cost management and IT capacity. A low level of reuse implies unoptimized use of resources and capabilities. Probably, the cost of maintaining two services will not be clear either, as well as their benefits. Reuse is an important enabler of IT agility.

8. Non-compliance with corporate security policies.

The greater the diversity of technologies for the same purpose, the greater the difficulty of managing and mitigating possible information security vulnerabilities. It implies in a lack of transparency of risks, lack of adequate information and exposition to information security policies and other internal policies.

9. Non-compliance with regulatory policies.

A regulatory rule may determine that data records must be immutable and stored for a given period of time in a given environment. For instance, the accounting system, in which a once-registered accounting event can no longer be changed, only compensated. This risk is directly related to IT compliance and support for business compliance with external law and regulations. Also, it is important to have the transparency of this risk, its cost and benefits to mitigate it. Also, a non-compliance may result in an inadequate use of information.

6.2.1 The questionnaire

The questionnaire was composed by the nine risks identified. For each risk two questions using a 5-points Likert scale was applied seeking to confirm the existence of the risk, and how much the presented model reduces the likelihood or impact of the risk.

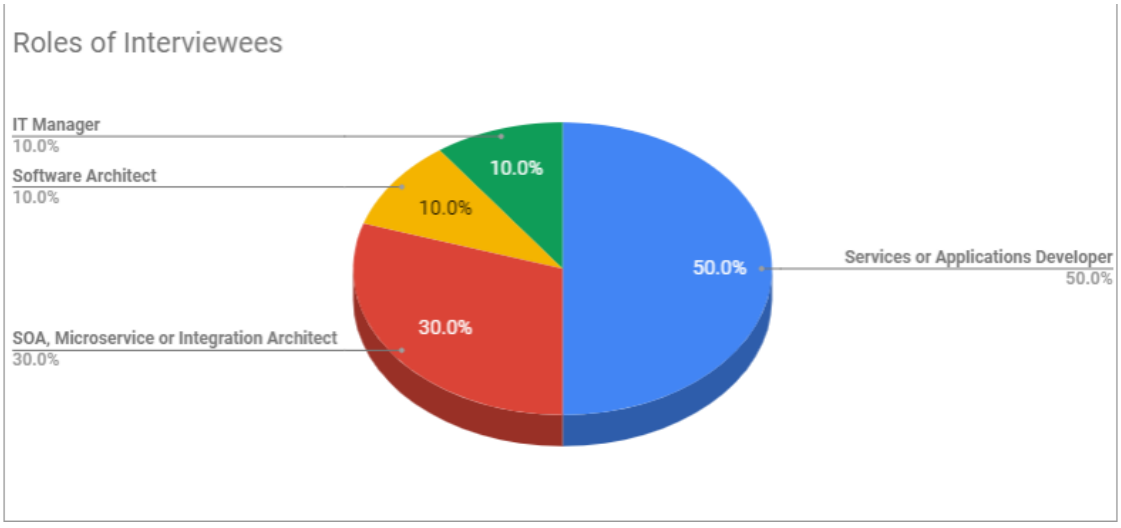
6.2.1.1 Respondents qualification

Ten IT professionals participated in the interviews. All of them being part of microservice teams as architects, senior developers, or even leading the microservices initiatives from the architectural perspective in four different organizations, described below.

- A company specialized in CRM tools with a strong presence in the Brazilian shopping centers market. This company is refactoring their CRM products to a microservice architecture, aiming scalability and changing their business model from licensing to a pay-as-you-go model.
- Two of the companies are part of the biggest media group of Latin America, one responsible for the most read newspaper and the other for a big number of magazines. Both groups have been using intensively microservices to support the big number of mobile and online readers.
- A consulting firm specializing in APIs and microservices that ranks as the leader in Forrester and as a visionary in Gartner with a strong customer base and that has supported MSA adoption in biggies companies in areas like banks, insurance, payments and others.

The roles played by the respondents and their distribution are listed in the pie chart below:

Figure 6.1 – Qualification of Respondents by Role



6.3 Results Analysis

6.3.1 Evaluation of Risk Analysis

We asked to the respondents how much they agreed with the risk assessment listed by adopting the microservice architecture (MSA) on a degree of agreement scale from 1 to 5 as shown in Table 6.3.

Table 6.3 - Legend of Risk Evaluation

1	**Totally Disagree:** meaning the non-recognition of the existence of risk at the enterprise level
2	**Partially Disagree:** meaning that partially disagrees with the risk, and the identified risk and its impacts are not significant.
3	**Indifferent:** Meaning a neutral position of neither agreeing nor disagreeing with the existence of risk
4	**Partially Agree:** Partially agree with the existence of the risk and its impacts
5	**Totally Agree:** Meaning full recognition of risk and its impacts at enterprise level

