# An agile enterprise architecture methodology for digital transformation

Department of Business Information Technology

Faculty of Electrical Engineering,

Mathematics and Computer Science

University of Twente

*Proponent:*
Suraj Visweswara
s.visweswara@student.utwente.nl

*Primary Research Supervisor:*
Dr. J.L. Rebelo Moreira

*Secondary Research Supervisor:*
Dr.IR. E.J.A. Folmer

*Company Supervisor:*
Dr. Adina Aldea

*Date:* August, 2023

## Acknowledgements

In 2021, I embarked on a journey to pursue a Master's study in the Netherlands and I can confidently say that the experience has been fruitful. I have grown both professionally and personally in the past 2 years. Today, I am more patient, resilient, and professionally more adept. As a part of the Master in Business Information Technology at the University of Twente, I have worked on my thesis at LeanIX, in Amsterdam from January till June of 2023.

As I reach the culmination of this transformative experience, I am deeply grateful to all those who have supported me throughout my academic journey the past 2 years. Their encouragement, guidance, and unwavering belief in my abilities have been instrumental in shaping this body of work. In expressing my heartfelt gratitude, I would like to acknowledge the following individuals:

I extend my gratitude to both of my supervisors at the university, Dr. J.L. Rebelo Moreira and Dr.IR. E.J.A. Folmer, for their invaluable guidance, insightful feedback, and patience. Their expertise and encouragement have been crucial in shaping the direction of this thesis. I am thankful for their constant support, which has helped me navigate through the challenges and complexities of this research.

I would also like to express my sincere appreciation to my mentor and friend at LeanIX, Dr. Adina Aldea. Your guidance, practical as well as academic insights, and industry knowledge have been instrumental in enriching my research and adding real-world relevance to this thesis. I am grateful for the opportunities provided by LeanIX, which allowed me to bridge the gap between theory and practice. To my colleagues in the solution engineering team at LeanIX and the Amsterdam office, I want to thank you for your support and encouragement. Your belief in my abilities has been a great source of motivation.

To my dear friends back home in India and the new friends I've made during my journey in the Netherlands, thank you for being you. Your presence, words of encouragement, occasional distractions and coffee breaks have kept me motivated and focused. Your unwavering support has made this experience not only academically enriching but also immensely enjoyable.

My heartfelt thanks also go to the faculty members, librarians, and administrative staff at University of Twente, whose tireless efforts have provided me with a conducive academic environment and access to valuable resources throughout my studies.

Most importantly, I want to thank my parents, my brother, my sister-in-law and my dear little niece for their unwavering love, support, and understanding throughout my academic pursuits. Their endless encouragement, sacrifices, and belief in me have been the driving force behind my accomplishments. I am forever indebted to them for instilling in me a strong work ethic and the determination to pursue my dreams fearlessly.

To all the individuals mentioned above and those unnamed but not forgotten, I extend my deepest gratitude.

## Abstract

In the field of enterprise architecture (EA), traditional frameworks have followed a linear, stepwise process akin to the waterfall model in software development, which limits flexibility once a phase is completed. This thesis identifies a gap in the literature regarding a methodology that aligns the principles of agile software development, as outlined in the Agile Manifesto, with the discipline of enterprise architecture. The primary research question addressed in this study is: "How to design a methodology that satisfies agile practices or principles so that companies can better transform digitally using enterprise architecture?" The Design Science Research Methodology (DSRM) guides the research process, leading to the development of the "Agile Enterprise Architecture Development Method" (AEA-DM) as the proposed methodology. By incorporating modified agile principles that address the limitations of the waterfall approach, it is hypothesized that merging these two disciplines can enhance digital transformation outcomes compared to current EA practices.

The study's main objective is to design a comprehensive and flexible methodology for constructing an agile enterprise architecture to facilitate digital transformation. To achieve this, the research questions are divided into three key objectives, all of which are successfully accomplished. The literature review focuses on exploring agile practices that benefit enterprise architecture management during digital transformation. Noteworthy practices identified include incremental development, collaboration, customer/stakeholder representation, continuous optimization, and test-driven design. These agile practices are integrated into the AEA-DM to ensure its alignment with agile principles.

The AEA-DM is specifically devised as an agile approach for developing enterprise architecture. A process flow diagram is used to provide a clear and structured representation of the methodology, while various team roles are assigned to facilitate agile testing, design of the EA, collaboration and demand management. The workflow encompasses steps such as demand collection, assessment, architecture change analysis, design of EA increment, and test-driven design. By following this methodology, organizations can effectively manage stakeholders' demands and foster continuous improvement within their enterprise architecture development process.

To evaluate the viability of the AEA-DM, a qualitative method, namely a thematic analysis is conducted on interview data obtained from experts in the enterprise architecture and digital transformation field. The evaluation aims to confirm the inclusion of all prerequisites and identify themes that validate the methodology. Valuable feedback received from the interviewees is incorporated into the final modified version of the AEA-DM, further enhancing its effectiveness for future implementation.

Therefore, this study provides valuable insights into the design of an agile enterprise architecture methodology and its potential to support digital transformation. By integrating agile practices and incorporating feedback from stakeholders, organizations can leverage enterprise architecture to optimize their IT landscape and drive successful digital transformation initiatives. The AEA-DM serves as a comprehensive and flexible methodology that enables enterprises to navigate the complexities of digital transformation in an agile and efficient manner. Further research and empirical validation of the AEA-DM, including quantitative analysis and additional interviews or observations, are recommended to strengthen its applicability and broaden its scope of implementation

*Keywords*: Agile Enterprise Architecture, Methodology, Digital Transformation, Qualitative review

# Contents

# List of Tables

# List of Figures

# Glossary

**AEA** Agile Enterprise Architecture

**AEAF** Agile Enterprise Architecture Framework

**AgEAMM** Agile Enterprise Architecture Modeling Method

**ASF** Agile Scaling Framework

**DS** Design Science

**DSRM** Design Science Research Methodology

**DT** Digital Transformation

**EA** Enterprise Architecture

**JIT** Just-in-Time

**LEAD** Lean Enterprise Architecture Development

**LEAF** Lean Enterprise Architecture Framework

**SAFE** Scaled Agile Framework

**SOA** Service Oriented Architecture

**TOGAF-ADM** The Open Group Architecture Framework - Architecture Development Method

# Chapter 1

# Introduction

In the field of enterprise architecture (EA), frameworks have historically followed a linear, stepwise process similar to the waterfall model of software development, which allows for limited flexibility once a phase is complete [68]. This thesis identifies the gap in literature of a methodology that aligns the principles of agile software development, as stated in the Agile Manifesto, with the discipline of Enterprise Architecture. Accordingly, this methodology is referred to as the "**Agile Enterprise Architecture Development Method**" or **AEA-DM** in subsequent sections of the document. By adopting modified agile principles, similar to those that addressed the limitations of waterfall development, it is hypothesized that the resulting merger of the two disciplines may lead to improved digital transformation outcomes for enterprises/firms compared to current EA practices.

## 1.1 Enterprise Architecture

The ISO/IEC/IEEE FDIS 42010 standard gives us the definition of an architecture in the software engineering and systems context:

> "Fundamental concepts or properties of an entity in its environment and governing principles for the realization and evolution of this entity and its related life cycle processes."[41]

Therefore, we can say that an architecture helps us to obtain an overall holistic view of a system/system of systems and ascertain its properties. The discipline of EA can be traced back to the 1960s, where the first EA framework was designed by John Zachmann [77]. It is important to note that an architecture can be built at different abstraction levels (eg: software architecture, business architecture etc.,). According to Gartner,

> "*Enterprise Architecture (EA)* is a discipline for proactively and holistically leading enterprise responses to disruptive forces by identifying and analyzing the execution of change toward desired business vision and outcomes. EA delivers value by presenting business and IT leaders with signature-ready recommendations for adjusting policies and projects to achieve targeted business outcomes that capitalize on relevant business disruptions"[27]

An EA captures both, business aspects (e.g., business processes, business objects) and IT aspects (e.g., interfaces, networks, devices) as well as their interrelations [16]. This gives us an overall view of the workings of an organization.

## 1.2 Agile Principles

In the field of software development, before the advent of the agile manifesto, practitioners were advocating for extensive planning, codified processes, and rigorous reuse to make development an efficient and predictable activity assuming that customer needs and requirements would stay the same throughout the development life-cycle [13]. However, this is not the case in the real world where customer needs are ever evolving. This led to the development of the Agile Manifesto in 2001 [11] which outlines 12 principles and 4 values of agile software development to deliver better products. These lean agile principles were derived from the Toyota Production System (TPS) where the primary aim of the TPS was to deliver value to customers by eliminating waste and increasing efficiency. The Toyota Production System (TPS) was derived from two concepts [46]:

- *"Jidoka"* which focuses on the social aspect of fully utilizing workers' capabilities
- *"Just-in-Time" (JIT)* concept which comprises of pull systems, one-piece flow, leveling.

Similarly, we can say that "Agility means to strip away as much of the heaviness, commonly associated with traditional software-development methodologies, as possible to promote quick response to changing environments, changes in user requirements, accelerated project deadlines, and the like." [24]. In essence, agile methods seem to embrace the idea of *uncertainty*, *changing requirements* and environments. These properties make it attractive for the domain of software development where these methodologies have proven to be effective [38].

## 1.3 Digital transformation

Digital transformation(DT) can help companies improve their strategic agility while enhancing the customer experience, simplifying operations, or creating new business models. DT has become a means for entrepreneurs to internalize external pressure as a driving force for change in the face of competitive and unpredictable external environments [78]. Based on the "2019 Digital Transformation Market Trends Report," the majority of respondents (78%) viewed digital transformation (DT) as essential for their organization's survival, with 24% ranking DT as their highest priority [78]. Digital transformation has many definitions, a comprehensive one by the Open group is:

> A strategy and an operating model change, in which technological advancements are leveraged to improve human experiences and operating efficiencies, and to evolve the products and services to which customers will remain loyal [69].

Digitalization is now known to be part of the fourth industrial revolution, which encompasses changes in information technology, automation, and the shift from offline to online services [4]. [14] define it as a consistent networking of all economic sectors and as adoption of new actors to new circumstances in the digital economy. The authors of [14] conclude that digital technologies such as cloud computing and mobile apps have transformative potential for businesses and that Enterprise architecture plays a vital role in communicating business plans and establishing a comprehensive framework, making it a suitable tool for selecting among multiple service offerings in the later stages of the Digital Transformation model.

## 1.4 Contribution to research

As mentioned in the introduction, agile methods and enterprise architecture are often looked at as opposites. In the realm of academia, the conventional approach to designing enterprise architecture has been likened to a waterfall process, while agile methods are often criticized for their perceived lack of emphasis on design. This thesis challenges this perception in trying to integrate the two domains.

Additionally, as mentioned in his book, *"The Agile Architecture Revolution"* [12], in a chapter titled, *"Can we do Agile Enterprise Architecture?"*, Mr. Jason Bloomberg suggests that the existing obstacles faced by enterprise architecture could potentially be overcome through the creation of an all-encompassing and efficient methodology. Such a methodology would act as a guiding framework, presenting explicit instructions for enterprises to follow in order to accomplish their objectives. However, even if this methodology were to be established, providing detailed step-by-step instructions for all members of the organization to adhere to, and even if the successful communication and adoption of these instructions were achieved across the entire enterprise—an endeavor that is undoubtedly arduous—the fundamental challenge of change would still persist. Despite its ambitious nature, this thesis aims to challenge the notion that a comprehensive methodology cannot be developed.

Overall, this study provides valuable insights into the design of an agile enterprise architecture methodology and its potential to support digital transformation. By incorporating agile practices and considering the feedback obtained from stakeholders, organizations can leverage enterprise architecture to optimize their IT landscape and drive successful digital transformation initiatives.

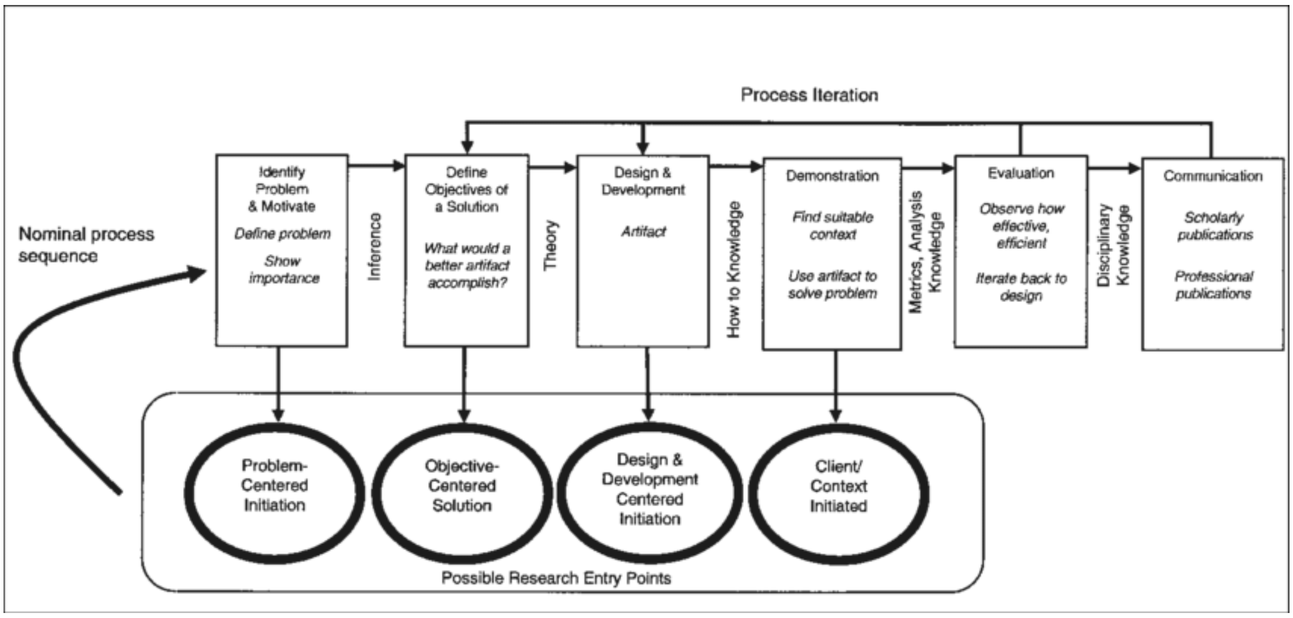
## 1.5 Document Structure

This thesis is divided into 6 chapters. Where the first one will focus on giving an introduction to the area of focus, the second goes into the design of the research and explanation of how the DSRM is applied in this thesis. The third lays out the design and execution process of the literature review and focuses on answering the first sub-research question through the means of explaining the results of the literature review. The fourth will endeavor to answer the second sub-research question through the means of the design of the IT artifact. The fifth chapter focuses on the evaluation of the IT artifact through the means of the qualitative data analysis of semi-structured interviews. And finally the thesis is concluded with a chapter focused on an updated methodology based off the results of the evaluation and remarks for future work.

Chapter 2

# Research Design

Since Enterprise Architecture is used to manage the complexity the arises from the development of a System-of-Systems in an organization, it follows that the study of EA is a part of the information systems domain. Hence, this research in the EA and information systems domain entails the use of Design Science since we can infer that the design of a methodology is first and foremost a design problem. According to the criteria set by [73], the Design Science Research methodology (DSRM) by Peffers et. al [56] was chosen for this thesis. Therefore, the steps laid out in the DSRM which can be seen in the figure 2.1 were followed to guide the research process.



Process Iteration
Nominal process sequence
Identify Problem & Motivate
Define problem
Show importance
Inference
Define Objectives of a Solution
What would a better artifact accomplish?
Theory
Design & Development
Artifact
How to Knowledge
Demonstration
Find suitable context
Use artifact to solve problem
Metrics, Analysis Knowledge
Evaluation
Observe how effective, efficient
Iterate back to design
Disciplinary Knowledge
Communication
Scholarly publications
Professional publications
Problem-Centered Initiation
Objective-Centered Solution
Design & Development Centered Initiation
Client/ Context Initiated
Possible Research Entry Points

FIGURE 2.1: Design Science Research Methodology from [56]

## 2.1 Problem identification

In the current information age, the continuous digital transformation of enterprises has become a crucial priority. In order to respond and adapt to the rapidly changing competition, demand, technology, and regulations, organizations need to be able to reconfigure their strategy, structure, processes, people, and technology quickly towards value-creating and value-protecting opportunities. However, research suggests that organization agility is challenging for many companies [60]. As a result, firms are increasingly turning to enterprise architects to lead their business approach to digital innovation, given its importance [57].

Mapping out the entire IT landscape of an organization and maintaining, using, and improving the information to support digital transformation typically takes a long time. Traditionally, enterprise architecture (EA) has been linked to the waterfall methodology of management, which was not a problem during the early days of EA. However, as organizations' IT landscapes become increasingly complex, and more and more enterprises employ SaaS solutions from different vendors for different purposes, this complexity will continue to grow as the IT landscape becomes more fragmented. Additionally, organizations need to transform quicker and in shorter cycles to keep up with the speed of innovation and competition, indicating that the traditional methodology of enterprise architecture may not be sufficient. The evolution of enterprise mobile IT and cloud computing has brought about new paradigms and requirements that must be considered by EA frameworks [15] [48].

According to [48], existing EA frameworks may not be suitable for achieving digital transformation. Thus, there is a need for EA frameworks to adapt to change and be more agile in addressing emerging trends and demands. Similar to the evolution of software development practices from following waterfall methodologies to a more agile form of development, the discipline of enterprise architecture could adopt similar ideas to enable a more agile approach.

## 2.2 Objectives

As mentioned in the previous sections, the main high level objective of this thesis is to design and develop a methodology for the development of an Agile Enterprise Architecture for the purposes of digital transformation of a firm/enterprise. By and large, the main concepts of this research are agile methodologies, enterprise architecture and the interactions between the two concepts.

### 2.2.1 Scope

The use of an Agile Enterprise Architecture in an enterprise comprises of many steps. It should be noted that the focus of the thesis is not on the scaling of agile throughout an organization directly. However, since Enterprise Architecture should play a role in the achievement of the overall strategic goal of an organization, it can be perceived that the key to scaling agile throughout an organization is through the effective use of EA. This thesis' primary objective is the design of an effective methodology that describes how to build/engineer an incremental agile enterprise architecture. The procedure for actual *adoption & deployment* of architecture increments is considered out-of-scope for this thesis. Furthermore, the decision of whether an organization is suitable for practicing such an agile EA is also considered out of scope.

### 2.2.2 Research questions

The main research question was designed using the guidelines outlined in [76] regarding the use of a problem context, IT artifact, stakeholder goals and requirements. Hence, the main research question can be summarized as:

> *How to design a methodology that satisfies agile practices or principles so that companies can better transform digitally using enterprise architecture?*

Where,

- **Methodology** is the *IT artifact*
- **Incorporating agile practices/principles** is the *requirement*
- **better digital transformation** is a *stakeholder goal*

- **Enterprise Architecture** is the *problem context*

This main research question was further subdivided into x amount of more sub-research questions. They are as follows:

Which was divided into the following sub-research questions

1. *What agile practices benefit enterprise architecture management in the digital transformation of an enterprise?*
2. *How can the design or engineering of an enterprise architecture be made agile?*
3. *Does the developed methodology provide a viable approach for designing and implementing enterprise architecture?*

As mentioned in [76], design science consists of 2 kinds of research concepts. Namely, design problems and knowledge questions. The knowledge questions are used to provide existing information about particular areas of interests. Consequently, before the design of an artifact, there was a need to obtain existing and relevant information about the domain of agile methods and their interaction with enterprise architecture. This led to the conduction of a systematic literature review as outlined by [45] combined with snowballing strategies and Grey literature. This literature review is aimed at answering the first sub-research question. Based on the background topics and the required knowledge, the formed research question was further divided into 4 other sub-research questions.