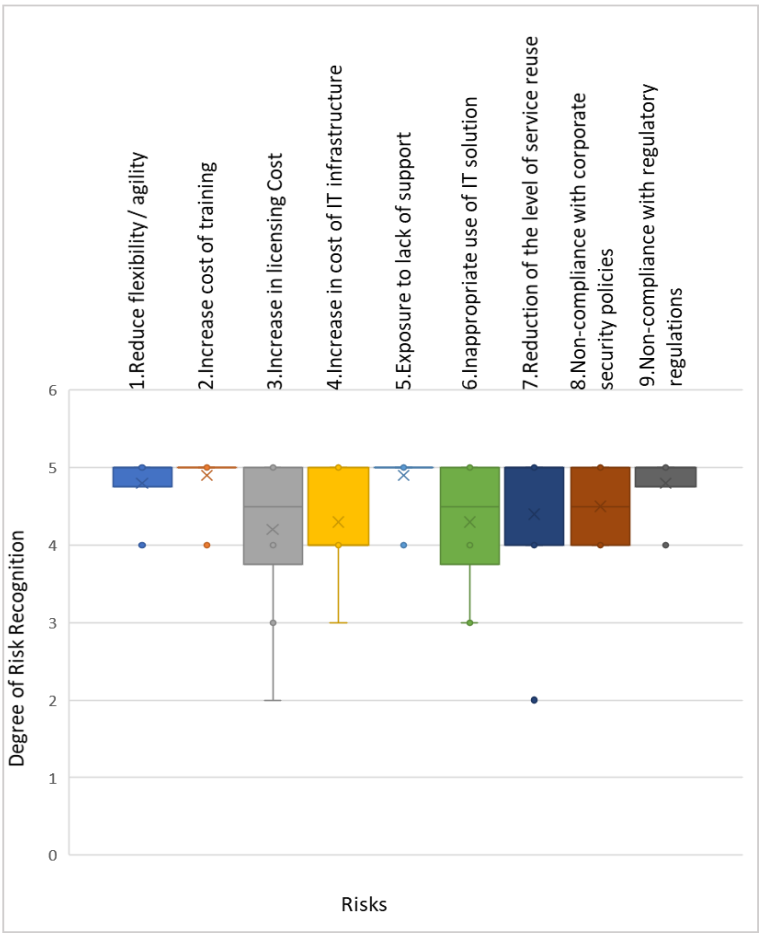
From the enterprise risks perspective most of respondents agree with the existence of the risks identified as shown in Table 6.4.

Table 6.4 - Risk evaluation

	Answers				
Degree of agreement with risk	**1**	**2**	**3**	**4**	**5**
1.Reduce flexibility / agility	-	-	-	2	8
2.Increase cost of training	-	-	-	1	9
3.Increase in licensing Cost	-	1	1	3	5
4.Increase in cost of IT infrastructure	-	-	1	4	5
5.Exposure to lack of support	-	-	-	1	9
6.Inappropriate use of IT solution	-	-	2	3	5
7.Reduction of the level of service reuse	-	1	-	3	6
8.Non-compliance with corporate security policies	-	-	-	5	5
9.Non-compliance with regulatory regulations	-	-	-	2	8

Figure 6.2 shows a box plot chart, where we can observe the distribution on risk recognition discarding outliers, which are very discrepant responses which could distort the analysis.

Figure 6.2 - Risks Box Plot Chart



From this chart, discarding outliers, we can note that the smallest variation and at the same time the highest value correspond to risks of increase cost of training and exposure to lack of support, followed by reduce flexibility/agility and non-compliance with regulatory regulations, also with a small variation and high value. Thus, we conclude that these risks have greater recognition among respondents. While the biggest variations are in risks of increase in licensing cost and inappropriate use of IT solution, indicating lower agreement among the respondents. However, by analyzing the range occupied between the first and third quartile (body of the candle) we realize that all the identified risks were recognized by

them. On the other hand, the major risks for these organizations, increase cost of training and exposure to lack of support, indicate a strong concern regarding the knowledge and effective mastery of the technologies used. These two risks point towards difficulties in solving problems in these technologies, reducing the speed of solutions delivery, which in turn reinforces the risk of reduce flexibility/agility.

Some considerations were made on the open feedback question about these risks:

- The probability of microservice increased licensing cost, as well infrastructure and cloud usage costs, related to the risks 3 and 4, may be low, considering the self-sizing capacity of the microservices instances. Microservices often use open-source tools and resources charged for their actual use. So, if well planned, they can reduce these costs.
- About the risk related to Inappropriate use of IT Solution, the argument that autonomous choices can make integration to other enterprise applications difficult may have low probability and impact, because microservices communicate through different ports to meet technical requirements for integration. They are easily evolved if a new port is needed. On the other hand, enterprise applications generally communicate through an API Gateway that centralizes and adapts the protocols.
- The duplication of function and eventual reduction of reuse may be acceptable in favor of agility to changes and not having to reconcile interests in different contexts. For example, a microservice responsible for "product" data seems to be reusable, but when we take it into the context of a pay TV subscriber it will have a universe of data and operations distinct from a product in the context of a print magazine. Although the entity seems the same, in different contexts that might not be true. Thus, the reduction of the level of reuse would not be so relevant in the context of microservices. It must be balanced with the delivery agility aiming at the time to market. Thus, it is a topic that needs analysis of risk and benefits.

6.3.2 Evaluation of the solution proposed

After presenting a summary of motivations, related work, issues, mapped risks, and the proposed solution we asked the respondents how much they agree that the solution presented helps to solve the risk identified related to the MSA on a degree of agreement scale from 1 to 5 as shown in Table 6.5.

Table 6.5 - Legend of Solution Evaluation

1	**Totally Disagree:** meaning that the solution does not contribute to reducing the associated risk.
2	**Partially Disagree:** Meaning a neutral position of neither agreeing nor disagreeing that the solution presented helps to reduce the associated risk.
3	**Indifferent:** Meaning a neutral position of neither agreeing nor disagreeing that the solution presented helps to reduce the associated risk.
4	**Partially Agree:** Partially agree that the solution presented helps to reduce the associated risk and its impacts on the enterprise
5	**Totally Agree:** Meaning full recognition that the solution presented helps in the reduction of the associated risk and its impacts at enterprise level

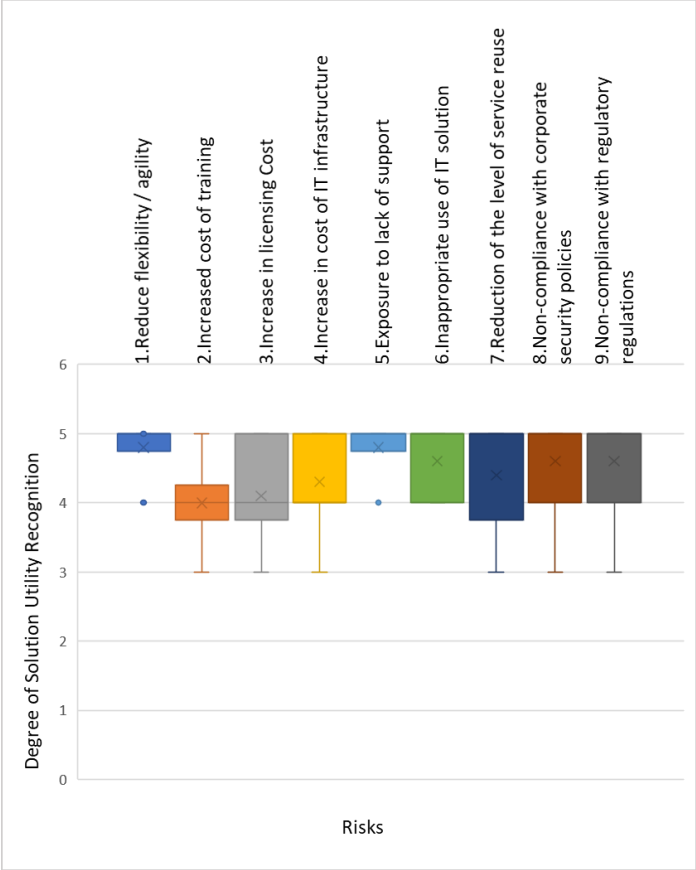
From the perspective that the solution presented is useful to mitigate the risk associated to each problem which it tries to solve, most of respondents agreed that in general the model is useful, which is demonstrated in Table 6.6.

Table 6.6 - Utility evaluation of the solution

	Answers				
Degree of agreement that the model mitigates the risk	**1**	**2**	**3**	**4**	**5**
1.Reduce flexibility / agility	-	-	-	2	8
2.Increased cost of training	-	-	2	6	2
3.Increase in Licensing Cost	-	-	2	5	3
4.Increase in cost of IT infrastructure	-	-	1	5	4
5.Exposure to no support	-	-	-	2	8
6.Inappropriate use of IT solution	-	-	-	4	6
7.Reduction of the level of service reuse	-	-	2	2	6
8.Non-compliance with corporate security policies	-	-	1	2	7
9.Non-compliance with regulatory regulations	-	-	1	2	7

Figure 6.3 shows a box plot chart, where we can observe the distribution on recognition of the utility of the proposal, and how it mitigated the risks identified. This chart discards outliers (discrepant responses), avoiding distorted analysis.

Figure 6.3 - Box Plot Chart of Solution Utility



It confirms that the proposal works to reduce all risks identified, since all risks had high scores when disregarding outliers. Still, we may have some analysis that helps us understand how much the proposed model contributes to reducing the identified risks. The smallest variations are relative to mitigation of the risks on reduce flexibility/agility and on exposure to lack of support, indicating greater consensus among respondents. Moreover, this consensus is in the highest range, which demonstrates that major impacts in terms of risk mitigation of the solutions are most strongly related to these two risks. On the other hand, the largest variations are related to risks mitigation on increase licensing cost and reduction of service reuse level, and therefore, there are differences among respondents in their perceptions on the value of the solution in reducing these risks. Although we find less

consensus on these risks, they are still in a high range. The smallest contribution is related to the reduction of increase cost of training, which is far from the maximum value, thus being the weakest point regarding the perception of the utility of the proposal. Mitigating or controlling training costs, in turn, is directly linked to the concern about the knowledge and effective mastery of the technologies used, diagnosed in risk probability analysis.

At the end, the participants were asked to give an open feedback focusing on the analysis of the perception of how the model could contribute to their organizations and the reasons that lead them to not completely agree with the solution when they did not respond "agree totally" with the model presented in terms of contribution to the reduction of identified risks.

In general, they reinforced that considered the model useful, mainly to give a drive to enterprise concerns, technologies and provide other insights to MSA adoption. They reported some similar cases, about needing to rewrite code due to wrong choices of technologies, the difficulty keeping teams trained, or substituting a team member specialized in one technology, and the challenge to calculate accurately the infrastructure cost before build each microservice, despite available calculators across cloud services. The CRM company demonstrated little concerns about licensing and support costs, they prefer use open source and free technologies, whereas they are very sensitive to the infrastructure cost. The media company, since it is much bigger, is more concerned about the exposition to support and the compliance with security and internal policies. These differences of focus show how the risks have different weights for each company.