Based off of the main research question, sub-research questions are designed so that the problem can be broken down into smaller easily addressable chunks. Of which, the main RQ is a knowledge question:

> *What agile practices benefit enterprise architecture management in the digital transformation of an enterprise?*

Which was divided into the following knowledge questions:

1. *Which agile methodologies are mentioned in state of the art literature?*
2. *What are the benefits of agile methodologies over traditional waterfall style methods?*
3. *What are best procedures to incorporate agile principles into enterprise architecture management?*
4. *Which techniques/methodologies and practices of enterprise architecture management support the digital transformation of an enterprise?*

## 2.3 Design and Development

The third phase of the DSRM is the design and development of the IT artifact. The design of the artifact requires the answering the first knowledge question. This was achieved in the form of a systematic literature review combined with snowballing with 1 iteration and some Grey literature. The results of the literature review determine the design of the AEA Development Methodology which is the "*IT Artifact*". Therefore, the artifact design answers the second sub-research question, *"How can the design or engineering of an enterprise architecture be made agile?"*

### 2.3.1 Artifact requirements

To achieve the main research goal, a set of requirements that the artifact must fulfill have been identified. These requirements ensure that the created artifact (methodology) is applicable for practice, if not empirical testing at the minimum. These requirements are:

1. The methodology must address the question of pre-requisites
2. The methodology must describe the various roles and responsibilities mentioned in it
3. The methodology must have a construct that accounts for the agility aspects that will be highlighted in later sections

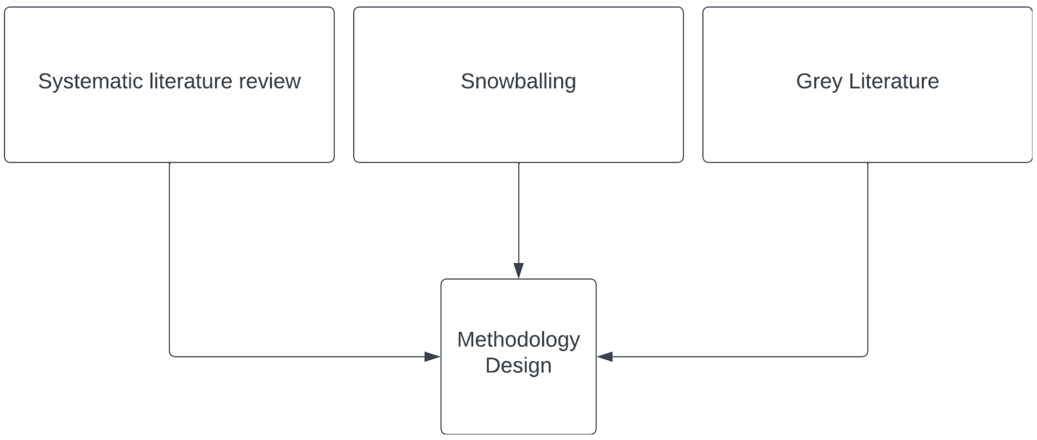
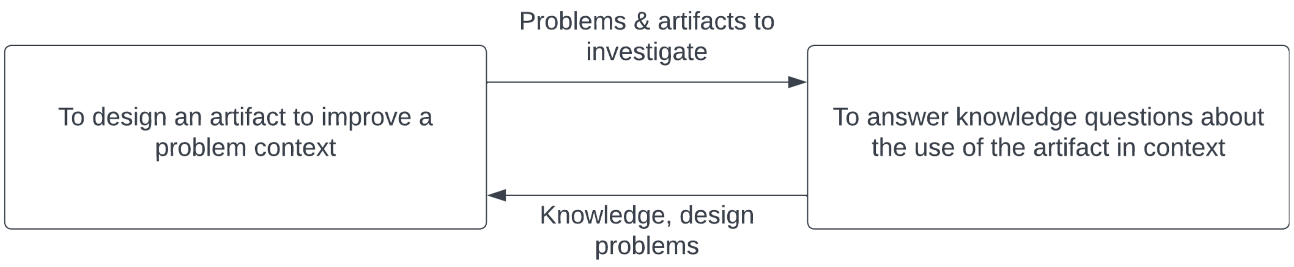


FIGURE 2.2: Development of the AEA-DM

## 2.4 Demonstration and Evaluation

Due to the inherent nature of the research conducted in this study and its context, the demonstration and evaluation stages are closely intertwined with one another. The main output of this research is the Agile Enterprise Architecture - Development Method. One of the best methods to validates the use of the AEA-DM empiricially, is for it to be used in an organization and obtain results in the form of case studies. However, this sort of validation will require a few months to a few years to show trustworthy results. Since that does not fit into the time frame of a master thesis, the chosen form of demonstration and evaluation of the use of this methodology will be done in the form of expert interviews. The methodology is first explained to the interviewees to help them understand the different constructs involved and also the scope of the methodology itself. And then a short semi-structured interview is conducted to verify the methodology itself and its short prototype that was created [redacted].

According to [76], there are 2 kinds of design science problems as shown in the figure 2.3. To compare these problems to the research conducted in this study, the first part, which is the design an artifact to improve a problem context is the part where the IT artifact, the AEA-DM is developed. The second part, which is to answer knowledge questions about the use of the artifact in context is dealt with in the demonstration and evaluation part of the DSRM. Therefore, the evaluation conducted answers the final sub-research question, *"Is the methodology a viable option to build an enterprise architecture in an enterprise?"*


Problems & artifacts to investigate
To design an artifact to improve a problem context
To answer knowledge questions about the use of the artifact in context
Knowledge, design problems
FIGURE 2.3: 2 kinds of DS Problems

## 2.5 Communication

This research and all of its findings will be published on essay.utwente.nl where the methodology and report can be freely downloaded after the conclusion of the researcher's Master thesis.

# Chapter 3

# Literature review

## 3.1 Review Methodology

### 3.1.1 Review Process

In order to ensure a thorough examination of the literature in the field, the review was divided into two phases. The first phase is the systematic literature review which was outlined in the previous sections. An overview of this can be seen in the figure 3.2. For the purposes of the systematic literature review, the guidelines prescribed by Barbara Kitchenham [45] were followed to avoid research bias for the below mentioned research questions. During the second phase of the review, additional studies were included to provide a broader context. This was achieved through the process of snowballing, where relevant studies were identified from the references of the initially selected papers. Furthermore, gray literature sources, such as company reports and industry publications, were also searched to gather additional information and insights.

In the first stage (SLR), the number of studies needed to be reviewed as a result of all the search queries was 211. This was later downsized by a simple title review which resulted in 79 studies. However, this was further downsized to 36 studies with a deeper abstract filter. Any studies that were found to be irrelevant to answering the posed research question were not investigated further. The final 36 studies were used as the starting set for the snowballing phase.

The second stage of the review process focuses on literature inclusion through snowballing and gray literature. This stage involves identifying and including studies that were mentioned as sources/references in other studies and that require further investigation. The main snowballing strategy is to look for relevant studies through the references of the studies examined at the end of the SLR. Hence, the filtering process begins with a title filter, followed by an abstract filter, and ultimately a full paper filter to determine the inclusion or exclusion of studies. In this study, only one iteration of snowballing was performed. Additionally, a web search was conducted to identify relevant information from gray literature sources. Figure 3.1 provides an overview of the entire review process.

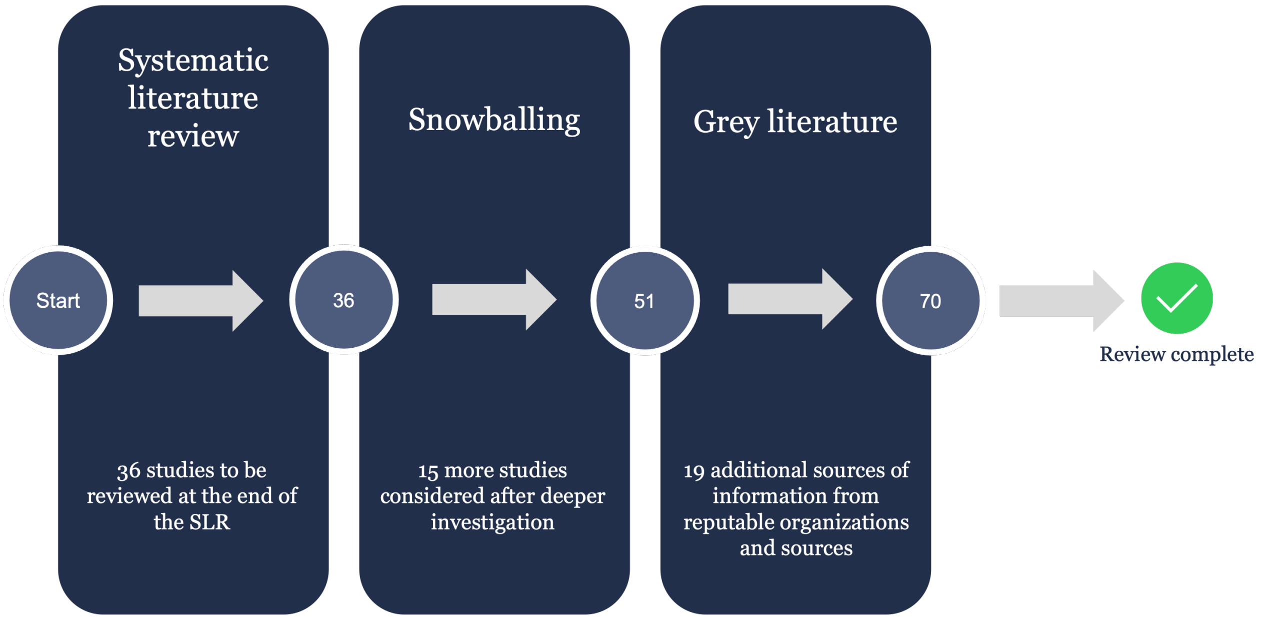
### 3.1.2 Research questions

As explained in the chapter Research Design, this literature review answers the main question:

> *What agile practices benefit enterprise architecture management in the digital transformation of an enterprise?*

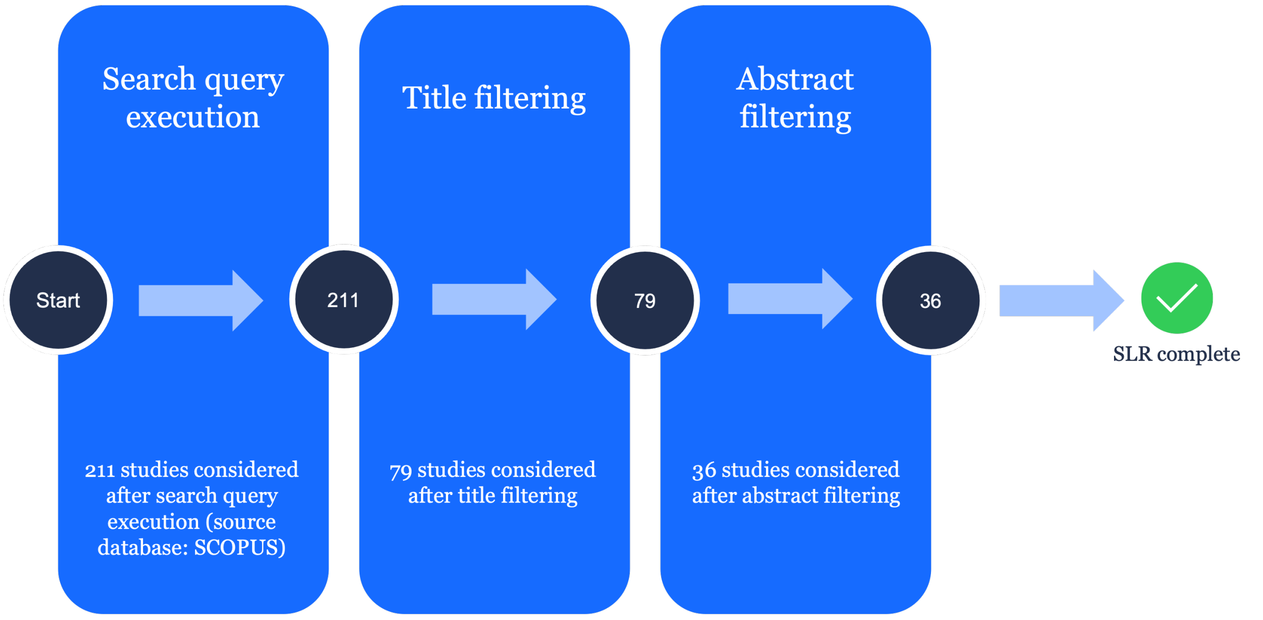
Which was divided into the following research questions

1. *Which agile methodologies are mentioned in state of the art literature?*
2. *What are the benefits of agile methodologies over traditional waterfall style methods?*
3. *What are best procedures to incorporate agile principles into enterprise architecture management?*
4. *Which techniques/methodologies and practices of enterprise architecture management support the digital transformation of an enterprise?*



Systematic literature review
Snowballing
Grey literature
Start
36
51
70
Review complete
36 studies to be reviewed at the end of the SLR
15 more studies considered after deeper investigation
19 additional sources of information from reputable organizations and sources

FIGURE 3.1: Overview of the overall review process



Search query execution
Title filtering
Abstract filtering
Start
211
79
36
SLR complete
211 studies considered after search query execution (source database: SCOPUS)
79 studies considered after title filtering
36 studies considered after abstract filtering

FIGURE 3.2: Overview of the systematic literature review

To understand why the research question was split into four sub questions, we must consider that there are four areas of interest. The first RQ is to gain knowledge of the different agile methodologies that are being used in different fields, the second is to understand the benefits of agile methods over waterfall methods and whether traditional waterfall practices must be considered to be maintained, enhanced or completely abandoned, the third is to understand how agile methodologies and enterprise architecture frameworks can be integrated (which is the main crux of this research). The final research question is formed to verify the connection between the domain of enterprise architecture and digital transformation projects. The idea is that if EA can act as an enabler to DT projects, so can Agile EA.

### 3.1.3 Search strategy

The main database source for the systematic literature review was the *"Scopus"* database, where appropriate keywords were used for the corresponding research question. The search queries were created using combinations ("OR" or "AND") of the keywords and limited to studies in English. The search queries for the specific research questions are as follows:

1. *Which agile methodologies are mentioned in state of the art literature?* - search conducted on 22/2/2023

   TITLE-ABS-KEY ( ( "agile principle" OR "agile method*" OR "agile manifesto" OR "agile framework" OR "agile approach" ) AND ( "systematic review" ) ) AND ( LIMIT-TO ( LANGUAGE , "English" ) )

2. *What are the benefits of agile methodologies over traditional waterfall style methods?* - search conducted on 27/2/2023

   TITLE-ABS-KEY ( waterfall AND ( "agile principle" OR "agile method*" OR "agile manifesto" OR "agile framework" OR "agile approach" ) AND ( benefits OR drawbacks ) ) AND ( LIMIT-TO ( LANGUAGE , "English" ) )

3. *What are best procedures to incorporate agile principles into enterprise architecture management?* - search conducted on 2/3/2023

   TITLE-ABS-KEY ( ( "agile principle" OR "agile method*" OR "agile manifesto" OR "agile framework" OR "agile approach" ) AND ( "enterprise architecture" OR "agile enterprise architecture" ) ) AND ( LIMIT-TO ( LANGUAGE , "English" ) )

4. *Which techniques/methodologies and practices of enterprise architecture management support the digital transformation of an enterprise?* - search conducted on 2/3/2023 and document type was limited to article considering the number of studies were nearly 280 which indicates that the field is quite mature.

   TITLE-ABS-KEY ( ( "Enterprise Architecture" OR "EA" OR "Enterprise architecture management" ) AND ( "digital transformation" OR "digital strategy" OR "Enterprise Strategy" ) ) AND ( LIMIT-TO ( DOCTYPE , "ar" ) ) AND ( LIMIT-TO ( LANGUAGE , "English" ) )

### 3.1.4 Inclusion and exclusion criteria

After executing the search query, results were first limited through the means of a title read-through and then further limited through a more thorough abstract read-through. This procedure can be seen in the figure 3.2. If studies were found to be irrelevant to the specific research question that it had to answer, then they were excluded. The exact inclusion and exclusion criteria are mentioned below.

Inclusion criteria:

1. Studies made in English
2. Studies available for download through the University of Twente
3. Studies that address the relevant research questions

Exclusion criteria:

1. Studies which titles have no similarity to the scope of study
2. Studies which were repeated in the form of conference proceedings
3. Studies with abstracts that indicate that they do not answer the relevant research question

## 3.2 Literature review findings

This chapter presents the findings and highlights the key insights from the systematic literature review that was outlined in Review Methodology and sets up the first step for further research and development towards an agile enterprise architecture methodology.

### 3.2.1 Agile methodologies

To address the first sub-sub-research question, the search query yielded 81 studies to be scrutinized. However, a more comprehensive analysis as outlined in the previous chapter Review Methodology led to the selection of only 10 studies for an in-depth review. The primary objective of the first research question was to examine the existence of secondary studies conducted on agile methodologies. This was crucial in identifying general agile practices or specific agile practices such as Scrum, XP, and Kanban. A previous study by [36] highlighted some trends of studies on agile methods in scientific databases. However, the study did not provide insights into how certain agile methods are implemented in organizations.

The initial approach to software development involved the Waterfall model, in which programmers would receive a list of customer requirements, match them up, and deliver the product. However, this model faced many issues, including frequent changes in customer requirements and uncertainty about what those requirements actually were. Programmers also faced their own problems, with a false sense of progress and incomplete coverage of the total project. As a result, there was a demand for a more iterative process with shorter development cycles [66]. During the 1990s and early 2000s, software engineering experts debated on how to organize software development in a way that would result in quicker, higher quality, and cost-effective solutions. Various proposals for improvement were put forward, including the standardization and measurement of software processes, as well as several specific tools, techniques, and practices [23]. One of these solutions for improvement were *"Agile Methodologies"*. In February 2001, a group of seventeen individuals came together to create the "Manifesto for Agile Software Development", which formally defined the principles and values of agile methodologies [52] [11]. Agile methods are seen as a response to plan-based or traditional approaches that prioritize a "rationalized, engineering-based" method where it's assumed that problems can be fully defined, and there is always an optimal and predictable solution for each issue. Those who follow traditional approaches are often believed to promote extensive planning, codified processes, and strict re-usability to make software development a reliable and efficient task [52].

The authors of [24] define agility as :

> Agility means to strip away as much of the heaviness, commonly associated with the traditional software-development methodologies, as possible to promote quick response to changing environments, changes in user requirements, accelerated project deadlines and the like.

Many attempts have been made to elucidate the fundamental principles of agile software development, often by drawing parallels with other fields. Primarily,

1. Agile manufacturing, which was introduced by researchers from Lehigh University in an attempt for the USA to regain its competitive position in manufacturing. Key concepts in agile manufacturing are integrating customer-supplier relationships, managing change, uncertainty, complexity, utilizing human resources, and information [23].

2. The principles of lean and just-in-time development which stem from the Toyota Production System that originated in the 1950s. These principles aim to eliminate waste, prioritize quality, and focus on problem-solving [46].

The main differences from [55] between the traditional methods and agile methods can be seen in the table 3.1.

Additionally there have been studies regarding AMFs (Agile Method Fragments) [25] [26]. The authors of [2] & [21] conducted a similar study which aimed to identify practices that can lead to

Table 3.1: Differences between the traditional methods and agile methods from [55]

| | **Traditional development** | **Agile development** |
|---|---|---|
| *Fundamental assumption* | Systems are fully specifiable, predictable, and are built through meticulous and extensive planning | High-quality adaptive software is developed by small teams using the principles of continuous design improvement and testing based on rapid feedback and change |
| *Management Style* | Command and control | Leadership and collaboration |
| *Knowledge Management* | Explicit | Tacit |
| *Communication* | Formal | Informal |
| *Development model* | Life-cycle(waterfall, spiral or some variation) | The evolutionary-delivery model |
| *Desired organizational structure* (Form/Structure) | Mechanistic, aimed at large organizations | Organic, aimed at small and medium enterprises |
| *Quality control* | Heavy planning and strict control. Late, heavy testing | Continuous control of requirements, design and solutions. Continuous testing. |

agility in software processes. [25] categorizes method fragments into two types: Process Fragments and Product Fragments. A fragment is a component of the agile software development method. As an example, "Pair Programming" is an AMF of XP methodology, which is a Process Fragment, while "Sprint Backlog" is a Product Fragment of Scrum [25].However, this study was not deeply delved into due to the inaccessibility of the agile method fragments. The authors of [2] identify a catalog of 17 agile practices after conducting a systematic review with relevant keywords. They are seen in the table 1 in the Tables The authors of [21] identify a 18 agile practices after conducting a systematic review with relevant keywords. However, both the studies identify mainly the same practices and the only differences being the *"Outcome review"*, *"Learning loop"*, *"Specification analysis"* & *"Progress monitoring"* practices mentioned in [21]. However, [21] does not give specific definitions to the agile practices mentioned.