- Related to increased cost of training, the model may help to control diversity, but regardless of the technologies, people must still be trained, and the model seems to not drive the cost of training directly.
- The presented model does not totally solve the risk of duplication of functions and microservices. But it contributes somewhat to the level of service reuse as it defines a context for the scope of the microservice system at a high level, though. It also does not go so far as to help to define the granularity of each microservice, since the scope is only defined in CRUD matrix at a very high level.
- As for the EA-Mini-Descriptions proposal, it is not clear how the navigation would be drilled down, even though drilling up seems to make sense. On the other hand, for the

internal architecture of the microservice that would be the responsibility of each team, the mini descriptions may not be clear enough to provide an accurate view and reduce the related risks.

6.4 Communication

This section addresses the communication activity prescribed by DSRM and targets to communicate the problem and its importance, the resulted artifacts model, its use, and its utility for the users. The communication tests the acceptance of the research outcomes, which provides information about the proposed work, the problem's importance, and the solution objective's feasibility. To communicate this work, we published two papers in the following academic events:

- 9th International Symposium on Business Modelling and Software Design (BMSD 2019) realized in 1-3 July, Lisbon, Portugal [9]
- 9th Enterprise Engineering Working Conference (EEWC 2019) realized in May 20th to 24th, Lisbon, Portugal [10].

In addition, this dissertation also contributes to the communication of the research realized, presenting to the scientific community all the details of the problem, solution and results achieved, as well as providing a view of possible future work about related issues still opened.

[9] https://link.springer.com/chapter/10.1007/978-3-030-24854-3_17
[10] http://ceur-ws.org/Vol-2408/paper8.pdf

7 Conclusions

In this Section we summarize the conclusions of this research, its contribution (section 7.1), limitations and future work (section 7.2).

This work addressed the lack of modelling elements to describe microservice-based systems within an EA model. This makes it difficult in EAM to handle the scope definition, technologies heterogeneity, cost volatility, and implications of the decentralized governance, which are relevant aspects of MSA that impact the whole organization and the EAM. We observed these difficulties and designed a solution, using the DSRM for Information Systems Research (described in section 1.3) to develop this research.

Through literature review, we identified that the major implications of MSA on EAM are related to the autonomy of the microservice team in decentralized governance, which demands new approaches to allow EAM to control technological diversity, to define the scope and the size of the MSA-based system, to define the team that owns a microservice, and to control costs and risks. We consolidate the MSA aspects related to EAM and propose a model to describe them at an EA model in order to allow EAM to govern EA concerns on MSA adoption, as well as keeping the EAM information up to date from the evolution of each microservice. We demonstrated the feasibility of the model through an example applied to a hypothetical case in section 5, and in section 6 we confirmed the utility of the model in helping EAM play its role in MSA governance to ensure strategic and IT goals realization, business benefits, cost transparency, and mitigation of IT and business risks. On the other hand, we proved that the model proposed can reduce enterprise risks related to COBIT's IT goals.

7.1 Contribution

In this Section we present the major contributions of this work.

The identification of MSA aspects and their impacts on EAM concerns provides a consolidated view linking both subjects that may see far from each other and which are described in section 4.1 and summarized here. We found that to achieve high decoupling in

microservices, it is necessary to break down a business domain mapped from the enterprise's architecture. The agonistic implementation, while bringing advantages and flexibility, also carries with it risks of having to deal with the excessive diversity of technologies, which can reduce the flexibility promised. Another aspect of microservices is to be cloud-based in order to achieve scalability. However, the costs of this infrastructure require attention in terms of control and transparency for enterprise stakeholders. The API, being the main communication layer, exposes the organization's capabilities to customers, and partnerships must be planned and governed at the enterprise level. Decentralized governance is also an important factor for flexibility and agility. However, it can reduce communication among teams and may represent a risk to strategic alignments. On the other hand, the bounded context of the microservice must be defined from the business domain view. Moreover, MSA requires a new organizational structure around products by adopting DevOps practices in a single team in order to support the business changes more quickly. This view may help other researches in this line of investigation providing a point of reflection and trade-off about the sizing and scope definition of microservice-based systems controlling technological diversity, cost volatility, the impact of decentralized governance, and the key EAM responsibilities in MSA governance, as addressed in section 4.2.

To support EAM in driving its most relevant concerns on MSA we developed an ArchiMate model defining principles, governance responsibilities, MSA product scopes, a team structure and an enterprise technology architecture view for MSA, which enables to easily maintain relevant requirements and recommendations from EAM requirements to the microservices teams, as well as keeping relevant information about microservices and their evolution, providing cost transparency and balancing the benefits of decentralized governance of microservices.

The model proposed addresses aspects that other references, when isolated, do not cover. For instance, The Open Group Microservice Architecture does not cover EA artifacts and runtime data to monitor information at the EA level. The Bogner & Zimmermann's EA-Mini-description also does not address aspects like scope definition, team structure and how exactly EAM plays its role to drive MSA adoption. Furthermore, we identified two important gaps which are not covered by any of the other references, which we addressed to complement the reference, such as a method for defining the scope and size of an MSA-

based system, and a model for maintaining an architecture catalog of technologies, able to avoid excessive diversity. This model can be applied to any company.

Lastly, the impact analysis of MSA over enterprise risks reinforces the need for EAM supports and governs cross concerns related to MSA at the enterprise level which may help companies that want to adopt MSA, as well as other researchers, to reflect about how to deal with these risks.

7.2 Limitations and Future Work

Despite the feasibility, the match of two different approaches, a top-down one, to plan MSA adoption and another to update the model from the bottom-up perspective, does not present an easy solution with fluid connection. Thus, some gaps persist, such as in knowledge management, which is not explicit in the proposed modeling yet. In addition, there seem to be still a lot of shared responsibilities between governance at the enterprise level and governance at the microservice level. It is not intentioned in this research, but it certainly opens a point of discussion about the actual benefits in keeping each aspect under EA governance, or simply delegating it to the microservice team. The assumptions made for the development of this research regarding the existence of the difficulty for companies to maintain the alignment between MSA and EAM in relation to IT governance, as well as the EAM aspects discussed and addressed in the context of this paper, could be confirmed. A machine-supported method to collect data and automatically update the model can be investigated as proposed by Bogner & Zimmermann (2016a).

We considered that this research achieved the objective and answered the research questions. However, a possible step that seems viable, is implementing some automation using the Archi Tool, connected to a GitHub repository with a plugin available on the Archi tool website. This combination enables joining different ArchiMate models in one. On the other hand, a way to collect data from API of APM tools would be developed, or perhaps a direct link to an APM dashboard of the relevant information concerning microservices EA would be recorded in the ArchiMate model. Lastly, the model proposed should be applied and evaluated in a real case, and other theoretical strategies can be investigated to enrich solutions.

Bibliography

Balakrushnan, S., Bell, J., Currier, B., Packard, H., Harrington, E. E., Helstrom, B., … Martins, M. (2016). *Microservices Architecture*. Retrieved from The Open Group website: https://publications.opengroup.org/white-papers/soa/w169

Balalaie, A., Heydarnoori, A., & Jamshidi, P. (2016). Migrating to Cloud-Native architectures using microservices: An experience report. *Communications in Computer and Information Science*. https://doi.org/10.1007/978-3-319-33313-7_15

Bischoff, S., Aier, S., & Winter, R. (2014). Use it or lose it? the role of pressure for use and utility of enterprise architecture artifacts. *Proceedings - 16th IEEE Conference on Business Informatics, CBI 2014*, *2*, 133–140. https://doi.org/10.1109/CBI.2014.56

Bogner, J., & Zimmermann, A. (2016a). Adaptable digital enterprise architecture with microservices. *10th Advanced Summer School on Service Oriented Computing*, 59–61. Retrieved from https://publikationen.reutlingen-university.de/frontdoor/index/index/docId/1384

Bogner, J., & Zimmermann, A. (2016b). Towards Integrating Microservices with Adaptable Enterprise Architecture. *Proceedings - IEEE International Enterprise Distributed Object Computing Workshop, EDOCW*. https://doi.org/10.1109/EDOCW.2016.7584392

Buchgeher, G., Winterer, M., Weinreich, R., Luger, J., Wingelhofer, R., & Aistleitner, M. (2017). Microservices in a small development organization: An industrial experience report. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. https://doi.org/10.1007/978-3-319-65831-5_15

Chen, W., Hess, C., Langermeier, M., Stuelpnagel, J., & Diefenthaler, P. (2013). Semantic Enterprise Architecture Management. *ICEIS2013-15thInternationalConferenceonEnterpriseInformationSystems*, 318–325. https://doi.org/10.5220/0004445003180325

COBIT. A Business Framework for the Governance and Management of Enterprise IT. (2012). Retrieved from http://www.isaca.org/cobit/pages/default.aspx

Di Francesco, P. (2017). Architecting microservices. *Proceedings - 2017 IEEE International Conference on Software Architecture Workshops, ICSAW 2017: Side Track Proceedings*, 224–229. https://doi.org/10.1109/ICSAW.2017.65

Dresch, A. (2015). *Design Science research: método de pesquisa para avanço da ciência e tecnologia* (Bookman, Ed.). Porto Alegre.

Drews, P., Schirmer, I., Horlach, B., & Tekaat, C. (2017). Bimodal Enterprise Architecture Management: The Emergence of a New EAM Function for a BizDevOps-Based Fast IT. *IEEE 21st International Enterprise Distributed Object Computing Workshop*

(EDOCW), 57–64. https://doi.org/10.1109/EDOCW.2017.18

Engel, T., Langermeier, M., Bauer, B., & Hofmann, A. (2018). Evaluation of microservice architectures: A metric and tool-based approach. *Lecture Notes in Business Information Processing*. https://doi.org/10.1007/978-3-319-92901-9_8

Fowler, M., & Lewis, J. (2014). Microservices - A definition of this new architectural term. Retrieved November 20, 2019, from ThoughtWorks website: https://martinfowler.com/articles/microservices.html

Francesco, P. Di, Malavolta, I., & Lago, P. (2017). Research on Architecting Microservices: Trends, Focus, and Potential for Industrial Adoption. *2017 IEEE International Conference on Software Architecture (ICSA)*, 21–30. https://doi.org/10.1109/ICSA.2017.24

Haselbock, S., & Weinreich, R. (2017). Decision guidance models for microservice monitoring. *Proceedings - 2017 IEEE International Conference on Software Architecture Workshops, ICSAW 2017: Side Track Proceedings*. https://doi.org/10.1109/ICSAW.2017.31