### SCRUM

Scrum is an agile software development framework introduced in 1997 [63], and it is widely used in modern times [1] [87% of the respondents say they leverage SCRUM].

It divides development into iterations of up to four weeks (commonly around 2 weeks), called sprints, with a shippable product increment delivered to the user at the end of each sprint. In each new sprint, developers, in collaboration with other stakeholders, select tasks during a sprint-planning meeting [40].

Scrum includes a *Product Owner* to represent the customer and capture requirements in the form of user stories. These stories are aggregated in a prioritized *Product Backlog* (MoSCoW, RICE etc.,), which is a continuously updated document reflecting the current understanding of user needs. Scrum is designed for *small interdisciplinary teams* of about six to nine developers who self-organize to decide on strategies to achieve the sprint objectives. The *Scrum Master* coordinates the daily work and adherence to the Scrum process. *Daily stand-up meetings* maintain a quick pace of work, and *retrospectives* after each sprint facilitate learning and reflection on work practices [40].

### Extreme Programming (XP)

The methodology, known as *Extreme Programming (XP)*, was initially developed by Kent Beck, Ward Cunningham, and Ron Jeffries in 1996. Beck continued to improve the methodology and eventually authored the book "Extreme Programming Explained" in 1999 [10]. XP is a team-oriented methodology usually practiced in small teams with up to 20 members. All the developers in the team are jointly responsible for product delivery, rather than just the owner or boss. It includes multiple releases in a short time period and promotes effective communication by being located near each other, ideally in a single room. XP also encourages customer interaction by having a representative of the customer become an integral part of the team for effective communication [66]. [66] also highlights some of the differences between XP and SCRUM which can be viewed in the table 3.2. The information from [66] has some degree of difference from what is mentioned about Scrum from [40] regarding sprint timelines, but some teams may carry out sprints of up to 2 months. It is worth noting that the Scrum methodology appears to provide greater flexibility in the organization and execution of projects.

Table 3.2: Differences between Scrum and XP from [66]

| | **Extreme Programming** | **Scrum** |
|---|---|---|
| 1 | The team works for 1-2 weeks | The team works in iterative periods, known as sprints. Usually 1-2 month long sprints. |
| 2 | Has a flexible timeline | Does not allow any changes in timeline or guidelines |
| 3 | Emphasizes on Software Engineering methodologies | Emphasizes on self-organization |
| 4 | Team has to strictly follow the predetermined priority order | he team decided the order of the priority |
| 5 | Ready to apply without any changes | Not complete, will have to fill the framework with any other software engineering technique |

### KANBAN

The coverage of Kanban in literature is lacking. [23] conducted a systematic literature review up to 2005, and identified 36 relevant empirical studies on agile methods and lean software development. However, only one study reported on the application of lean practices in software development and they observed that majority of the studies tended to focus on a single development method. In [23] 76% [25 out of 33]of the studies investigated used XP. However, [7] is a study in which the authors conduct a systematic review to provide insight into the Kanban approach and its elements (concepts, principles, practices, techniques, and tools) that have been empirically reported by scholars and practitioners. They define it as:

> A set of concepts, principles, practices, techniques, and tools for managing the product development process with an emphasis on the continual delivery of value to customers, while promoting ongoing learning and continuous improvements.

However, this definition may be applied to any agile methodology and does not give any information as to how Kanban is different from other agile methods. But they identify 20 elements of Kanban which can be seen in the table 2 in the Tables

### 3.2.2 The benefits and challenges of waterfall methods

A search query which resulted in 57 studies was executed to answer the second sub-sub-research question. Following the steps described in the previous chapter Review Methodology, 5 studies were selected for an in-depth examination. The aim of this query was to compare traditional software development methodologies with agile methodologies, and to identify the advantages and disadvantages of each approach.

Software development is a complex process that involves various stages from the the inception of an idea to deployment. There are two main schools of thought in software development: the traditional sequential or the “Waterfall method” and the iterative or the “Agile method” [44].

The Waterfall method, a traditional sequential software development methodology also referred to as the "heavyweight" method, was initially introduced by Rolls Royce in 1970 [18]. It adheres to a linear and sequential approach, where each stage of the development process is completed before moving on to the next stage. The Waterfall approach is recognized for its inflexible structure that presupposes the identification of all requirements upfront and requires any changes to requirements or design to be managed through a change control process [71]. It is a structured method that follows a predefined process, comprising well-defined phases such as requirements gathering, design, development, testing, and deployment.

The iterative or the “Agile method” is a more flexible approach that emphasizes collaboration, test-driven design & adaptability [71]. The Agile method is based on the principle of breaking down development into smaller and more manageable tasks. These tasks are then prioritized and completed in short cycles, known as sprints, with regular feedback and testing to ensure that the product meets the customer’s requirements. The Agile method is more responsive to changing customer needs, and as such, it allows for more frequent changes and adaptations to the development process.

According to the authors of [18], where a system dynamics model is designed to study the differences between Waterfall methods and agile methods (specifically, Scrum and Lean Kanban), they highlight the *waterfall delay* aspect as change in requirements tend to cause delays and waterfall methods assume requirements do not change.

#### Waterfall vs Agile

In their work [6], the authors aim to verify the perceived benefits on using an agile approach, when applied in the context of systems development. They suggest using an agile project management approach in projects which involve uncertainty and constant change, where traditional techniques of project management do not fully meet the flexibility requirements to meet the requirement changes of the project. Accordingly, they highlight a classification chart of types of project and what methodology to use by [17] which can be seen in the figure 1 in the Figures. [6] also highlight the main differences between traditional and agile methods which is seen in the table 3.3.

[67] investigate and summarize the major limitations of both agile and plan-based methods through the means of a systematic review. This can be seen in the table 3.4

[71] investigate and summarize the perceived advantages of both agile and traditional methods through the means of an empirical survey in the context of project management. This can be seen by their perceived amount of importance in the table 3.5

## 3.3 Agility of an Enterprise

This section focuses on answering the third knowledge question, *"What are the best procedures to incorporate agile principles into enterprise architecture management?"* It is judged that the answer to this KQ would be in the from of a framework or methodology to integrate agile methodologies and enterprise architecture. In the end, the various frameworks that come up when the key words are queried are measured against the agility aspects highlighted in [47]

The authors of [47] recognize 3 main sources of enterprise agility. They are:

1. *Business Agility*: Effective enterprise agility begins with the top-level management of the organization. They prioritize promptly identifying changes in the business environment, quickly

Table 3.3: Agile vs Traditional methods (differences) from [6]

| | **Traditional** | **Agile** |
|---|---|---|
| *Project objectives* | The emphasis is on meeting the stipulated deadline, adhering to budgetary constraints, and achieving the desired level of quality | Emphasis is on achieving multiple success criteria and various business outcomes |
| *Project plan* | A set of planned activities carried out to satisfy the three main project constraints, namely time, cost, and quality | The organizational structure and processes put in place to attain the intended business outcomes and objectives |
| *Planning* | Conducted at the beginning of a project | Performed at the beginning and held whenever necessary |
| *Managerial approach* | The managerial approach tends to be inflexible, as it is highly focused on adhering to the initial plan without allowing for much deviation or adjustment | The managerial approach tends to be flexible, adaptable and variable |
| *Work/execution* | Predictable, measurable, linear, simple, straight-forward | Unpredictable and not measurable, non-linear, complex, looping |
| *Organizational influence* | Minimun, impartial, from the project kick-off. | Affect the project throughout its execution |
| *Project control* | Detect any discrepancies from the initial plan and take corrective actions to adhere to it | Recognize alterations in the surrounding circumstances and modify the plan accordingly |
| *Methodology application* | Application of a standardized process for all projects without considering the project-specific requirements | Adapting the process methodology based on the nature and characteristics of each project |
| *Management style* | One model is utilized for all types of projects | An adaptive approach is necessary as a single model cannot cater to all types of projects |

responding to them, and ensuring valuable delivery.

2. *Process Agility*: Processes must be evaluated on their focus on people, value delivery, and responsiveness to change. They must also be adaptable to changing circumstances and allow for modifications to the development process in response to changes in goals, environment, or project

Table 3.4: Agile vs Traditional methods (limitations) from [67]

| | **Traditional** | **Agile** |
|---|---|---|
| 1 | Not robust enough | Unsuitable for distributed development environment |
| 2 | Project isolated environment | Unsuitable for development with many number of teams |
| 3 | Time constraints may not align with what is possible | Is limited to software with low criticality or not suitable for large scale software |
| 4 | Uncertainty in setting goals | More complex development compared to traditional methods |
| 5 | Lack of customer representation | Insufficient focus on software design |
| 6 | Testing is insufficient | Lack of documentation can be a problem |
| 7 | Lack of adaptability to changing requirements | Requires members with more skill and ownership capabilities |

outcomes.

3. *System Agility*: Systems, including technical and organizational, easily changeable. Business knowledge must be accessible and transparent. New **architectures** are necessary for describing and implementing business knowledge using models, such as those for business processes or rules.

This study mainly focuses on the area of system agility since enterprise architecture can be most effectively utilized here. In response to the third research question, the search query yielded 62 studies, of which 16 were selected for further analysis after careful consideration.

Table 3.5: Agile vs Traditional methods (advantages) from [71]

| | **Traditional** | **Agile** |
|---|---|---|
| 1 | Fixed roles and processes with clear responsibilities | Fast recognition of changed requirements |
| 2 | Stable, systematic and documented planning | Fast identification of errors due to short development cycles |
| 3 | Predictive capacities of team members | Ability to react flexible and quick to changing requirements |
| 4 | Measurability of project progress via milestones | Lower risk of false developments |
| 5 | Content dependencies can be taken into account from the start | The continuous optimization of the project processes |
| 6 | Reliable estimation of time and budget | High motivation of team through personal responsibility |
| 7 | Option to achieve high efficiency by planning | No limitation in the process of finding a solution |

### 3.3.1 Aspects of system agility

In [47], the authors outline 5 aspects of system agility that need to be considered when building an agile architecture framework. Additional to the 5 aspects, 2 more aspects (model-based development & service-oriented architecture (SOA)) were considered based on the frequency of its appearance in relevant studies. These aspects will be the basis for evaluation of the different frameworks presented in the literature. These aspects include:

- **Making changes**: Research findings suggest that organizations typically undergo major changes approximately once every three years, while smaller changes occur on an almost continuous basis [42] [43]. These changes may be induced by regulatory requirements, customer demand changes or maybe because of the development of new technologies (example: advent of AI based chat bots in recent times such as chatGPT). Agile methodologies are acknowledged as a viable approach for organizations to respond to these stimuli. The architectural principles from [32] that apply here are:

    1. *Separation of concerns* - using strategies such as layering and modularity to keep changes as local as possible. This maintains that the functionality of the broader system is minimally affected.
    2. *Low coupling and high cohesion* - Low number of relations between subsystems and a high internal cohesion of each subsystem help with overall understanding of the system and similar to what is mentioned above, keeps changes local and maintains functionality of the broader system
    3. *Encapsulation* - If a system has well-defined interfaces and prevents the external environment from depending on its internal implementation, it allows for changes to be made to the implementation without impacting the external environment.
    4. *Team structure* - The authors of [47] also identify team structure as an aspect of system modularity. They say that effective communication between the designers and implementers of each system element is necessary for ensuring proper connectivity between the elements.

- **Deploying changes**: Before deploying a system or its components, factors such as *learnability*, *testability*, *installability* and *manageability* should be taken into consideration. This is particularly important in the iterative development process that agile methodologies promote. For a system to be deployed,

    1. Users must be able to learn and use the new system effectively & *Test-driven design* is an important agile practice and therefore, the system must be tested before it is deployed. This may be accomplished easily with pilot testing of a system with a small user group or so.
    2. The system should be install-able relatively easily, or in a hassle-free manner.
    3. The agility of a system can be negatively impacted if it requires significant management effort.

- **Dealing with the effects of changes**: Agile methodologies advocate for customer/stakeholder feedback. In cases where the changes are negatively impacting stakeholders, such as when bugs are found, it would be helpful if the system architecture incorporated for *redundancy* and *recovery.*

- **Integration**: Nowadays companies are creating new systems instead of using existing ones that have the same function. This is expensive and makes things more complex [49]. For rapid deployment and use of a system, it is crucial for the system to be *interoperable* and compliant with relevant standards. A system which exhibits high interoperability can connect with other systems in a seamless manner which in-turn promotes agility in its operations. "The main reason for not being able to reuse existing systems and applications is due to a lack of knowledge about them. This is often caused by not working enough with architecture and not having a good overall understanding of the IT landscape" [49]. Therefore, utilization and re-utilization (*reusability*) of standardized components greatly streamlines the rapid development of new solutions by leveraging existing building blocks.

- **Decoupling**: As mentioned earlier, a change in a part of the system should not affect or influence another part of the system. We could also call this *low dependence*. A system with low number of dependencies is considered to be more agile than a system with a high number of dependencies. Another advantage of decoupling is modularity. Modularity of a system may help with *reuse*. Creating a component that is independent and reusable necessitates considering the potential contexts in which the component may be utilized in the future.

- **Model-based development**: In [58], the author highlights the effectiveness of visually depicting EA in a *single graphical form* to iteratively communicate with senior management. The authors of [42] propose visualizing the architectural processes and deliverables as a key facilitator for applying agile methodologies to EA frameworks and practices.

- **Service-Oriented Architecture (SOA)**: There are multiple sources that seem to support the idea of using a service-oriented architecture in the development of an EA [42] [33] [22] [47] [12]. First used in the context of software architecture, it has since been adopted into TOGAF as well [33]. According to IBM, SOA is an integration architectural style and an enterprise-wide concept. It enables existing applications to be exposed over loosely-coupled interfaces, each corresponding to a business function, that enables applications in one part of an extended enterprise to reuse functionality in other applications [75]. By definition, SOA seems to complement & support *integration* and *decoupling*.

### 3.3.2 Current state of AEA

A significant outcome of adopting the mentioned EA frameworks based on the waterfall methodology is their failure to cater to the agility needs of modern businesses. As a result, firms lack the ability to adapt to change. Therefore, there is a need to address this issue. In very simple words, agile EA is the process for managing enterprise architecture modeling and redesign efforts with principles of agile methods [72]. However, this can be challenging because of the contradictory nature of the domains (agile methodologies and EA). EAM frameworks usually focus on achieving a long-term vision for the organization or a specific business case, whereas agile practices prioritize the incorporation of ongoing project findings into the development process on the short-term. This may be one of the reasons why there aren't many studies (as evidenced in the literature review) in the combining agile methods with EAM. As explained in the Aspects of system agility, AEA must incorporate principles from agile methodologies [11], including lean thinking and iterative approaches. Additionally, to succeed in implementing agile enterprise architecture, it is important to recognize that people are an essential part of the system, rather than just passive users [39].

However, industry experts are not blind to the possibilities of employing an AEA in their firms/enterprises. As evidenced by the mentioned recent McKinsey survey, firms are keen on being agile throughout their organization, ie., organizational agility is the need of the hour [60]. Many firms are deploying agile-at-scale frameworks such as Scaled Agile Framework® (SAFe®), Large-Scale Scrum (LeSS™) or the Spotify® Model [9] [1]. There have been studies which show that AEA has been investigated in the automotive industry [30] and in the software industry [62]. In light of this, several researchers have attempted to develop an Agile Enterprise Architecture (AEA) framework or methodology. Allen Brown, the former president and CEO of the Open Group supported the idea of adopting agile principles into an enterprise architecture in order to account for changing business requirements [42]. Accordingly, the 10th edition of TOGAF (The Open Group Architecture Framework) concentrates also on supporting organizational agility. It includes a new section about enabling enterprise agility and applying the TOGAF-ADM using agile sprints similar to the SCRUM method [33].

The subsequent subsections will introduce and further compare the noteworthy frameworks that are available in body of literature considered for this study with the agility aspects introduced in Aspects of system agility. One could argue that these aspects mainly just address best practices of developing an agile architecture and not optimal as evaluation metrics. However, it is important to note that whatever framework developed in a study, it must be able to facilitate in the development of an EA using the aspects mentioned.

### 3.3.3 Agile Enterprise Architecture Framework (AEAF)

The first mention of an agile EA framework can be traced back to [59]. In this framework, the authors lay out a framework for enterprise requirements using agile methods and practices. The framework consists of 7 constructs and 11 interactions between the said constructs. This can be seen in the figure 3.3. Each construct represents a model. Furthermore, the definition of each construct can be viewed in the table 3 and their relations are described in the table 4, both of which are available to view in the Tables

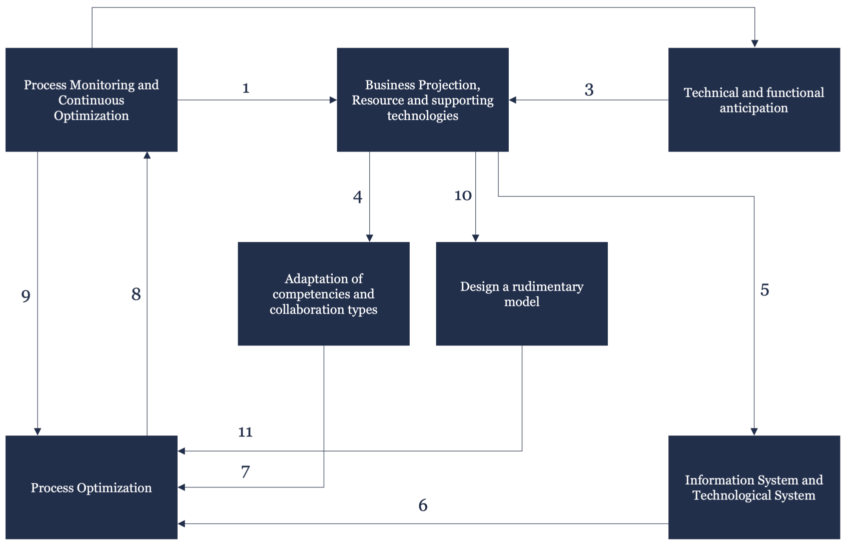

Process Monitoring and Continuous Optimization
1
Business Projection, Resource and supporting technologies
3
Technical and functional anticipation
4
10
Adaptation of competencies and collaboration types
Design a rudimentary model
9
8
5
11
7
Process Optimization
6
Information System and Technological System

FIGURE 3.3: Agile Enterprise Architecture Framework by [59]

When these constructs are seen side-to-side with the agility aspects mentioned in section Aspects of system agility, we see that the AEAF addresses quite a few of these aspects.

1. Making Changes
    - Separation of concerns - **Addressed** through use of various models, 5 viewpoints, 6 enterprise aspects and 4 classes of requirements. Although, this type of concern separation is not typical, it can be effective.
    - Low coupling and high cohesion - **Not addressed or mentioned**
    - Encapsulation - **Not addressed or mentioned**
    - Team Structure - **Not addressed or mentioned**
2. Deploying changes - **Addressed** in the form of a few constructs, namely Technical and Functional Anticipation, Process Monitoring and continuous optimization and Business Projection, Requirements and Supporting Technologies. However, usability is not addressed adequately.
3. Dealing with effects of changes - Redundacy is **addressed** in the form of the Process Monitoring and continuous optimization, Adaptation of competencies and collaboration types constructs & the Business Projection, Requirements and Supporting Technologies constructs. Reusability of systems is **addressed** through the The Business projection, Requirements and supporting technologies construct. It allows for reuse of existing systems and efficient resource allocation. However, the aspect of recovery has **not been adequately addressed**.
4. Integration - The inter-operability of said systems is **not addressed** adequately.
5. Decoupling - **Not addressed or mentioned**. Re-use of system elements not mentioned either.
6. Model-based development - **Addressed** in the form of all constructs and specifically the Rudimentary model to communicate with stakeholders.
7. SOA - **Not addressed or mentioned**

### 3.3.4 TOGAF-ADM with SCRUM

SCRUM being the most popular agile method used in modern times, and TOGAF being the standard for EA, there have been multiple sources which have tried to combine the 2 methods for the development of an AEA [15] [35] [5]. There is also the Agile Enterprise Architecture Framework by [61], which divides the different phases of the TOGAF-ADM into four "dimensions", namely Architecture Foundations, Architecture Delivery, Transformation Planning and Architecture Governance. But the usage of agile methods were only limited to the Architecture Governance dimension and not part of the delivery or planning dimensions. Therefore, this study also did not meet the criteria for a deeper dive as I felt the integration of agile methods and EA development was lacking. The most well designed and comprehensively built methodology is by the authors of [35]. This is because the team structure has been extensively modified from the common SCRUM team structure to a team structure suitable for EA through the inclusion of architecture experts, the *architecture product backlog* and the *impediment backlog*. In the [15] study, the authors suggest to comprise the team of people with an *interdisciplinary* background which is a good idea, but do not include the more extensive backlog constructs. The article by the open group in the TOGAF series guides [5] gives us a step by step idea of how to use TOGAF in sprints, but does not specify it to the detail of either [15] or [35]. For the purposes of this literature review, we take a look at the more complete TOGAF-ADM and SCRUM integration by [35].

The authors of [35], try to answer the questions of whether & how agile methods such as SCRUM can be used to create architectural deliverables and also how enterprise architects can collaborate with agile software development teams. For the purposes of this literature review, we are interested in the answer to the first question. They develop an integration between the TOGAF-ADM and SCRUM based on the design science research process. They acknowledge the significance of facilitating the definition of long-term goals and coordinated development of the IT landscape, all the while aligning with the business strategy and overarching goals of the enterprise.

They address the first question by developing a procedural model that is applied at the Enterprise strategic and segment architecture level of TOGAF, allowing for the creation of Enterprise Architecture through SCRUM projects. They emphasize the importance of *collaboration* between the implementation teams (called agile teams/development teams in the study) and the enterprise architects.

They argue that the integration should leverage the knowledge and expertise of the teams and should leave as many decisions as possible to them. The teams should therefore be empowered to define solutions on their own, which have to consider the limitations set by the current and future IT landscape. However, this approach mentioned in the study is limited to firms where software is built in-house. They suggest to leverage the knowledge of the development teams. The authors consider a situation where after every project built, the project must be integrated into the overall EA. These types of changes are important. However, in-house built applications are not the only types of applications used in an organization or an EA. And one could argue that leaving the definition of the solutions to the development teams may be beyond the scope of their expertise. Development teams do not understand the business needs of an application, and whether there are alternatives to building an application in-house. The overview of the integration can be seen in the figure 3.4.

When we compare the framework seen in the figure 3.4, we find a few relations to the agility aspects discussed in subsection 3.3.1. They can be summarized as:

1. Making Changes
    - Separation of concerns - **Addressed** through use of TOGAF
    - Low coupling and high cohesion - **Not addressed or mentioned**
    - Encapsulation - We can assume that systems will be designed to have interfaces as TOGAF advocates for it [33]. Therefore, **addressed**.
    - Team Structure - **Addressed extensively** in the form of Architecture development, implementation teams with dedicated Product owners and Enterprise architecture experts.
2. Deploying changes - Install-ability and usability is **addressed** in the form of SCRUM cycles. At the completion of a sprint, the scrum team receives feedback from customers and/or end-users

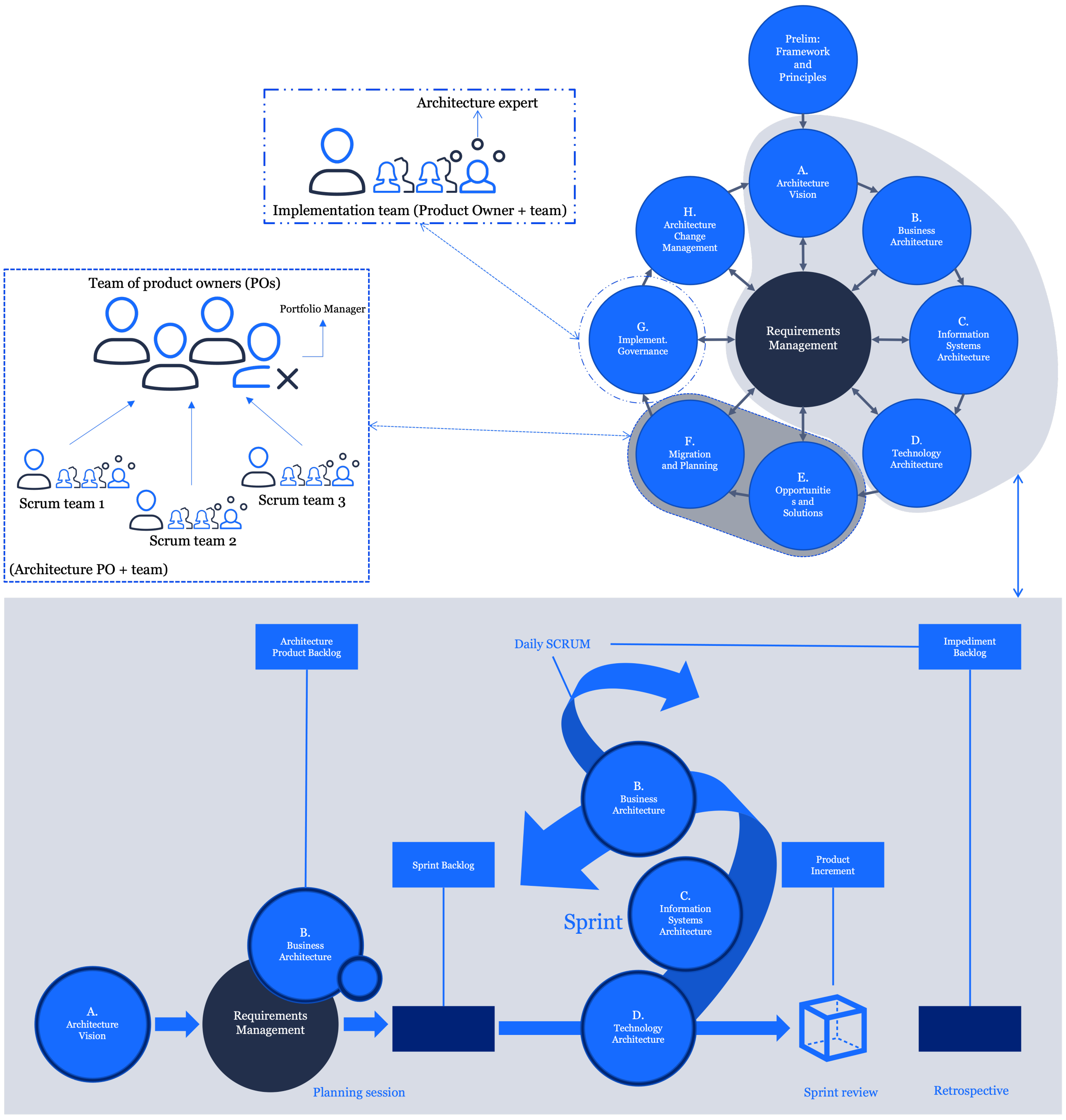


FIGURE 3.4: Overview of the integration between TOGAF-ADM and SCRUM described in [35]

through User Acceptance Testing. This feedback is used as input for the subsequent sprint in the form of the backlog.

3. Dealing with effects of changes - **Not addressed adequately** because there is no mention of the consideration for redundancy or fault tolerant design of the architecture to enhance recovery.
4. Integration - Extensive mention of integration between SCRUM and TOGAF, but **no mention** *interoperability* of systems and *re-usability* of resources/systems.
5. Decoupling - **Not addressed or mentioned**. Re-use of system elements not mentioned either.
6. Model-based development - **Addressed** in the form of EA deliverables and usage of TOGAF.
7. SOA - **Not addressed or mentioned**

### 3.3.5 SCRUM based AEA approaches

There are additionally 2 more SCRUM based AEA approaches that maybe examined from the literature. They are:

1. The Agile Enterprise Architecture Modeling Method (AgEAMM) mentioned in [49] is an AEA modeling method rooted in the SCRUM agile method. This methodology consists of 9 activities and 8 artifacts. However, the details of how the actual modeling takes place left a lot to be desired. Except for the creation of BPMN and ArchiMate models in the modeling session, there are no details of a different methodology to modeling. Similar to the other methods mentioned above, there is use of a "EA Product Backlog" which is used to account for changing EA requirements, the definition of an EA Committee and the task of performing a Sprint review and Retrospective to continuously improve. Since there is not much else to discuss about the AgEAMM, it has not been extensively discussed in this review.

2. The Adaptive Enterprise Architecture model by [20]. The authors propose moving an enterprise from one stage to the next (called an "elementary transition") by treating each transition as a project with the goal of closing the gap between the current state and the desired future state. This ensures continuous progress. The authors suggest having a committee of owners made up of an architecture owner, a business owner, and an application owner where the business and application/IT owners each have a team that includes a scrum master and members responsible for developing and engineering solutions. As for the artifacts, for the business and IT layers, they are the same as in the Scrum framework, such as the “Business Backlog,” “Application/IT Backlog,” “Sprint Backlog,” and “Burn-down chart”. However, on the architecture level, additional artifacts are needed, such as a matrix that describes the current and desired state of the enterprise, including its vision, mission, values, and goals, a chart of KPIs to monitor the transition from the As-Is to the To-Be architecture sand finally a Architecture Backlog that that lists prioritized features at the architecture level. This framework lists out more artifacts than the previous one without any empirical evidence of their usefulness. Therefore, it has also not been extensively discussed in this review.

### 3.3.6 LEAD - Lean Enterprise Architecture Development

The concept described in [39] is based on the Lean Enterprise Architecture Framework (LEAF) [19], which provides guidance for operational development. The structure of the Lean Enterprise Architecture Framework (LEAF) is presented in Figure 3.5, which comprises three main components, namely Management, Value Delivery Chain, and Architecture Landscape. The Management component is responsible for directing enterprise growth and driving change initiatives. The Value Delivery Chain is the module responsible for transforming customer needs/demands into services and other development goals. Finally, the Architecture Landscape is where the EA content resides, and all the artifacts are generated in a *just-in-time* manner.

The LEAF is a framework provides a content metamodel and placeholders for typical EA elements, and the LEAD operating model from the figure 3.7 helps implement it [39]. The LEAD approach from the figure 3.6, a customer-centric view of the enterprise, integrates the organization’s capabilities around the value stream model, instead of function- or process-based approaches. At the heart of the LEAD approach is developing value-adding services within the existing organizational environment and getting feedback from users. The authors claim that compared to traditional EA development approaches, which are time-consuming and resource-intensive, LEAD is lightweight and agile. The LEAD approach continuously produces and delivers EA content to the Architecture Landscape, while portfolios and road-maps can be adjusted according to changing conditions on an ongoing basis [39]. In the LEAD approach, there is no need to maintain separate as-is and to-be architectures, which saves time and increases efficiency in architecture modeling. However, if necessary, only certain development targets can be visualized in separate as-is and to-be views. The authors emphasize the importance of using a visualization tool to manage the Architecture Landscape, including all the EA content like the organization’s services, processes, and applications. They claim that the LEAD approach enables faster

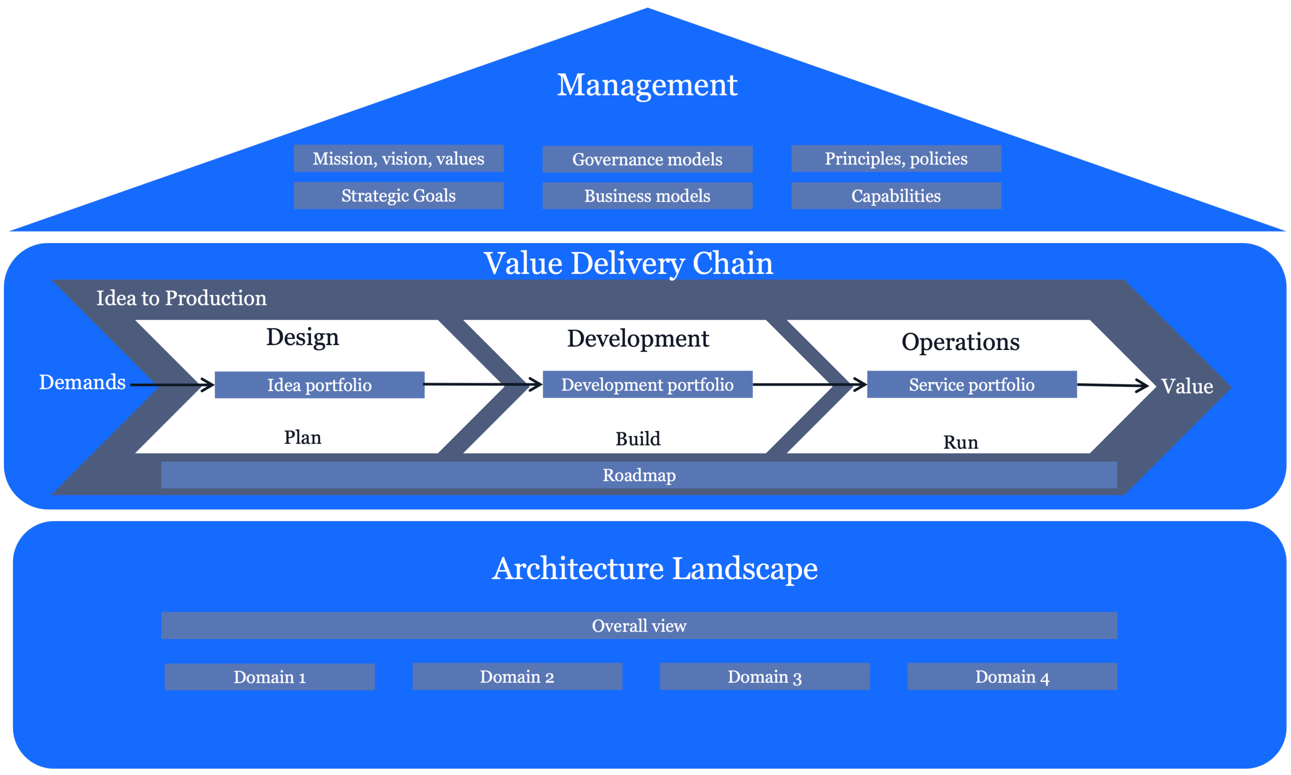


FIGURE 3.5: The Lean Enterprise Architecture Framework

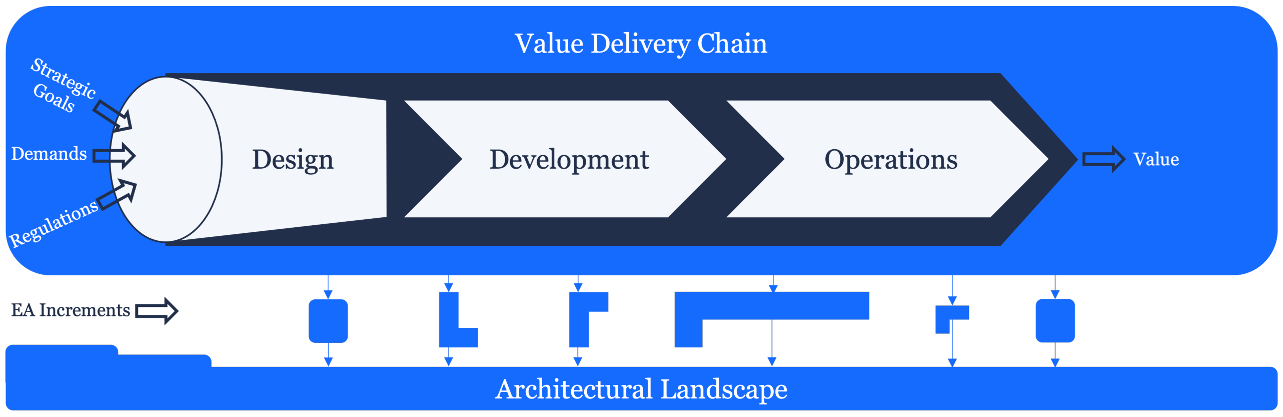


FIGURE 3.6: The LEAD Approach

development cycles, shorter time-to-market, better reactivity, and higher productivity compared to traditional approaches. But, the empirical validation of this notion has not been sufficiently established across a wide range of cases.

According to the LEAD framework, a cross-functional team consisting of specialists from customer relationship management, operational development, and enterprise architecture management is formed to manage customer demands during the design phase. This team employs agile methods and tools to identify the optimal solution for the customer. The subsequent development phase is overseen by the project management office (PMO) to ensure that everything is on schedule. In this phase, services or business activities are designed if required. After the development phase, the service management office (SMO) takes over in the operations phase, responsible for managing the project's production capabilities and ensuring smooth functioning. LEAD further supports portfolio management, which encompasses monitoring various concepts, development, IT services, and applications within the LEAF framework. The key objective is to maintain a streamlined and adaptable methodology that can be modified as per the requirement. To achieve this, a Lean Manager is appointed to supervise the entire value chain continuously, striving to enhance operational processes. Within the LEAD

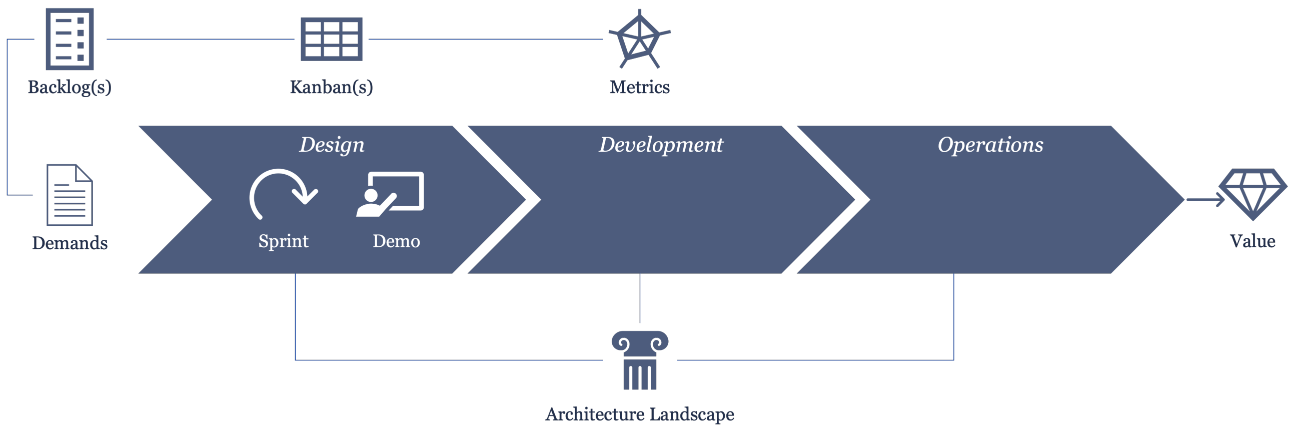


FIGURE 3.7: The LEAD operating model at high-level

framework, architects are responsible for actively participating in the development procedures and providing support whenever necessary.