Haselböck, S., Weinreich, R., & Buchgeher, G. (2017). *Decision Models for Microservices: Design Areas, Stakeholders, Use Cases, and Requirements*. https://doi.org/10.1007/978-3-319-65831-5_11

Hassan, S., & Bahsoon, R. (2016). Microservices and Their Design Trade-Offs: A Self-Adaptive Roadmap. *2016 IEEE International Conference on Services Computing (SCC)*, 813–818. https://doi.org/10.1109/SCC.2016.113

Hevner, A. R., & Chatterjee, S. (2010). Design Research in Information Systems: Theory and Practice. In: Design Research in Information Systems. Integrated Series in Information Systems. In *Springer* (Vol. 22). https://doi.org/https://doi.org/10.1007/978-1-4419-5653-8_

ISO/IEC/IEEE. (2011). *Systems and Software Engineering - Architecture Description (Iso/Iec/Ieee 42010:2011)*.

Jonkers, H., Band, I., Quartel, D., & Lankhorst, M. (2017). ArchiSurance Case Study V2. *Henk Jonkers*. Retrieved from https://publications.opengroup.org/y163

Knoche, H., & Hasselbring, W. (2019). *Drivers and Barriers for Microservice Adoption-A Survey among Professionals in Germany 1 Drivers and Barriers for Microservice Adoption-A Survey among Professionals in Germany*. *14*(1). https://doi.org/10.18417/emisa.14.1

Lenarduzzi, V., & Sievi-Korte, O. (2018). On the negative impact of team independence in microservices software development. *2th International Workshop on Microservices: Agile and DevOps Experience (MADE18)*. Porto, Portugal: ACM Digital Library.

Mayer, B., & Weinreich, R. (2017). A dashboard for microservice monitoring and

management. *Proceedings - 2017 IEEE International Conference on Software Architecture Workshops, ICSAW 2017: Side Track Proceedings*. https://doi.org/10.1109/ICSAW.2017.44

Mettler, T., Eurich, M., & Winter, R. (2014). On the Use of Experiments in Design Science Research: A Proposition of an Evaluation Framework. *Communications of the Association for Information Systems*, *34*(1). https://doi.org/10.17705/1CAIS.03410

Newman, S. (2015). *Building microservices*. O'Reilly Media.

Peffers, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2007). A Design Science Research Methodology for Information Systems Research. *Journal of Management Information Systems*, *24*(3), 45–77. https://doi.org/10.2753/mis0742-1222240302

Ramos, H., & Vasconcelos, A. (2014). eXtreme enterprise architecture planning (ACM Press). https://doi.org/10.1145/2554850.2555130

Salah, T., Jamal Zemerly, M., Chan Yeob Yeun, Al-Qutayri, M., & Al-Hammadi, Y. (2016). The evolution of distributed systems towards microservices architecture. *2016 11th International Conference for Internet Technology and Secured Transactions (ICITST)*, 318–325. https://doi.org/10.1109/ICITST.2016.7856721

Singleton, A. (2016). The Economics of Microservices. *IEEE Cloud Computing*. https://doi.org/10.1109/MCC.2016.109

Soldani, J., Tamburri, D. A., & Van Den Heuvel, W.-J. J. (2018). The pains and gains of microservices: A Systematic grey literature review. *Journal of Systems and Software*, *146*, 215–232. https://doi.org/10.1016/j.jss.2018.09.082

The Open Group. (2017). ArchiMate® 3.0.1 Specification. *The Open Group Standard*. Retrieved from http://pubs.opengroup.org/architecture/archimate3-doc/

The Open Group. (2018). The TOGAF® Standard, Version 9.2. Retrieved February 16, 2019, from https://publications.opengroup.org/standards/togaf/c182

The SOA Source Book - Microservices Architecture. (2016). Retrieved November 20, 2018, from The Open Group website: http://www.opengroup.org/soa/source-book/msawp/index.htm

Thomas, A. (2018). *Hype Cycle for Application Architecture, 2018*. Retrieved from https://www.gartner.com/doc/3886164?ref=mrktg-srch

Weiss, S., Aier, S., & Winter, R. (2013). Institutionalization and the Effectiveness of Enterprise Architecture Management. In *ICIS 2013 Proceedings*. Retrieved from https://aisel.aisnet.org/icis2013/proceedings/GovernanceManagement/9

Wißotzki, M., Wismar, H., & Sonnenberger, A. (2013). *Adoption of Enterprise Architecture Management in Small and Medium Enterprises*. Retrieved from https://www.researchgate.net/publication/257936712

Yale Yu, Silveira, H., & Sundaram, M. (2016). A microservice based reference architecture model in the context of enterprise architecture. *2016 IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC)*, 1856–1860. https://doi.org/10.1109/IMCEC.2016.7867539

Zachman, J. A. (2016). The Concise Definition of The Zachman Framework by: John A. Zachman. Retrieved November 18, 2019, from Zachman International, Inc. website: https://www.zachman.com/about-the-zachman-framework

Zimmermann, A., Schmidt, R., Sandkuhl, K., Jugel, D., Bogner, J., & Mohring, M. (2017). Decision management for micro-granular digital architecture. *Proceedings - IEEE International Enterprise Distributed Object Computing Workshop, EDOCW*, *2017-Octob*, 29–38. https://doi.org/10.1109/EDOCW.2017.14