When comparing LEAD to agility aspects, several relationships can be identified:

1. Making Changes
    - Separation of concerns - **Addressed** through use of LEAF in the form of Management, value delivery chain and the Architecture Landscape which is further separated by domains (similar to ArchiMate viewpoints)
    - Low coupling and high cohesion - **Not addressed or mentioned**
    - Encapsulation - **Not addressed or mentioned**. The term "solution vision" is mentioned which refers to a plan or design for one or more software applications that is created in response to current business requirements. This plan involves breaking down the business initiative into smaller, logical components or modules, and identifying their capabilities and how they interact with each other. In other words, it is a way of visualizing how the software application will work to meet the needs of the business. This approach helps to ensure that the software development process is aligned with the goals of the business and that the final product will be effective in addressing the identified business requirements.
    - Team Structure - **Addressed adequately** with the use of the "Lean Manager", "Demand Management Team" and the assigning of responsibilities to the PMO and SMO
2. Deploying changes - **Addressed** in the form of SCRUM/KANBAN cycles and demand backlogs.
3. Dealing with effects of changes - **Not addressed adequately** because there is no mention of the consideration for redundancy or fault tolerant design of the architecture to enhance recovery.
4. Integration - **Not addressed**. Some may assume that through the mention of value-adding services, the authors claim to argue for the inter-operability of systems. However there is no explicit mention of *inter-operability*. Therefore, I consider it to not be a factor in the LEAD.
5. Decoupling - **Addressed** through the use of small modular architecture increments which specify the scope of the changes and through the reuse of resources.
6. Model-based development - **Addressed**. The authors suggest using a visualization tool and confirm that LEAD can be used with the ArchiMate language and any Domain Driven Design modeling language.
7. SOA - **Not addressed or mentioned**

### 3.3.7 AEA with SAFe

In [74], the authors investigate how effective governance of Agile methods and EA can transpire in an organization and later propose a conceptual model that incorporates factors that take EA into account in the governance of agile scaling frameworks. Gartner defines governance as the processes that ensure the effective and efficient use of IT in enabling an organization to achieve its goals. This also includes the assignment of roles and responsibilities [28]. This study may not provide us with a framework for AEA, but can be considered as a valuable source of information about EA in agile scaling frameworks for the digital transformation of an enterprise.

According to their argument, *agile scaling frameworks (ASFs)* offer benefits such as streamlined prioritization of business requirements, improved management of dependencies, frequent deliveries, and increased employee satisfaction, motivation, and engagement. However, they also highlight common challenges such as cross-team coordination, resistance to change and existing power structures, lack of management buy-in, and the need to maintain an agile mindset.

The authors compare three ASFs, namely, SAFe (Scaled Agile Framework), LeSS (Large-Scale Scrum) & the Spotify model. They mainly conclude that there is no straight forward and widely accepted best practice or framework for scaling agile. In their research, all the 3 companies examined started out with using one of the scaling frameworks mentioned, but in the end used a combination of the 3. But they do point out that out of the 3, SAFe incorporates some EA coordination mechanisms. This implies that none of these frameworks are adequately equipped for a seamless transition from existing enterprise architecture practices to a successful combination of Agile EA at an enterprise-wide scale, necessitating the need for customized solutions. They emphasize that regardless of the methodology chosen, the Agile teams, comprising the Product Owner, Scrum Master, and Agile Coach, must be integrated into the Agile EA governance structure, which encompasses traditional management roles, architecture roles, and supportive EA processes.

There has been a lot of criticism directed towards scaling frameworks for agile, particularly towards SAFe [31] [70]. Concurrently, SAFe is also the currently most widely adopted agile framework [1] and many organizations see its benefits [65]. Therefore, it is important to consider if scaling agile is even necessary for the context of the study. The research question at hand pertains to whether a methodology/framework exists for constructing an enterprise architecture in an agile fashion. This inquiry may bear relevance to the establishment of an agile enterprise, yet it differs from the construction of an enterprise architecture which is agile (AEA). It is important to note here that building an EA in an agile manner does not necessarily equate to the agility of an organization, but it is an important step in getting to overall organizational agility.

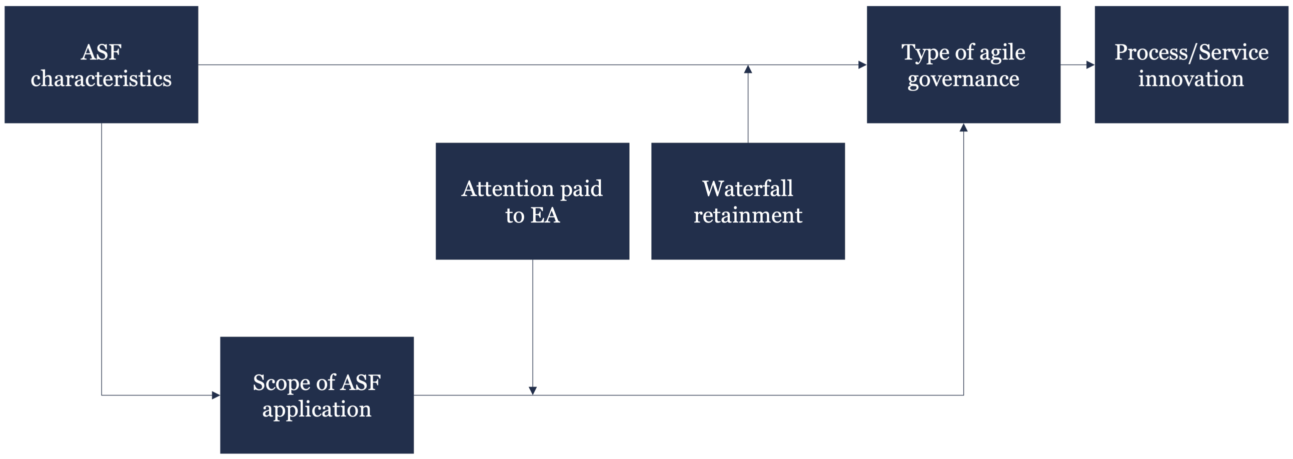


FIGURE 3.8: Conceptual model of factors that take EA into account in the governance of ASFs described in [74]

The authors' conclusion is noteworthy: they suggest that organizations should incorporate certain characteristics of traditional waterfall-oriented approaches into their EA strategy and use them in conjunction with their chosen agile approach. They say that too much team autonomy and too little directions setting by management can be detrimental to the effectiveness of the combination of ASFs and EA.

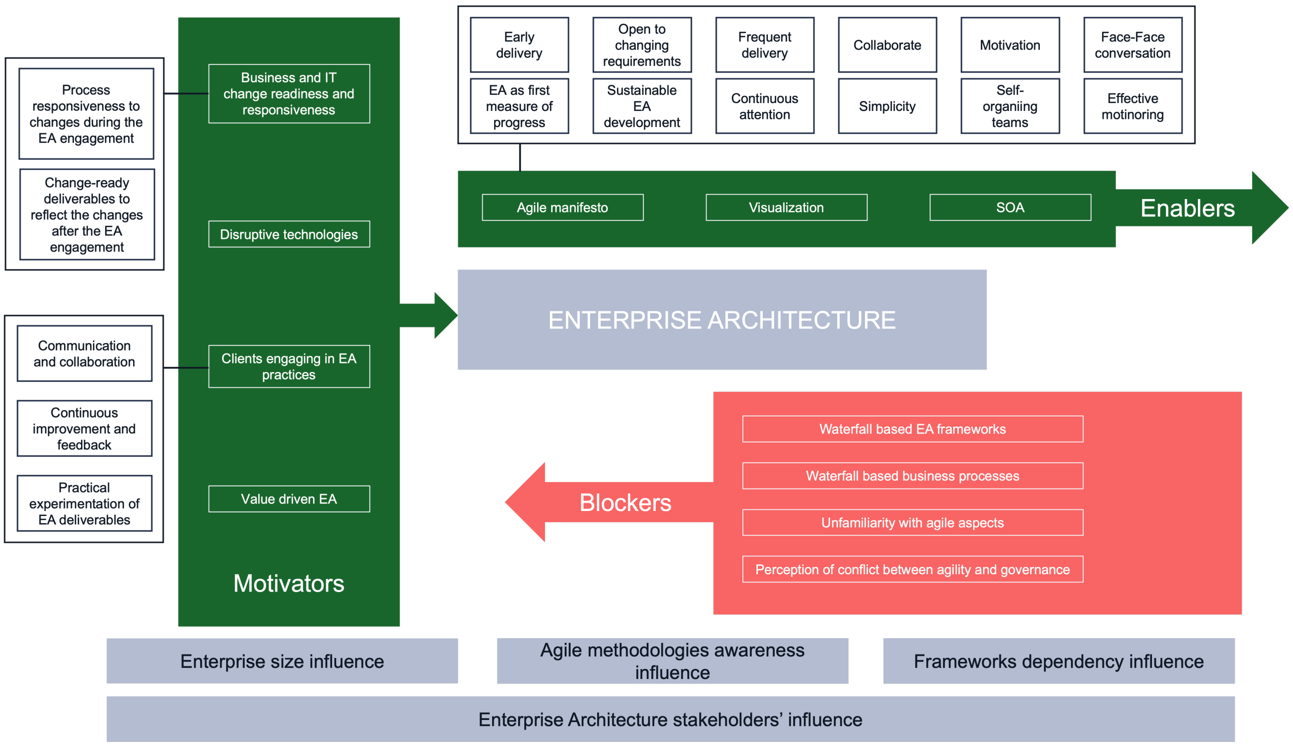
The study also proposes a conceptual model which can be seen in figure 3.8. The constructs in the figure are explained in the table 5 which can be seen in the Tables.

When comparing the constructs presented alongside the agility aspects outlined in section Aspects of system agility, it becomes clear that they are fundamentally different in nature. While agility aspects serve as a framework for constructing an agile architecture, the conceptual model proposed by [74] provides an outline of the factors that need to be considered when implementing enterprise architecture within the governance of an ASF. Thus, attempting to draw a comparison between these two approaches would be inappropriate. As highlighted above, this study may not offer a comprehensive framework for Agile Enterprise Architecture; however, it presents a significant contribution as a source of information concerning Enterprise Architecture within the context of agile scaling frameworks utilized for the digital transformation of an organization.

### 3.3.8 The AEA foundational framework

In their work [42], the authors present a framework for Agile Enterprise Architecture (AEA), which includes identification of motivators, enablers, and blockers. Additionally, they propose a Key Performance Indicator (KPI) called the *Enterprise Architecture Agility Index (EAAI)* for evaluating the degree of enterprise agility. Moreover, the authors translate the principles of the Agile Manifesto into the context of Enterprise Architecture.

The authors argue that investigating AEA requires an examination of the foundational constructs that form the basis for the relationship between agile methodologies and enterprise architecture. These constructs are categorized into motivators, enablers, and blockers, and are presented in the framework shown in Figure 3.9.



Early delivery
Open to changing requirements
Frequent delivery
Collaborate
Motivation
Face-Face conversation
EA as first measure of progress
Sustainable EA development
Continuous attention
Simplicity
Self-organiing teams
Effective motinoring
Agile manifesto
Visualization
SOA
Enablers
Process responsiveness to changes during the EA engagement
Change-ready deliverables to reflect the changes after the EA engagement
Business and IT change readiness and responsiveness
Disruptive technologies
Clients engaging in EA practices
Value driven EA
Motivators
Communication and collaboration
Continuous improvement and feedback
Practical experimentation of EA deliverables
ENTERPRISE ARCHITECTURE
Waterfall based EA frameworks
Waterfall based business processes
Unfamiliarity with agile aspects
Perception of conflict between agility and governance
Blockers
Enterprise size influence
Agile methodologies awareness influence
Frameworks dependency influence
Enterprise Architecture stakeholders' influence

FIGURE 3.9: Foundational Agile Enterprise Architecture framework by [42]

The "*Motivators*" can simply be defined as the major reasons to develop an AEA. The current EA practices that supply and align with agile conceptual instruments and tools are referred to as "*Enablers*". The various forces/obstacles/challenges that may affect the process of agility of an enterprise architecture are defined as "*Blockers*".

The corresponding framework can be seen in Figure 3.9. Upon comparison of the constructs in the framework proposed in [42] and the agility aspects discussed in Subsection 3.3.1, some relations can be observed:

1. Making Changes
   - Separation of concerns - **Not addressed**
   - Low coupling and high cohesion - **Not addressed or mentioned**
   - Encapsulation - **Not addressed or mentioned**.
   - Team Structure - **Not addressed adequately**. Only self-organizing teams mentioned. Roles and responsibilities not mentioned
2. Deploying changes - **Not addressed adequately**. Only frequent and simple deliveries mentioned. However, no mention of test-driven design or resuability.
3. Dealing with effects of changes - **Not addressed adequately**
4. Integration - **Addressed** through the use of SOA
5. Decoupling -**Not addressed or mentioned**. Re-use of system elements not mentioned either.
6. Model-based development - **Addressed** through the visualization enabler
7. SOA - **Addressed**

In addition, as per the definition given by ISO/IEC in [41], an architecture description framework refers to a set of conventions, principles, and practices for the description of architectures within a specific domain of application or community of stakeholders. By this definition, the "framework" presented in [42] falls short in not specifying any conventions, practices, or implementation methods. However, it effectively concentrates on the principles that may be applicable to AEA. Therefore, the concepts proposed in this study can serve as a foundation for developing an AEA methodology that would complement the "framework".

## 3.4 Enterprise Architecture and Digital Transformation

39 studies resulted from the execution of the search query to answer the fourth research question. After further analysis, 5 studies were selected for a deep dive. The motivation to frame the fourth research question is to verify whether and how EA can help in the digital transformation of an organization.

Nowadays, businesses encounter significant changes more frequently, and research suggests that organizations typically implement significant changes once every three years, while minor changes occur almost continuously [43]. Companies are confronted with the task of adapting to changes in business processes and IT infrastructures, driven by globalization, technological advancements, and organizational growth. Consequently, there is a growing need for them to contemplate and pursue significant changes or transformations to sustain or attain a competitive edge [8]. Moreover, significant improvements in technologies such as cloud computing, generative AI, big data technology and processes have given rise to a "digital IT economy," which presents both business opportunities and risks. As a result, organizations are compelled to innovate or confront the consequences [48].

As a solution, Enterprise architecture management has been proposed as a method to steer the development of the enterprise as a whole and the development of its IT portfolio in particular. [68]. By providing a model-based depiction of an organization's business processes, IT systems, and organizational structure, enterprise architecture frameworks aim to enhance the alignment between business and its supporting IT systems. Popular frameworks, such as TOGAF and Zachmann, offer a structured approach to enterprise modeling. Nonetheless, some EA frameworks, such as The Open Group Architecture Framework (TOGAF), have been criticized for their cumbersome size, lack of flexibility, and intricate nature [48]. "EA encompasses all enterprise artifacts, such as business, the organization, applications, data, and infrastructure, which are necessary to establish current architecture visibility and future architecture to produce a road-map" [48].

### 3.4.1 Enterprise Architecture Management

Enterprise Architecture Management (EAM) involves the capture, modeling, analysis, and definition of the present, plateau(transition) and future architectures, along with the development of a road-map that leads from the current state to the desired end state [34]. The purpose of Enterprise Architecture is to optimize across the enterprise the often fragmented legacy of processes (both manual and automated) into an integrated environment that is responsive to change and supportive of the delivery of the business strategy. [33]

#### Types of EA development methodologies

EA methods use many artifacts to represent the information assets of one organization. They are holistic and can be represented from different perspectives or layers, with different (although similar) nomenclatures, which range from the strategy level to the infrastructure one [3].The most commonly used EA frameworks can be classified based on two categories [42]:

- **Grid filling**: These methodologies employ a visual grid that enterprise architects must fill in similar to a Sudoku. Each cell of the grid corresponds to a question about the business, and answering it requires collecting the necessary information from the corresponding stakeholders and articulating a response. (example: Zachmann framework)
- **Structural process**: These methodologies involve structured phases and steps that enterprise architects must follow to produce deliverables upon completion of each step. (example: TOGAF-ADM)

The grid filling frameworks tend to be more classification type schemes rather than a methodology to develop an enterprise architecture [43]. They tend to give a good overview of the current scenario of an enterprise but tend to fall-short when looking at future scenarios. The process methodologies that provide a structure for developing and delivering enterprise architecture deliverables are predominantly designed using the waterfall development style. Consequently, they are susceptible to the same problems, challenges, and issues associated with waterfall management.

### 3.4.2 Traditional EA

The usual approach adopted by companies to meet the demand for enterprise architecture conversion is the traditional "big-bang approach" [8]. However, this big-bang approach of enterprise architecture development may not be the best method considering that modern businesses need to be able reconfigure strategy, structure, processes, people, and technology on the fly and when necessary. To address these issues, a big-bang approach may not be suitable because they encounter several challenges, such as delayed return on investment and the slow recognition of the discipline by the relevant stakeholders [37].

The most popular EA frameworks today (TOGAF) recommend a linear step wise development process for EAM [68]. These approaches bear resemblance to the waterfall style development process that was seen in the software development field. Both methods utilize sequential processes, involve distinct roles in different development phases, rely on detailed front-end plans and documentation, and operate with well-defined guidelines and milestones throughout the project's (in this case an architecture's) development. As highlighted in "The benefits and challenges of waterfall methods" section, the waterfall methodology comes with its own set of problems which are also observed in the traditional methods of EA development.

### 3.4.3 Digital transformation

As mentioned before, digital transformation (DT) has many different definitions. According to a briefing provided by the European Commission, digital transformation encompasses the incorporation of digital technologies into European enterprises and the consequential effects on society from emerging technologies like the Internet of Things (IoT), cloud computing, innovative digital platforms, and blockchain technologies. It is progressively recognized as a critical prerequisite for the success of modern economies and holds the potential to profoundly impact various sectors of the economy, including transportation, energy, agri-food, telecommunications, financial services, manufacturing, and healthcare, thereby transforming the lives of individuals. [54].

[50] provides a definition for digital transformation in the public sector after interviewing several experts in the field;

> Digital transformation is a holistic effort to revise core processes and services of government beyond the traditional digitization efforts. It evolves along a continuum of transition from analog to digital to a full stack review of policies, current processes, and user needs and results in a complete revision of the existing and the creation of new digital services. The outcome of digital transformation efforts focuses among others on the satisfaction of user needs, new forms of service delivery, and the expansion of the user base.

### 3.4.4 EA as an enabler for digital transformation

[4] present an EA framework specifically catered to the digital transformation of a smart city. They suggest that EA frameworks are logical structures in consistent expressed language for organizing and classifying complex information and data sources. Hence, an EA framework can be referred to as conventional practices and principles for the description of architectures established within a particular domain. The authors suggest that within the context of smart cities, EA can facilitate improved communication and information sharing among stakeholders, including enterprises, municipalities, and citizens, thus enhancing the delivery of digital services. They conclude that Enterprise architecture can be employed to support the transformation of smart cities to establish the current and the anticipated state of the city.

[53] proposes an EA framework to drive the digital transformation of primary health services in India. The authors point out that various national healthcare schemes replicate citizen healthcare data in silos which creates data redundancies. This in-turn places more burden on the healthcare workers to collect the right and complete information of a patient to deliver the whole picture of a patients health history. The authors summarize the issues faced by the healthcare industry in India by outlining that the lack of standardization and fragmentation of health technology solutions which results in duplicative and inefficient processes that are unsustainable due to high costs associated with maintaining necessary levels of agility, capacity, and security at scale. As a solution, the authors propose the implementation of an enterprise architecture framework that can effectively address these challenges and present a well-structured digital health infrastructure capable of providing comprehensive primary healthcare to Indian citizens in an efficient and effective manner.

In [78], the authors examine the intellectual structure of digital transformation research. As a part of this research, they identify 7 research themes of DT research one of which is *digital enterprise architecture* and *agile digital transformation.* In conclusion, the authors suggest that a well-designed enterprise architecture (EA) plan can facilitate, regulate, and supervise the changes involved in enterprise transformation.

The authors of [61] confirm that EA is viewed as a promising approach to achieve more effective change management, sustainable planning, efficient IT operations, improved return on investment, and faster, simpler, and cheaper procurement for digital transformation. They even propose an Agile Enterprise Architecture Framework to guide organizations to develop a successful digital transformation strategy.

## 3.5 Summary

This literature review serves as an investigation for developing a methodology for an agile enterprise architecture to support an organization's digital transformation. The literature review was undertaken to answer the research question:

> *What agile practices benefit enterprise architecture management in the digital transformation of an enterprise?*

The research question mentioned above was further was divided into the following sub-sub-research questions:

1. *Which agile methodologies are mentioned in state of the art literature?*
2. *What are the benefits of agile methodologies over traditional waterfall style methods?*
3. *What are best procedures to incorporate agile principles into enterprise architecture management?*
4. *Which techniques/methodologies and practices of enterprise architecture management support the digital transformation of an enterprise?*

Therefore, the research comprises four main areas of interest:

1. The fundamental principles of agile methodologies, including a study of different popular agile development methods.
2. The advantages of agile methods over traditional development methods.
3. The compatibility, necessity, and feasibility of the converging EA with agile methods.
4. The potential of EA to support the digital transformation of an enterprise.

Four research questions were formulated based on these areas of interest, which are explained in the section Review Methodology according to the methodology outlined in [45]. Popular methods such as SCRUM, extreme programming, and KANBAN were studied in detail to answer the first research question. The second research question was straightforward, as most of the literature confirms the advantages of agile methods over traditional methods, and is better outlined in the Literature review findings chapter. The literature on the third research question presents a few frameworks and conceptual models, which are then compared to the aspects of agility outlined in [47]. The comparison of these frameworks and conceptual models with the aspects of agility is presented in Table 3.6. Finally, the studies analyzed to answer the fourth research question confirm that enterprise architecture can be a driving force behind the digital transformation of an enterprise.

### 3.5.1 Agile Methods

It is observed that agile methodologies arose in the domain of software development to mainly address the concern of changing requirements and introducing flexibility in software project development. SCRUM, Extreme programming and Kanban seem to be the most popular agile development frameworks and methodologies used in the industry currently. In all 3 of them development is divided into iterative releases which slowly add functionality as and when required. This perfectly addresses the need for flexibility in software development. However, we also see that XP may not be suitable for distributed development. All of the methods utilize an artifact to represent requirements (KANBAN board, Sprint backlog, XP has customer representation but no physical artifact for this). In the end, all 3 of them promote agility, but SCRUM seems to be the most popular methodology.

### 3.5.2 Waterfall and Agile

When comparing the waterfall and agile methods, it is evident that both approaches offer their own set of advantages and drawbacks. These methodologies have their respective roles and significance within the realm of software development. The benefits associated with both agile and waterfall methods are outlined in Table 3.5, while their limitations are presented in Table 3.4. It is worth noting that agile development **may not be considered suitable** for large-scale, highly critical projects. However, it is essential to acknowledge that this observation could be influenced by other factors, such as organizations relying on waterfall-based business processes or limited familiarity with agile methods throughout the organization.

### 3.5.3 Agile Enterprise Architecture

Across various studies, numerous commonalities can be observed in the approaches to Agile Enterprise Architecture (AEA). Firstly, in the development of an AEA, the use of model-based visualization for developing EA deliverables is confirmed in all the frameworks mentioned in 3.6. Second, the concept of a ***"Backlog"*** is frequently employed to document requirements for the upcoming architecture deliverable which is also the case with the popular agile methods such as SCRUM and Kanban. This approach also represents the input of necessary stakeholders and will be an essential construct in the development of a methodology for an AEA. Third, the studies do not adequately address the concepts of *redundancy* and *recovery* of systems, which is a significant shortcoming in the current body of work in the agile enterprise architecture domain. Fourth, there is limited consideration given to the concept of *"Interoperability"*. The incorporation of an SOA design may help alleviate this issue.

The table 3.6 summarizes the frameworks/conceptual models by specific agility aspects.

TABLE 3.6: Comparison of different AEA frameworks with Agility aspects

| **Agility Aspects** | [59] | [35] | [39] | [42] |
|---|---|---|---|---|
| *Making Changes* | ✣ | ✣ | ✣ | ✗ |
| Separation of concerns | ✓ | ✓ | ✓ | ✗ |
| Low coupling and high cohesion | ✗ | ✗ | ✗ | ✗ |
| Encapsulation | ✗ | ✓ | ✗ | ✗ |
| Team Structure | ✗ | ✓ | ✓ | ✗ |
| *Deploying Changes* | ✣ | ✓ | ✓ | ✗ |
| Install-ability | ✓ | ✓ | ✓ | ✗ |
| Usability | ✗ | ✓ | ✓ | ✗ |
| *Dealing with effects of changes* | ✣ | ✗ | ✗ | ✗ |
| Redundancy | ✓ | ✗ | ✗ | ✗ |
| Recovery | ✗ | ✗ | ✗ | ✗ |
| *Integration* | ✗ | ✗ | ✗ | ✓ |
| *Decoupling* | ✗ | ✗ | ✓ | ✗ |
| *Model-based development* | ✓ | ✓ | ✓ | ✓ |
| *SOA* | ✗ | ✗ | ✗ | ✓ |

Legend: ✓- Addressed ; ✗- Not addressed ; ✣- Partially addressed

### 3.5.4 Enterprise Architecture and Digital Transformation

Digital transformation (DT) has emerged as a strategic approach for enterprises to effectively respond to external pressures and navigate the challenges posed by competitive and unpredictable environments. In this context, entrepreneurs recognize the significance of DT as a catalyst for change within their organizations. According to the "2019 Digital Transformation Market Trends Report", majority of respondents consider DT to be indispensable for the survival of their organizations and many have prioritized DT as their topmost concern, emphasizing its criticality in achieving organizational objectives and staying competitive in today's rapidly evolving landscape. As can be seen from the 3.4.4, enterprise architecture (EA) has been validated as an effective tool for facilitating the successful DT of a firm. This suggests that AEA maybe able to further enhance the DT of an enterprise. Furthermore, this allows us to further investigate the idea of *Continuous Improvement* where the focus is on the establishment of an ever-improving culture at a firm.

### 3.5.5 Agile practices that benefit Enterprise Architecture

When trying to answer the question which agile practices should be used for the development of an agile enterprise architecture methodology, it can be observed that not all agile practices should be adopted in the development of an Enterprise Architecture.

Firstly, The idea of having no documentation in an EA is not realistic, but what is realistic is that the documentation should be "*just-enough*". This means that the documentation of EA elements or constructs should give required information at a glance and no more than that.

Secondly, it is observed from the table 3.4 that agile methods were unsuitable for the development of large-scale software with many dependencies. This may also apply to the domain of enterprise architecture since in many cases, the design of the dependencies is related to large-scale organization spanning software solutions such as ERP softwares. However, it can be argued that agile methods are suitable for the design of these solutions, just not the development and implementation of them.

Third, from table 3.4, it is observed that following an agile approach requires members to be highly skillful and with excellent ownership capabilities. This must be addressed in the design of the IT artifact, AEA-DM.

Fourth, To answer the main research question, practices such as incremental development, collaboration, customer/stakeholder representation in the form backlogs or a representative (XP), continuous optimization and test-driven design are all considered beneficial for the domain of enterprise architecture. Additionally, it is decided that the agility aspects must be considered as a *testing criteria* for an enterprise architecture in the development of the methodology.

# Chapter 4

# Methodology Design

This chapter is devoted to addressing the second sub-research question, namely, *"How can the design or engineering of an enterprise architecture be made agile?"* The primary objective, therefore, is to propose a solution that enables the adoption of an agile approach to enterprise architecture. By examining the structure of the question, it becomes evident that this is a design problem, making it a design-oriented inquiry. Consequently, a solution needs to be meticulously *designed*. As discussed in the chapter "Research Design," the methodology serves as the main output of this research. It represents the IT artifact employed to tackle the aforementioned problem. The design of the methodology is predominantly influenced by the literature study conducted to answer the first sub-research question. This methodology is further divided into two distinct sections. The first section provides a comprehensive overview of the prerequisites that an enterprise or organization must fulfill to effectively leverage the methodology. The second section delves into the methodology itself, elaborating on its key components and processes in detail.

## 4.1 Pre-requisites

A high-level overview of the required pre-requisites have been identified during the literature review and can be viewed at one glance in the figure 4.1

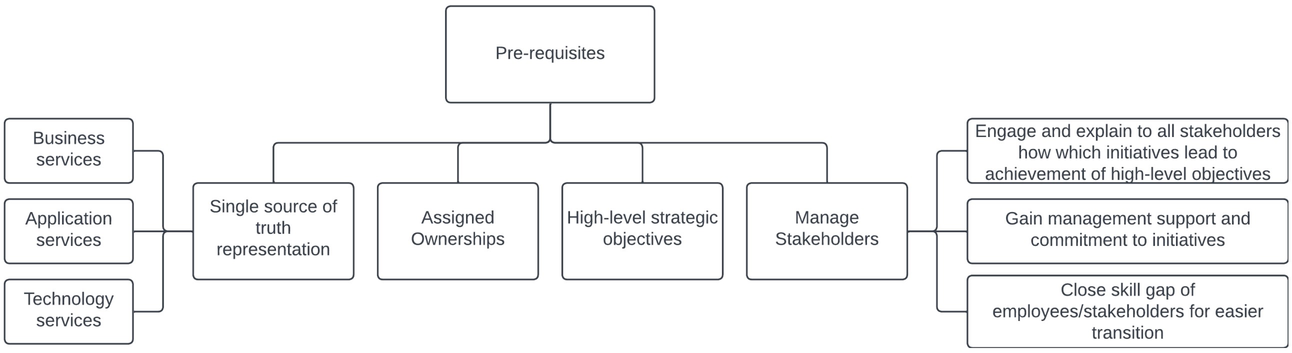

Pre-requisites
Business services
Application services
Technology services
Single source of truth representation
Assigned Ownerships
High-level strategic objectives
Manage Stakeholders
Engage and explain to all stakeholders how which initiatives lead to achievement of high-level objectives
Gain management support and commitment to initiatives
Close skill gap of employees/stakeholders for easier transition

FIGURE 4.1: Pre-requisites of the AEA-DM

It is obvious that not all organizations are the same and some organizations are ahead of others in terms of being able to adopt the AEA-DM. But there is no way to quantify how ahead these organizations are. As such, it was decided to define a set of criteria to deduce if organizations are ready or not to be able to adopt the AEA-DM. It can be seen as a check-list of sorts. Of-course, just because an organization satisfies these criteria does not mean that the adoption of the methodology will definitely go well. That depends on many other factors and is considered as not within the scope of the thesis. These are just objective criteria.

### 4.1.1 Single source of truth representation

In their study [74], the authors emphasize that the primary aim of Enterprise Architecture is to offer comprehensive holistic perspectives and standards, and guiding the cohesive design and implementation process. This is achieved by facilitating technology decision-making and setting the direction for business and IT stakeholders. [47] mentions that EA provides a comprehensive understanding of the enterprise by considering the inter-dependencies between different aspects of the business-IT domains and by shedding light on these dependencies, EA enables informed management decisions by clarifying the potential impacts and implications. A single source of truth philosophy dictates that an organization must create a single aggregated repository of company data at one source at a component level. This allows anyone in the company to freely access trustworthy data at a component level as it essentially eliminates duplicate data. However, this is not the pre-requisite for an AEA-DM, the pre-requisite for the methodology is when the philosophy is applied to the level of an enterprise architecture. This basically boils down to describing your data all the way from its business services to the application services to the technology services. Experts in the domain will immediately recognize this as having an enterprise architecture *"As-Is"* state.

Without this representation of how your business is served by its IT, there is no way that enterprise architecture can represent a transformed state of your business. According to [47], classical agile development discourages big upfront design (BDUF) due to the inherent uncertainty involved. However, it is advisable to proactively design the elements of your organizational and technical landscape that serve as a stable foundation for enterprise agility. This implies that having a well-defined as-is landscape, which accurately captures the alignment between the organizational business and IT, can serve as a solid groundwork for successful transformation. In simple terms, without an "As-Is" landscape, there is no "To-Be" landscape. This "As-Is" landscape must not only be good representation of your organization's business services or business capabilities, applications, and technology but also the dependencies between different applications and technologies.

### 4.1.2 Assigned Ownerships

As seen from the results of the literature review, adopting an agile approach requires advanced ownership capabilities. According to [67], it is observed that team members of an agile team are required to be both skillfull and have ownership capabilities. According to [20], it is recommended to establish a committee of owners consisting of an architecture owner, a business owner, and an application owner, with each owner fulfilling specific responsibilities. The business owner is responsible for overseeing all business processes within the enterprise and ensuring their proper implementation on a daily basis. On the other hand, the application owner is tasked with defining and maintaining the metrics related to the applications and IT systems in use.

In the context of a digital transformation initiative, it is crucial to ensure that enterprise architects have access to pertinent information about specific business functions, applications, and technology elements. However, obtaining such information can be challenging as architects may not know whom to approach for the required details. This underscores the importance of documenting ownerships for all business, application, and technology elements as part of the "As-Is" landscape. By describing ownerships, enterprise architects can readily identify the relevant stakeholders and efficiently gather the necessary information, thereby facilitating the successful execution of digital transformation initiatives.

### 4.1.3 High-level strategic objectives

The effective implementation of business transformation, which poses a significant challenge in managing the complexity of existing business processes and integrating Information Technology (IT) at a large scale to meet the diverse requirements of all stakeholders within the organization, relies heavily on the crucial role played by corporate management in developing a clear mission, vision, and strategy. Having high-level strategic objectives is crucial for organizations in achieving their goals and aligning their business structure with digital technology. Enterprise Architecture plays a significant role in this process by ensuring that the objectives and needs of the organization are aligned with a common direction. By creating a blueprint of corporate architecture, organizations can effectively plan and accommodate modifications to adapt to digital transitions and prepare for flexibility. This alignment between strategic objectives and architecture enables organizations to navigate the complexities of digital transformation and optimize their operations accordingly [61]. The integration of agile methods and EA should enable the definition of long-term goals for and the concerted development of the IT landscape, while keeping an eye on the business strategy and the overall goals of the enterprise [35]. The agile enterprise architecture framework by [59] has 4 levels of concern. Strategy and implementation constraints at the top, which describes the objectives and their priorities. This being at the top is a good indicator that having high level objectives to strive towards is important. Attaining flexibility within an enterprise architecture requires deliberate consideration and entails addressing a variety of concerns from many disciplines. It is imperative to explicitly determine the areas where agility is desired, while also identifying aspects that can be standardized or permanently established. This decision-making process is guided by a well-defined set of business objectives and drivers [47].

In the absence of high-level strategic objectives, all initiatives within an organization may lack direction and fail to contribute to the overall improvement of the enterprise. While agile methods emphasize adapting to changing requirements, it is crucial to have well-defined high-level goals in place. Implementing agile practices within an enterprise does not imply the abandonment of goal-setting. On the contrary, having clearly articulated objectives remains essential to guide and align agile initiatives effectively.

### 4.1.4 Stakeholder Management

The engagement of employees and the communication of company-wide goals are essential for successful agile scaling frameworks. Challenges in this regard include cross-team coordination, resistance to change, existing power structures, lack of management buy-in, and maintaining an agile mindset [74]. The significance of adequately equipping and preparing all stakeholders, including top executives and end users, in terms of skills, is underscored in [8]. The aim is to ensure that these stakeholders are well-prepared to fully utilize the advantages offered by new solutions.

Hence, it is imperative within an academic context to actively involve and provide comprehensive explanations to all stakeholders engaged in a digital transformation endeavor regarding the objectives of the initiative and how it aligns with the company's overarching goals. Additionally, gaining C-level and management support is crucial in overcoming challenges that may arise during the adoption of such initiatives.

## 4.2 Methodology

This section primarily focuses on the development of the Agile Enterprise Architecture - Development Method. The entire methodology can be viewed in the figure 4.2

### 4.2.1 Design choices

A method refers to a specific procedure or approach employed to accomplish a particular objective, often characterized by its systematic and established nature. Hence, to effectively depict the methodology, a process flow diagram was selected as the suitable tool. The utilization of a process flow diagram ensures a clear and structured representation of the methodology, facilitating a thorough understanding of the steps involved and their logical sequence. This visual depiction aids in conveying the methodological approach in a coherent and satisfactory manner, enabling readers to grasp the overall framework and flow of the process.

The incorporation of icons in the diagram serves to indicate different artifacts generated throughout the methodology. The presence of white boxes containing text within the diagram illustrates various processes, while the arrows denote the chronological occurrence of these processes. Additionally, the utilization of the stakeholder icon in the methodology diagram signifies the diverse roles required for successful implementation of the methodology.

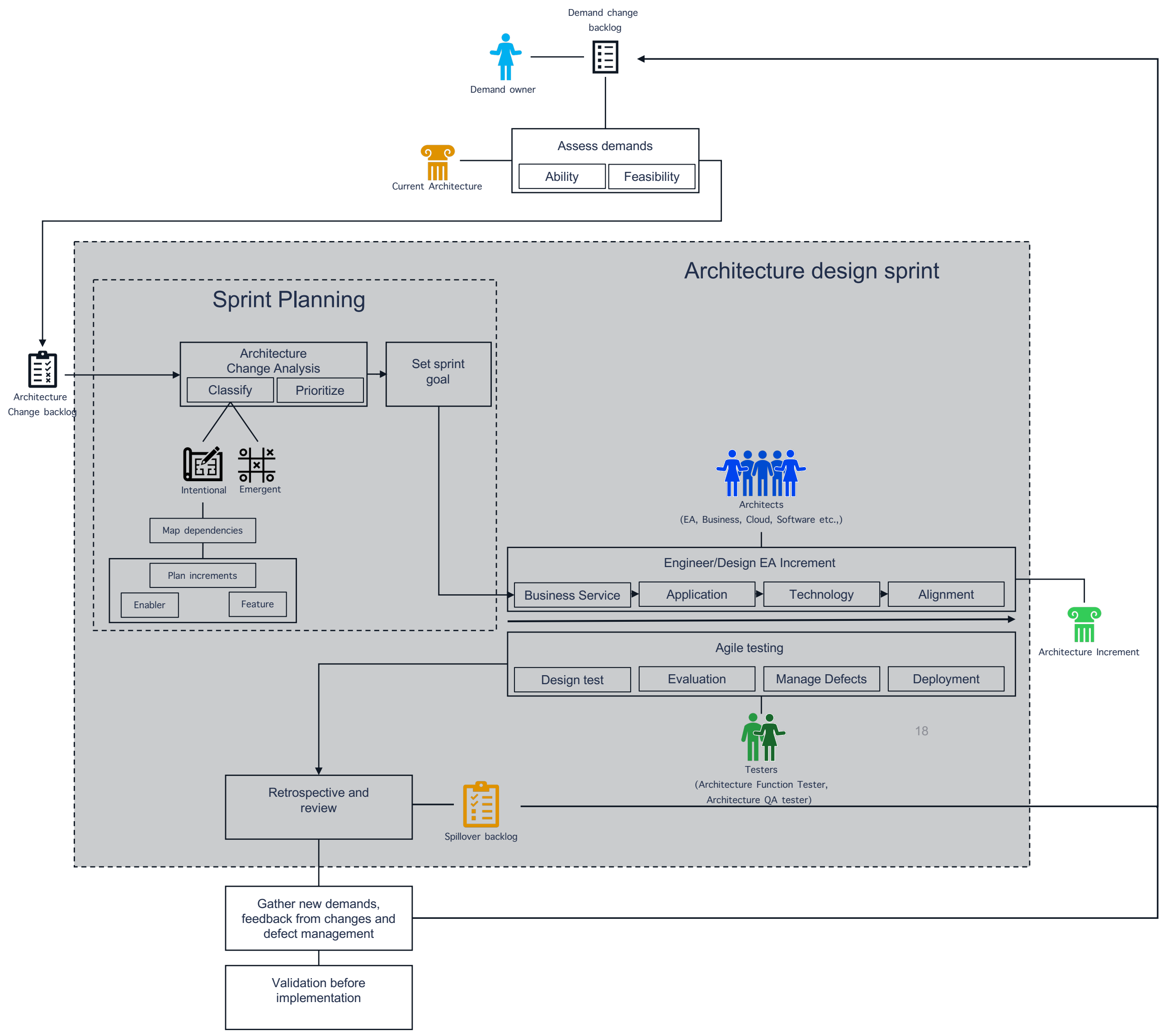


Figure 4.2: The Agile Enterprise Architecture - Development Method (AEA-DM)

### 4.2.2 Team Roles

The team roles will first be discussed before the methodology is explained in detail. This is to ensure better explanation. The various team roles employed in the methodology are:

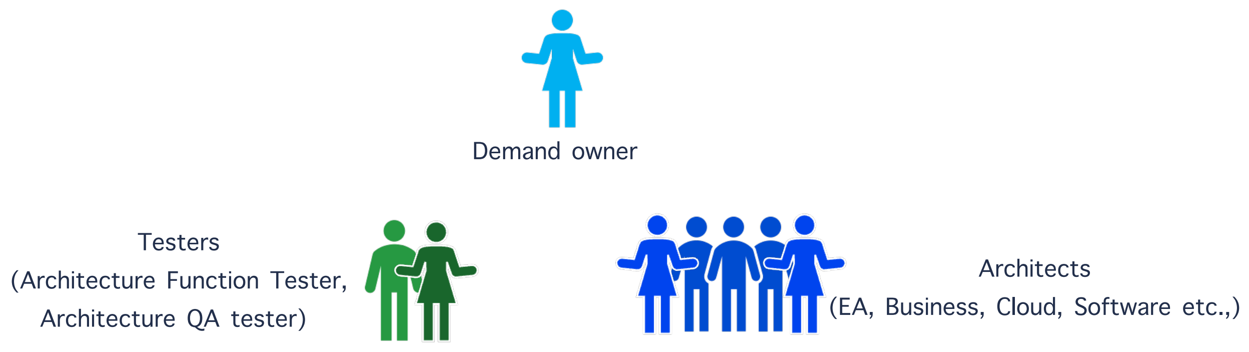

Demand owner
Testers
(Architecture Function Tester,
Architecture QA tester)
Architects
(EA, Business, Cloud, Software etc.,)

FIGURE 4.3: Team roles of the AEA-DM

1. *Demand Owner* - Responsible for the demand backlog and the processes associated with it, namely assessing the demands.
2. *Architects* - The design and engineering of an EA increment, the Architecture change analysis and the sprint goal setting are the responsibility of the architects. It is recommended to have an Enterprise Architect, Business Architect and a Solution Architect as the people working together on this. However, depending on the situation, other architect roles might be necessary such as a Cloud Architect and maybe a representative from the provider of a cloud solution
3. *Architecture function tester* & *Architecture quality assurance tester* - Both of these roles are responsible for the agile testing block in the sprint

### 4.2.3 Methodology Workflow

#### Demand Collection

As previously mentioned, the initial phase of the methodology commences with an existing "As-Is" Landscape and the "Demand Change Backlog." The demand change backlog essentially comprises a compilation of all the demands expressed by stakeholders pertaining to the enterprise architecture function within an organization. In order for this backlog to effectively serve its purpose, certain requirements must be met, including its wide awareness and transparency throughout the organization.