Zimmermann, A., Schmidt, R., Sandkuhl, K., Jugel, D., Bogner, J., Mohring, M., & Möhring, M. (2018). Evolution of Enterprise Architecture for Digital Transformation. *2018 IEEE 22nd International Enterprise Distributed Object Computing Workshop (EDOCW)*, 87–96. https://doi.org/10.1109/EDOCW.2018.00023

Zimmermann, O. (2017). Microservices tenets. *Computer Science - Research and Development*, *32*(3–4), 301–310. https://doi.org/10.1007/s00450-016-0337-0

Appendixes

Appendix A – Questionnaire

The role of EAM to Govern MSA adoption

Este questionário é parte de um trabalho acadêmico do curso de mestrado em informações e sistemas empresariais da UAB/IST de Portugal e visa confirmar os aspetos da arquitetura de microserviços relevantes para o gerenciamento da arquitetura corporativa e validar o modelo proposto para governação destes aspetos.

É estimado um tempo de 5 a 10 minutos para a conclusão deste questionário

Todas as informações aqui prestadas serão resguardadas, não sendo divulgadas nenhuma hipótese o nome do respondente e se prestam apenas a alimentar a pesquisa científica da qual é objeto. Caso necessário, o uso do nome da empresa só será utilizado após autorização expressa dos responsáveis pela empresa.

Empresa: _____ Nome:_____

Qual papel mais se assemelha ao teu na organização?

□ Gerente de TI	□ Arquiteto Corporativo	□ Arquiteto SOA, Microserviço ou Integração	□ Arquiteto de Software	□ Desenvolvedor SOA, Microserviços ou Aplicações

Quanto ao modelo apresentado, responda o quanto concorda com a avaliação dos riscos listados ao se adotar a arquitetura de microserviços (MSA).

Riscos Relacionados:

1. **Reduzir a flexibilidade/agilidade**: A diversidade totalmente descontrolada de tecnologias pode ter impactos negativos ao reduzir a flexibilidade e mobilidade de recursos humanos, faz com que se perca a agilidade e o "*time to market*".

   O quanto você concorda com o risco e a afirmação?

□ Discordo Totalmente	□ Discordo Parcialmente	□ Indiferente	□ Concordo Parcialmente	□ Concordo Totalmente

   O modelo apresentado reduz a probabilidade deste risco?

□ Discordo Totalmente	□ Discordo Parcialmente	□ Indiferente	□ Concordo Parcialmente	□ Concordo Totalmente

2. **Aumento no custo de treinamentos**: A diversidade descontrolada pode aumentar o custo de capacitação.

   O quanto você concorda com o risco e a afirmação?

□ Discordo Totalmente	□ Discordo Parcialmente	□ Indiferente	□ Concordo Parcialmente	□ Concordo Totalmente

   O modelo apresentado reduz a probabilidade deste risco?

□ Discordo Totalmente	□ Discordo Parcialmente	□ Indiferente	□ Concordo Parcialmente	□ Concordo Totalmente

3. **Aumento no custo de licenciamento**: A diversidade de tecnologias para o mesmo propósito pode aumentar o custo de licenças, o custo de uso de clouds e as dificuldade de gerenciá-las.

O quanto você concorda com o risco e a afirmação?

□ Discordo Totalmente	□ Discordo Parcialmente	□ Indiferente	□ Concordo Parcialmente	□ Concordo Totalmente

O modelo apresentado reduz a probabilidade deste risco?

□ Discordo Totalmente	□ Discordo Parcialmente	□ Indiferente	□ Concordo Parcialmente	□ Concordo Totalmente

4. **Aumento no custo de infraestrutura de TI.** A diversidade de nuvens pode causar um aumento ou descontrole no custo do uso de recursos de infraestrutura na nuvem.

O quanto você concorda com o risco e a afirmação?

□ Discordo Totalmente	□ Discordo Parcialmente	□ Indiferente	□ Concordo Parcialmente	□ Concordo Totalmente

O modelo apresentado reduz a probabilidade deste risco?

□ Discordo Totalmente	□ Discordo Parcialmente	□ Indiferente	□ Concordo Parcialmente	□ Concordo Totalmente

5. **Exposição à ausência de suporte**. Escolhas totalmente autônomas, pode levar a escolha de tecnologias pouco robustas em termos de garantia de continuidade com tecnologias que podem ser descontinuadas, abandonada pela comunidade ou trazendo dificuldade no recrutamento de pessoas com conhecimento na tecnologia ou com interesse em aprendê-la.

O quanto você concorda com o risco e a afirmação?

□ Discordo Totalmente	□ Discordo Parcialmente	□ Indiferente	□ Concordo Parcialmente	□ Concordo Totalmente

O modelo apresentado reduz a probabilidade deste risco

□ Discordo Totalmente	□ Discordo Parcialmente	□ Indiferente	□ Concordo Parcialmente	□ Concordo Totalmente

6. **Uso inapropriado de solução de TI**. A escolha autônoma da tecnologia pode levar a escolhas inapropriadas do ponto de vista da integração com as demais aplicações corporativas. Por exemplo, a utilização de um protocolo de comunicação cujo as demais aplicações da cia e parceiros não estão preparadas para integrar.

O quanto você concorda com o risco e a afirmação?