Therefore, prior to implementing this methodology, it is imperative to educate stakeholders about the process through which they can submit their demands to the demand change backlog. It is worth noting that the specifics of this pipeline are tailored to each organization and are not extensively discussed within the methodology.

#### Demand Assessment

Once the demands are registered in the demand change backlog, they need to undergo an assessment process to determine their technological changeability and feasibility. This underlines the significance of the "Demand Owner" role, which necessitates possessing specialized knowledge regarding the organization's technological capabilities, as well as financial acumen to gauge the demands' feasibility in light of the organization's financial situation. Moreover, the Demand Owner must have a comprehensive overview of the landscape to ascertain whether a demand calls for a new solution or if an existing solution within the enterprise can suffice. For instance, let's consider an example where Employee A from the HR department submits a demand for a new application to enhance a specific HR function. Unbeknownst to Employee A, the requested application is already available to employees in a different department within the same company. In such a scenario, the demand remains valid; however, fulfilling the demand may not require an entire sprint but could be accomplished within a few hours, depending on the company's specific procedures. Therefore, the assessment process carried out by the Demand Owner plays a crucial role in determining the appropriate course of action for each demand, taking into account existing solutions, potential implementation efforts, and overall organizational efficiency.

The output of this assessment process leads to the "Architecture Change Backlog". The Architecture change backlog (ACB) is simply the list of demands that have been accepted for further development after the assessment process. These demands serve as the input for the Architecture Development/Design Sprint.

A zoomed in view of the demand assessment process can be viewed in the figure 4.4

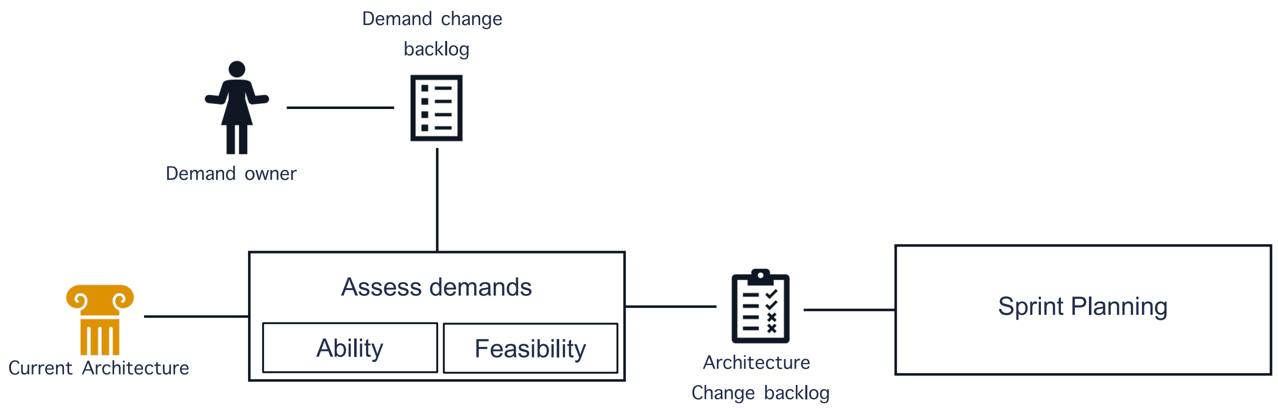


FIGURE 4.4: Demand Assessment

### Architecture Change Analysis

A zoomed in view of the Architecture Change analysis and the total sprint planning process can be viewed in the figure 4.5

The first process in the sprint is the Architecture Change Analysis where the demands are first prioritized and then classified into intentional and emergent changes. The prioritization of the demands is a critical step in order to realize the order in which demands will be fulfilled. For the prioritization, any methodology such as MoSCoW or BRICE or the 4 quadrant of time management or etc., may be used. The classification of steps into intentional and emergent changes is adapted from the Scaled Agile Framework's (SAFe) Architectural Runway.

As per the principles outlined by SAFe (Scaled Agile Framework), intentional design involves the deliberate establishment of architectural strategies and initiatives aimed at improving solution design, performance, and usability. These intentional design elements serve as guiding principles for cross-team collaboration in designing and implementing synchronized solutions. On the other hand, emergent design approach enables a fully evolutionary and incremental implementation methodology, wherein developers and designers can promptly address immediate user requirements. This approach allows the system's structure to evolve organically throughout the development and deployment process, ensuring a more responsive and adaptable solution.

As mentioned in the literature review, in the present day, businesses are confronted with frequent and substantial changes, as evidenced by research indicating that organizations tend to undergo major changes approximately once every three years, alongside the occurrence of continuous minor changes [43]. This implies that organizations will encounter less frequent changes that necessitate intentional design, while experiencing more frequent changes that call for emergent design. Consequently, architects must possess the ability to address these smaller demand changes within a span of 1-2 sprints. Conversely, changes requiring intentional architecture demand a longer timeframe, spanning multiple sprints, to ensure their successful execution. This step is crucial in determining the project type required to meet such demands. These prioritized and classified demands are subsequently recorded in the modified change backlog. Following this stage, a sprint goal is established. The sprint goal should be sufficiently manageable so that the required changes can be accomplished within a single sprint.

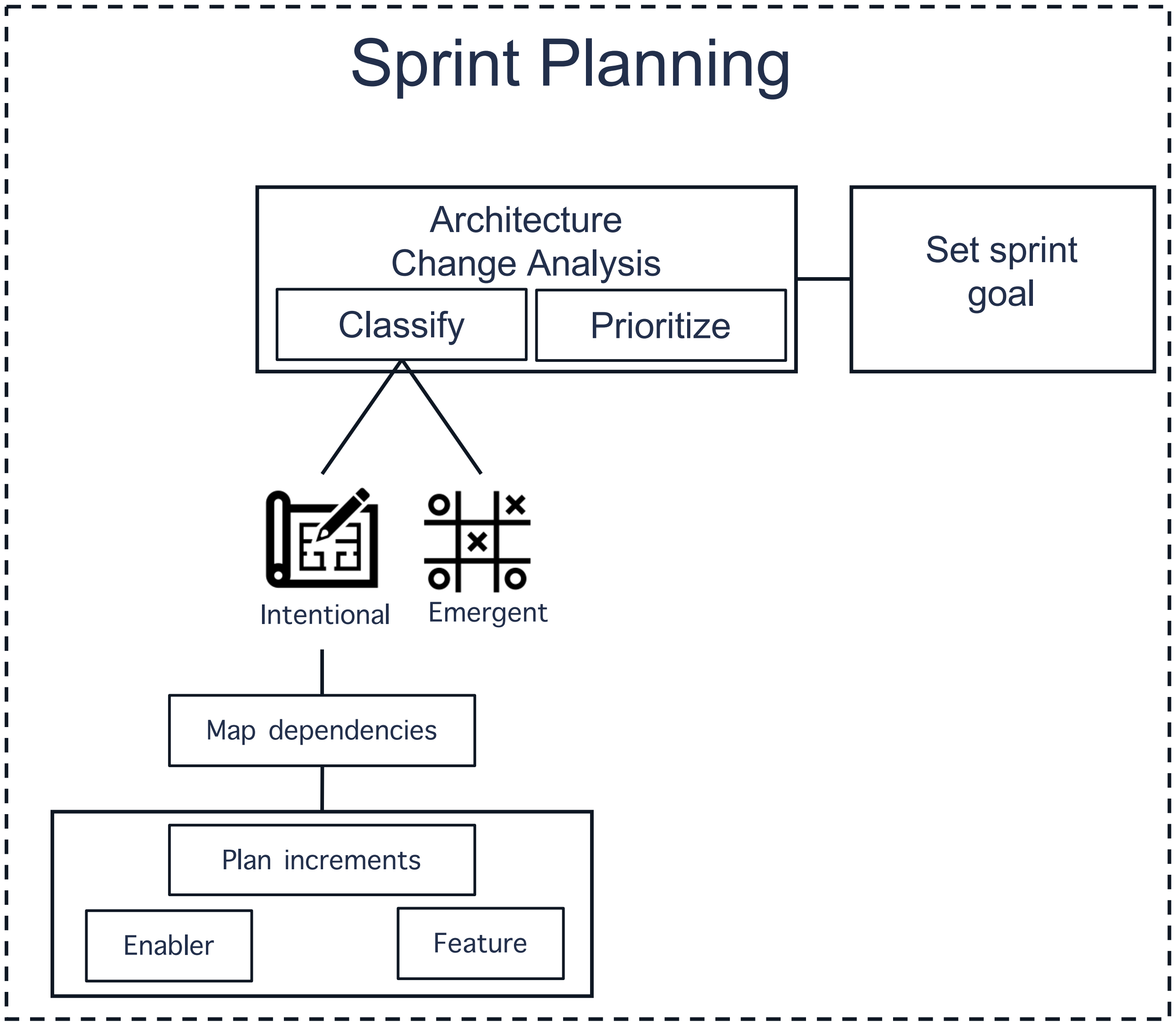


FIGURE 4.5: Architecture Change Analysis

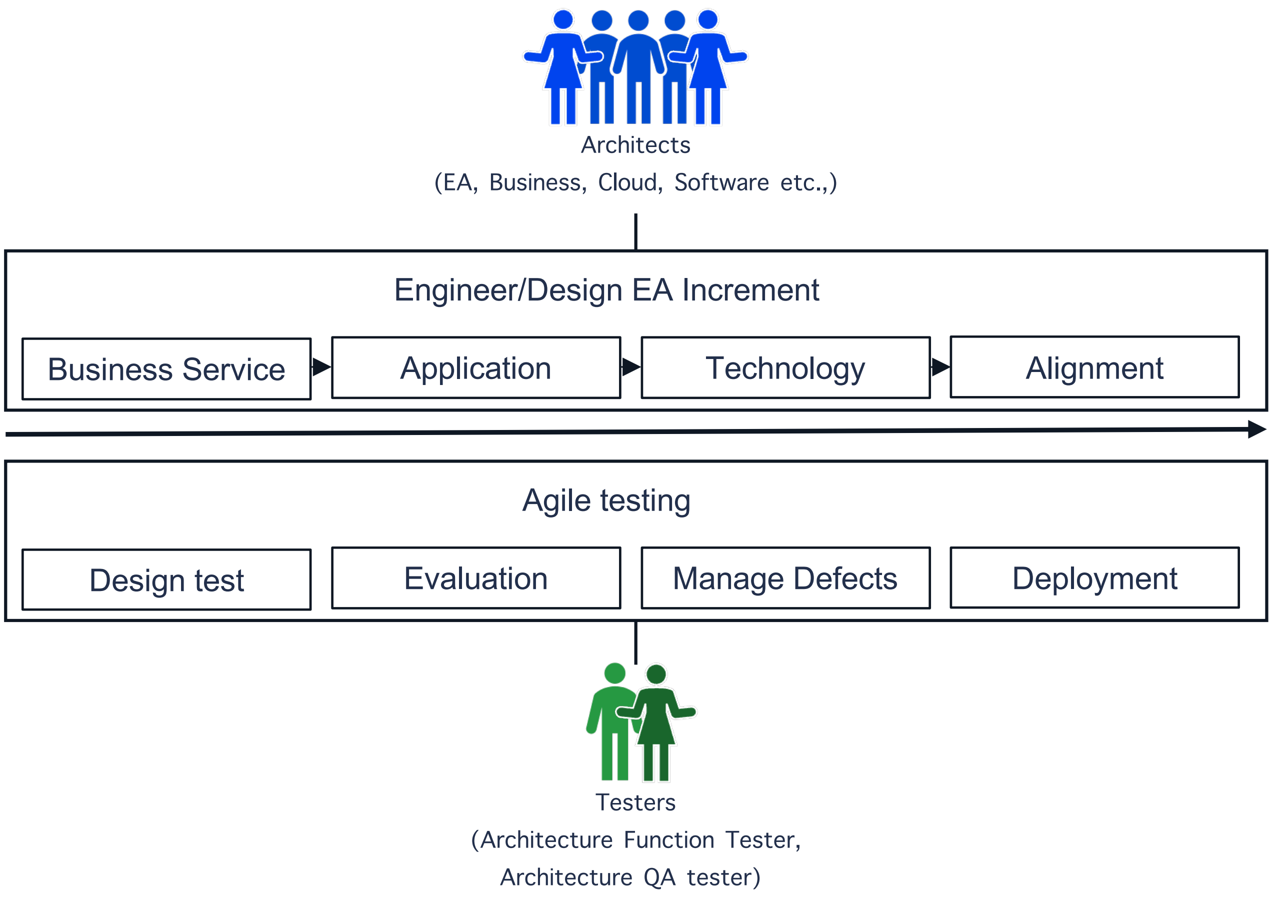


FIGURE 4.6: Design of the new EA increment

It is advisable to address 1-2 demands per sprint, ensuring that the development team can effectively handle the workload without being overwhelmed.

### Design of EA Increment

A zoomed in view of the Design of EA Increment process can be viewed in the figure 4.6.

The subsequent phase of the enterprise architecture (EA) development process entails designing or engineering the EA increment based on the selected demands for the sprint. In the agile approach to EA, Service-Oriented Architecture (SOA) is the recommended architectural style [42]. While SOA is presented as a form of Agile Architecture [12], the emphasis is placed on addressing business problems rather than the agility of the solution being built. Therefore, the primary focus in architecture design is to determine whether the modifications made to the architecture effectively resolve business problems.

The critical question arises due to the inherent nature of SOA, which revolves around Services. It is essential to define the concept of a "Service" and assess whether the implementation adheres to an architecture that is designed with a Service-oriented approach. In the context of SOA, a Service is a simplified representation of business functionality and/or data presented within a business context [12]. To qualify as an SOA implementation, it is necessary to include at least one abstracted Business Service. Consequently, throughout the design process of the EA increment, this architectural style must be followed. Elements within the application layer must align with the business, and elements within the technology layer must align with the application or business layers. By adhering to this style of EA design, business-IT alignment is ensured.

### Test-driven design

From the figure 4.2 it can be seen that simultaneously with the design of the EA increment, the agile testing of the increment takes place in parallel. This testing of the EA increment has 4 functions, namely,

A zoomed in view of the agile architecture testing process can be viewed in the figure 4.6

1. *Design test or use-case test*: The design test is a test to understand whether the changes that are needed to be accomplished by the architecture have been accomplished. This can be achieved via checking for whether the main use-case that needed to be achieved has been achieved by the sprint.

2. *Architecture evaluation*: The evaluation of the quality of the architecture increment is one the many aspects of the methodology. Here, the architecture increment is tested for quality metrics like the various agility aspects mentioned in the section 3.3.1. Moreover, the EA Evaluation model proposed by [51] which is seen in the figure 2 in the Appendix - Figures offers a comprehensive set of quality attributes for assessing the effectiveness of an EA implementation. These attributes can be utilized as a valuable checklist in the form of a questionnaire to ensure the quality and success of the EA initiative.

   An important observation to note is the significant overlap between the agility aspects and the quality attributes outlined in the evaluation model. For instance, alignment and integrity are closely related to the concept of separating concerns and ensuring that IT effectively meets the business requirements, as exemplified in the design of Service-Oriented Architecture (SOA). Similarly, reusability aligns with the agility aspects of installability and usability. Furthermore, scalability of EA services can be achieved through the implementation of decoupled and integrated architectures, such as SOA. To address this, a potential approach would involve integrating both the agility aspects and quality attributes into a comprehensive checklist for evaluating the quality of an EA increment.

3. *Defect management*: In the realm of software development, an "Defect" refers to the difference between the actual outcomes and expected outputs. However, within the context of architecture, a defect is characterized as a deviation or malfunction in the functionality of the enterprise architecture increment. These defects could be identified in the same sprint or outside the sprint by another stakeholder. It must become a priority to fix these defects as soon as possible to resume normal EA functionality

4. *Deployment management*: Deployment management is the practice of updating the Enterprise Architecture increment as the new "As-Is" landscape. This must be executed after careful consideration and at the final stage of the sprint

### Retrospective and Review

This practice is directly adapted from SCRUM for the purposes of continuous improvement of the architecture development. The team has an end of the sprint meeting to discuss whether all sprint goals have been achieved to the quality that was desired and also discuss team dynamics and how they can improve of the next sprint.

Any demands that are not adequately addressed are added to a "Spillover backlog" which then gets added to the new Demand change backlog so that these unaddressed demands maybe considered for future sprints.

### Repeat

After the end of the sprint, new demands and defects are collected and added to the demand change backlog by the demand owner. This ensures that that organization is continuously evolving and is agile.

The implementation of the enterprise architecture increments in the organization are considered out-of-scope for the thesis. But it is suggested that any EA increment must be validated before complete implementation. This could be in the form of user acceptance tests, pilot tests or stakeholder feedback etc.,

## 4.3 Summary

This chapter focuses on addressing the second sub-research question, which examines how the design and engineering of an enterprise architecture can be made agile. This can only be answered through the design of the methodology, but first it is important to understand that the readiness of organizations to adopt Agile Enterprise Architecture Decision Making (AEA-DM) cannot be quantified, but a set of criteria has been defined to assess their readiness. These criteria serve as a checklist and do not guarantee the success of the methodology adoption, as other factors are also influential and beyond the scope of the thesis.

1. The first criterion is the establishment of a single source of truth representation, where an organization creates a representation of a consolidated repository of company data at the component level. This representation captures the alignment between the business and IT domains and enables informed management decisions. It is crucial for an "As-Is" landscape to be well-defined, encompassing business services, applications, technology, and their dependencies. Without this representation, transformation to a desired state cannot be achieved.
2. The second criterion emphasizes the importance of assigned ownerships for business, application, and technology elements. Documenting ownerships in the "As-Is" landscape enables efficient information gathering and supports successful digital transformation
3. High-level strategic objectives form the third criterion, guiding organizations in achieving their goals and aligning their business structure with digital technology.
4. The final criterion highlights the importance of prepared stakeholders, including employees, top executives, and end users, who need to possess the necessary skills and understanding of the initiative's objectives and alignment with company goals.

The sub-research question is answered through the means of the Agile Enterprise Architecture - Development Method (AEA-DM) which is a systematic approach for developing an enterprise architecture in an agile manner. It employs a process flow diagram to provide a clear and structured representation of the methodology, facilitating understanding of the steps involved. The methodology incorporates various team roles, including the Demand Owner, Architects, and testers responsible for agile testing.