□ Discordo Totalmente	□ Discordo Parcialmente	□ Indiferente	□ Concordo Parcialmente	□ Concordo Totalmente

O modelo apresentado reduz a probabilidade deste risco

□ Discordo Totalmente	□ Discordo Parcialmente	□ Indiferente	□ Concordo Parcialmente	□ Concordo Totalmente

7. **Redução do nível de reuso dos serviços**. O reuso é um importante princípio e benefício do SOA. Uma má definição do escopo e da granularidade dos microserviços pode levar a serviços duplicados contribuindo para ineficiências na gestão de custo e na capacidade de TI.

O quanto você concorda com o risco e a afirmação?

□ Discordo Totalmente	□ Discordo Parcialmente	□ Indiferente	□ Concordo Parcialmente	□ Concordo Totalmente

O modelo apresentado reduz a probabilidade deste risco

□ Discordo Totalmente	□ Discordo Parcialmente	□ Indiferente	□ Concordo Parcialmente	□ Concordo Totalmente

8. **Não conformidade a políticas de segurança corporativas**. Quanto maior a diversidade de tecnologias para o mesmo propósito, maior a dificuldade de gerenciar e mitigar possíveis vulnerabilidades de segurança da informação.

   O quanto você concorda com o risco e a afirmação?

□ Discordo Totalmente	□ Discordo Parcialmente	□ Indiferente	□ Concordo Parcialmente	□ Concordo Totalmente

   O modelo apresentado reduz a probabilidade deste risco

□ Discordo Totalmente	□ Discordo Parcialmente	□ Indiferente	□ Concordo Parcialmente	□ Concordo Totalmente

9. **Não conformidade a normas de legais de regulação.** Por exemplo, uma norma de regulação pode determinar que os registros dos dados sejam imutáveis e guardadas por período determinado em determinado meio. Eg. Contabilidade, em que uma vez registrado o evento contábil este não pode mais ser alterado, apenas compensado.

   O quanto você concorda com o risco e a afirmação?

□ Discordo Totalmente	□ Discordo Parcialmente	□ Indiferente	□ Concordo Parcialmente	□ Concordo Totalmente

   O modelo apresentado reduz a probabilidade deste risco

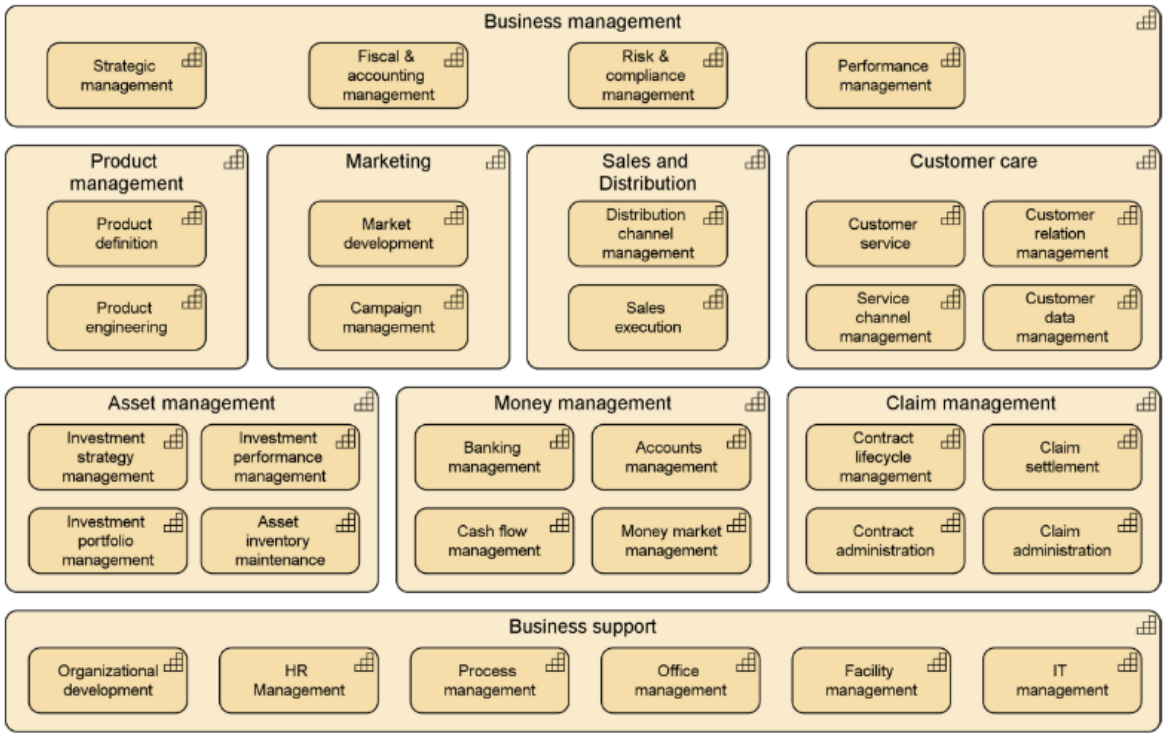
□ Discordo Totalmente	□ Discordo Parcialmente	□ Indiferente	□ Concordo Parcialmente	□ Concordo Totalmente

Comentários Livres sobre a percepção da utilidade da discussão e do modelo apresentado.

Annexes

Annex A – ArchiSurance Capability Map View

The capabilities are realized by the behavior of the organization, as shown in Figure 13. These capabilities need to be supported by the right resources.



Source (Jonkers et al., 2016, p.17)