The workflow of the methodology begins with demand collection, where demands expressed by stakeholders are registered in the demand change backlog. These demands then undergo an assessment process to determine their technological changeability and feasibility. The demands that pass the assessment are recorded in the Architecture Change Backlog, which serves as input for the Architecture Development/Design Sprint. The sprint involves Architecture Change Analysis, prioritization, and classification of demands into intentional and emergent changes. The Design of EA Increment follows, focusing on designing the enterprise architecture increment based on the selected demands. Concurrently, test-driven design takes place, including design testing, architecture evaluation, defect management, and deployment management. The sprint concludes with a retrospective and review meeting to discuss achievements and improvements. The cycle repeats with the addition of new demands and defects to the demand change backlog.

# Chapter 5

# Demonstration and Evaluation

Following the development of the AEA-DM, this chapter provides a comprehensive evaluation of the various constructs within the methodology. The subsequent sections present the results obtained from expert interviews and conclude with an enhanced version of the AEA-DM, incorporating the insights gained from the interview analysis. This chapter attempts to answer the final sub-research question, *Does the developed methodology provide a viable approach for designing and implementing enterprise architecture?* by evaluating the various constructs mentioned in the previous chapter, Methodology Design.

As mentioned in Chapter Research Design, this study adopts the Design Science Research Methodology (DSRM) proposed in [56] where both the demonstration and evaluation of the artifact in this case go hand-in-hand. Design cycles are an integral part of this methodology, allowing for the evaluation, validation, and improvement of artifacts. In the case of the Agile Enterprise Architecture - Development Method (AEA-DM), which aims to enhance organizational agility through an agile enterprise architecture, comprehensive validation is a long-term endeavor due to the time required for organizational change and digital transformation. Fully validating the AEA-DM would involve the adoption of the methodology by organizations and its application in specific digital transformation initiatives, a process that may span several years. However, given the timeframe of this thesis, this type of validation of the AEA-DM is beyond its scope. Nevertheless, evaluation of the methodology is within the scope of this study through the means of expert interviews and opinions. An initial version of the AEA-DM which was explained in the Chapter Methodology Design will be evaluated through expert interviews. The methodology will be then be updated based on the feedback received.

Before the evaluation of the AEA-DM the methodology was explained to the interviewees and a consent form for the use of this thesis was obtained. Along with the methodology, a proto-type of how the methodology could be implemented was shown to the interviewees with the help of the [redacted]

## 5.1 Demonstration of the AEA-DM

The methodology shown in the figure 4.2 has been demonstrated using an enterprise architecture management tool. This demonstration will address various parts of the methodology.

### 5.1.1 Demand Management

[redacted]

### 5.1.2 Demand Classification

Since 1 project would only deal with one or 2 demands, it was decided that the [redacted] was suitable to denote the "enabler" or "feature deployment" sprints. This was done using a new data field called "Epoch type" or "Project type". [redacted]

### 5.1.3 Architecture development

[redacted] This visualization shows that the application layer is beneath the business layer and all application layer decisions, must be dictated by the business layer (business capabilities).

Additionally, the projects which deal with a demand each can also be visualized through the use of the in-built [redacted] This could also be viewed via the use of other PM tools such as JIRA etc.,

## 5.2 Evaluation Design

The examination of the relationship between business and information technology has long been acknowledged as a complex and challenging area of study. This complexity arises from technical intricacies, the intricate interaction between technology and human capabilities, and the crucial role of human behavior. While the former two aspects have presented numerous intricate and captivating research problems, it is the latter aspect - human behavior - that has only recently begun to receive significant attention in a more holistic manner. In the research conducted in [64], the main objective is to demonstrate the adaptation and incorporation of qualitative methods in empirical studies within the field of software engineering. The key advantage of utilizing qualitative methods lies in their capacity to compel researchers to thoroughly explore the complexities of the specific problem rather than simplifying or abstracting it. Consequently, the author contends that employing qualitative methods leads to more comprehensive and insightful outcomes. As a result, for this study, a qualitative approach has been chosen as the method of evaluation.

### 5.2.1 Interviews

According to [64], interviews are a commonly employed qualitative data collection technique. They serve as a means to gather opinions and impressions about a particular subject in research studies. In line with the purpose of this study, interviews were conducted to obtain the interviewees' impressions on the various constructs of the methodology. The interviews comprised a combination of open-ended and specific questions, allowing both the researcher and the interviewees to delve into the focus areas and expand their thoughts on them.

It is worth noting that the conducted interviews followed a semi-structured format, involving a mix of open-ended and specific questions. The questions were thoughtfully designed to capture both anticipated and unforeseen information. To ensure consistency and alignment with the research objectives and questions, an interview guide was prepared as a reference. However, the interviews progressed smoothly and followed a well-organized structure, making the use of the interview guide unnecessary. The interviewees consistently provided relevant and meaningful responses throughout the interviews. The transcripts of the interviews can be found in the Appendix Transcripts chapter .

The details of the interviewees are presented in Table 5.1. For confidentiality reasons, the names of the interviewees are not provided. The table includes columns indicating the interviewees' job roles, job responsibilities, experience in the enterprise architecture (EA) domain, and their familiarity with agile methods (AM), digital transformation (DT), and enterprise architecture (EA). [redacted]

[redacted]

Table 5.1: Interviewees and their details

| Interviewee | Job Responsibilities | Experience | Familiarity with AM, DT, EA |
|---|---|---|---|
| Interviewee 1 | Team lead for various transformation projects | 5 years | Yes |
| Interviewee 2 | Acquiring projects with existing customers & speak at external events & Start, lead and complete various projects, build internal course work for employee education | 9+ years | Yes |
| Interviewee 3 | Drive customer adoption of EA tool | 6+ years | Yes |
| Interviewee 4 | To try and shorten the time to value with customers | 9+ years | Yes |

### 5.2.2 Qualitative Data Analysis and Coding of interviews

Due to the study's constraints, which include a limited time-frame and reliance solely on interviews for evaluation, traditional qualitative data analysis methods such as triangulation, negative case analysis, and replication could not be employed. Consequently, the analysis of the interviews became the primary means of assessing the constructs of the methodology. In light of this, codes were developed to substantiate the utilization of the methodology's constructs and reinforce the corresponding ideas associated with these constructs.

The interviews were conducted remotely using the Zoom software, facilitating video calls between the interviewer and the participants. Once the interviews were completed, they were transcribed and adapted to a business context. During the transcription process, repetitive words and filler expressions such as "uhmm" and "ah" were excluded as they did not contribute any additional information, aside from indicating that the interviewees took a moment to consider their responses.

Following the transcription, an initial round of coding was performed. In simple terms, coding in this context refers to labeling or attaching tags to specific parts of text that are related to a particular theme or idea that is important in the study. These labels, or codes, help organize and categorize the text passages. This naturally suggests that a "Thematic Analysis" approach was the main mode of qualitative analysis for this study. The coding of the interviews was done so with a mixture of both inductive and deductive approaches. Based off of the pre-requisites and the constructs of the methodology, a pre-set amount of codes were set-up and through the analysis of the interviews, more codes were formed for a more thorough analysis. This involved the creation of 66 distinct codes to represent different themes and concepts derived from the interview data. Through further evaluation and analysis, these codes were subsequently consolidated and grouped, resulting in a reduction to 28 codes. The aim of this coding process was to organize and categorize the data, confirming the various constructs of the methodology. In essence, this step can be likened to the "Confirmation of Theory" phase described in [64].

Subsequently, the codes were analyzed collectively to determine their combined support or refutation of the constructs of the methodology. This qualitative method of validation serves to examine the relationships and patterns within the data, providing insights into the extent to which the constructs align with the interview findings.

## 5.3 Interview results

The interviews had 2 goals, confirm the author's pre-requisites and methodology constructs and to obtain feedback on improvement of the methodology. Some of the interviewee's answers did not require the use of the specific questions due to them being answered in the previous open-ended questions.

### 5.3.1 Single source of truth representation

All the interviewees **emphasize the significance of establishing a representation of single source of truth, albeit in different ways**. The code "Single source of truth" was employed to represent any information derived from the interviews concerning the theme of "*current state of the enterprise*" (As-Is landscape) or a "*comprehensive repository of information*". It is noteworthy that the code "transparency" is frequently associated with the concept of the "As-Is" landscape. To summarize, the main points are as follows:

- Creating transparency and building a repository of information is important.
- Defining the current position and future goals is a challenge that enterprise architecture can address.
- Enterprise architecture helps bring transparency and enables decision-making on how to achieve goals.
- Data quality is crucial, and enterprise architecture collects information from subject matter experts.
- Having an "As-Is" landscape is essential for planning a future IT landscape.
- Enterprise architecture provides insights into application usage, interconnections, and overall IT landscape transparency.

Overall, the interviewees emphasized the significance of transparency, defining goals, data quality, and the role of enterprise architecture in managing and optimizing the IT landscape.

### 5.3.2 Assigned ownerships

All the interviewees **confirmed the pre-requisite of having ownerships assigned to the enterprise architecture elements**. The code "Ownernship" was used to indicate any information that led to the theme of having assigned ownerships to EA elements.

Overall, the interviewees provided valuable insights on the importance of enterprise architecture in managing and optimizing the IT landscape. The key points can be summarized as follows:

- **Clear Role and Responsibility:** It is crucial to establish a clear role and responsibility model for artifacts such as applications, capabilities, processes, and projects. Having designated owners and domain experts helps in understanding requirements and ensuring accountability.
- **Modernization:** Adopting the methodology may facilitate modernization efforts. Regular communication with business domain representatives and incorporating their feedback allows for quicker adjustments and better outcomes.

- **Transparency and Decision-making:** Enterprise architecture plays a vital role in bringing transparency to the IT landscape. It provides the necessary information for informed decision-making, including understanding application ownership, goals to be achieved, and the overall road-map.
- **Data Quality and Subject Matter Experts:** Data quality is of utmost importance, and enterprise architecture relies on subject matter experts to provide accurate information. Collaboration with application owners, solution architects, business domain representatives, and other stakeholders ensures comprehensive and reliable data.
- **Ownership and Governance:** Understanding asset ownership is essential for effective governance and the ability to run initiatives and modernization efforts. Lack of ownership leads to challenges in maintaining and governing assets, hindering progress and optimization.
- **Meaningful Information Maintenance:** Avoiding the maintenance of unnecessary information is crucial to prevent a data graveyard. Information should be maintained with a clear purpose and consumer in mind to ensure efficient use of resources.

### 5.3.3 High-level strategic objectives

All the interviewees confirmed the pre-requisite of having ownerships assigned to the enterprise architecture elements. The code "Objective" was used to indicate any information that led to the theme of having high-level objectives before undertaking any project initiative.

According to the first interviewee, having a high-level objective is beneficial as it helps in understanding the purpose behind the actions taken. Relating projects and transformation initiatives to an overall goal provides clarity and context. However, the interviewee also suggests that a high-level objective may not be a mandatory requirement solely for enterprise architecture.

The second interviewee mentions that one of the first problems they identified in their experience was the lack of clarity regarding the desired destination or goal among the stakeholders of the customer company. Without a clear objective, it becomes challenging to make informed decisions and take appropriate actions. Defining the current state and the desired future state is crucial for understanding the journey that needs to be undertaken. Additionally, they emphasize that digital transformation is not solely about technical improvements but also involves integrating the business aspects. Focusing only on technical aspects, such as migrating to the cloud, without considering the broader business implications, may not lead to desired benefits. Therefore, aligning both technical and business elements is essential for a successful digital transformation.

The third interviewee mentions that it is crucial to establish a clear goal that aligns with the business strategy to obtain buy-in from various stakeholders. By defining the common goal, the stakeholders understand the purpose and significance of the transformation. Additionally, they emphasize that not every change is business critical, and enterprise architects should prioritize and align their actions with the overall business strategy. This underlines the need for strategic alignment and focusing resources on changes that contribute to the overarching objectives.

The fourth interviewee mentions that by setting clear objectives, organizations can avoid being overly focused on technology alone. Starting with objectives allows for a broader perspective and prevents getting confined to specific solutions. It enables a more comprehensive understanding of the overall construct and aids in identifying the best approach for realizing the objectives. This approach is crucial for enterprise architecture, as it encourages taking a step back from technical details and gaining a clearer and more holistic view of the organization's goals

In summary, interviewees emphasize the need to understand the purpose and direction of projects and transformation initiatives. They stress the significance of aligning enterprise architecture with the overall business strategy and having a clear vision of the desired outcomes and **confirm the need for high-level objectives**. The interviewees emphasize the importance of setting higher-level objectives and then breaking them down into actionable steps, rather than focusing solely on technology solutions. Taking a holistic view of the enterprise is considered essential for effective architecture.

### 5.3.4 Stakeholder Management

All the interviewees **confirmed the pre-requisite of needing effective stakeholder management**. The codes "resistance to change", "effective communication", "engage employees" & "strategic support" were grouped to a "Stakeholder Management" code group to effectively manage multiple themes that lead to an overarching theme.

Interviewee 1 highlights the importance of employee engagement and clear communication in the success of a company's digital transformation initiative. While resistance from staff can occur, especially in projects related to enterprise architecture, effective communication of the value and purpose of the transformation is crucial. Engaging employees and gaining their support is essential for the implementation of new applications or system transformations. Without proper change management and addressing potential resistance, the project can face risks and potential failure. Thus, emphasizing the significance of change management and employee involvement in ensuring the success of digital transformation initiatives.

Interviewee 2 emphasizes that employee resistance is a significant factor that can pause, misguide, or even halt digital transformation initiatives. Using the example of cloud migration projects, the interviewee explains the importance of gathering input from various stakeholders, including business, technical, and HR partners. However, if individuals only respond to their specific questions without considering the bigger picture, it can hinder the progress of the project. The interviewee highlights the need for collaboration and buy-in from management, as their support and understanding are crucial for successful implementation. However, despite potential management pressures, ultimately, the success of digital transformation initiatives depends on the willingness and cooperation of the people involved.

Interviewee 3 explains that in the context of implementing transformation projects, one common challenge is encountering political or stakeholder resistance. When transitioning from one application to another, there are often individuals who are emotionally attached to their current application and resist any change. This resistance stems from the belief that the existing application is sufficient and does not require alteration. However, if there is buy-in from key stakeholders or the steering committee consistently approves such changes, it becomes easier to obtain the necessary information and maintain updated data.

Interviewee 4 takes [redacted] Failure to recognize the importance of maintaining and supporting the overall architecture strategy may result in a lack of interest in documenting changes and hinder the success of the implementation.

### 5.3.5 Demand Management

Interviewee 1 suggests that stakeholder goals and needs were not adequately addressed through the use of the demands change backlog in the context of enterprise architecture. They highlight the lack of available options for employees to share feedback or raise demands, such as a dedicated portal. Additionally, there is a lack of awareness about the existence of enterprise architecture within many organizations. The interviewee mentions that initiatives and software purchases are often started without proper channels for feedback and demands from enterprise architecture. They also note a gap between procurement and enterprise architecture processes.

Interviewee 2 shares the idea of having one primary business domain with multiple demand owners divided by business units. The importance of communication and collaboration between different domain business demand owners is highlighted, especially when considering overlaps or dependencies between demands. The interviewee emphasizes the need for the demand owner to be well-connected and have a comprehensive understanding of the enterprise strategy, current state, and future goals.

The interviewee emphasizes that being a demand owner is a highly involved responsibility that should be treated as a full-time job rather than a casual commitment. Simply checking the As-Is landscape and prioritizing tasks would not be sufficient for effective decision-making and implementation. They suggest that being a demand owner should be a full-time job rather than a part-time responsibility to ensure effective prioritization and decision-making or that the demand owner should be replaced with a commitee of demand owners. The interviewee emphasizes the importance of having an accessible demand backlog where everyone can easily access and enter their demands. While having an open backlog may result in a large number of entries, the focus should be on prioritizing effectively rather than restricting the backlog. The interviewee supports the approach of prioritizing demands over limiting the accessibility of the backlog.

Interviewee 3 suggested that there is not enough representation of strategic support in the methodology and the addition of a steering committee maybe a noteworthy inclusion in the demand management section. They stress the importance of a governance procedure for the the evaluation and rejection of initial demands. The interviewee highlights the importance of having well-defined enterprise architecture (EA) processes where each role, such as the demand owner, has ownership of the process and possesses domain expertise to evaluate the demands. This ensures that the evaluation of demands is conducted effectively and efficiently.

Interviewee 4 highlights the importance of aligning the organizational setup and practices in documenting changes in enterprise architecture. The engagement and interest of individuals in documenting their wanted changes depend on their roles and goals within the organization. If they are aware that supporting the overall architecture strategy is part of their responsibility and can benefit them in the long run, they are more likely to actively participate in documentation and collaborate with enterprise architects. It is crucial to communicate the benefits and incentives of following the process to encourage their involvement while considering their own project goals.

In summary from the interviews, several key points can be derived:

1. Stakeholder goals and needs are not effectively addressed through the demands change backlog in enterprise architecture. There is a lack of feedback options and awareness about enterprise architecture within organizations.

2. A primary business domain with dedicated demand owners divided by business units may be necessary. Effective communication and collaboration between these owners are crucial, considering overlaps and dependencies. The demand owner role should be treated as a full-time commitment, with a comprehensive understanding of the enterprise strategy.

3. Strategic support and the inclusion of a steering committee in the demand management function is noteworthy feedback. Well-defined enterprise architecture processes and domain expertise are important for evaluating demands efficiently.

4. Aligning organizational practices and engaging individuals in documenting changes is crucial for effective demand management. Awareness of the benefits and incentives of supporting the architecture strategy motivates active participation and collaboration with enterprise architects, considering individual project goals.

Overall, the interviews highlight the need for improved feedback mechanisms, effective communication, a comprehensive understanding of enterprise goals, well-defined processes, and alignment of organizational practices to enhance the effectiveness of enterprise architecture.

### 5.3.6 Service Orientation

Remembering that a service is defined as a simplified representation of business functionality and/or data presented within a business context in the context of SOA, the interviewees reference service-orientation in the following ways:

**Interviewee 1:**

While not explicitly mentioning service-orientation, they emphasize the importance of introducing new technologies into an organization to impact the business first and improve the customer experience. This aligns with the underlying principles of service-orientation, which aim to enhance business functionality and provide value to customers through technology-enabled services.

**Interviewee 2:**
They highlight the need to integrate the business side of things in a digital transformation initiative. This recognition of integrating business processes aligns with the service-orientation approach, where services are designed to encapsulate business functionality and data within a business context. By integrating the business side, the organization can leverage service-orientation principles to drive improvements and achieve better outcomes.

**Interviewee 3:**
They mention how EA can help in planning a future landscape by considering capabilities, applications, and interfaces. This aligns with service-orientation principles as EAs, along with solution architects, assess and select future solutions that fulfill the business capabilities currently provided by existing solutions. This approach ensures a smooth transition and alignment of services within the enterprise architecture.

**Interviewee 4:**
They explicitly state that enterprise architecture is service-oriented. This reflects the understanding that enterprise architecture, as a discipline, embraces service-orientation principles to design and manage the delivery of services that align with business objectives. By considering the business perspective and focusing on overall objectives, enterprise architects can create a clear and comprehensive understanding of the enterprise's service landscape.

Overall, while the interviewees may not directly refer to service-orientation, their statements **indirectly reflect the principles and concepts associated with service-orientation architecture**, such as integrating business processes and aligning technology solutions with overall business abstractions and to overall objectives.

### 5.3.7 Dependency Management

The interviewees view dependency management as a key use-case of enterprise architecture. This is exemplified in the case of agile EA where changes are made ever more frequently and dependency management becomes a daily task. Enterprise Architecture (EA) plays a crucial role in dependency management by providing overall transparency, adaptability, and integration across various systems and processes. The interview extracts highlight different aspects of how EA facilitates dependency management:

Interviewee 1 uses the analogy of real architecture in a city to emphasize the importance of considering dependencies during the transformation process. Just like forgetting to connect a new penthouse with essential utilities renders it unusable, neglecting to address dependencies in digital transformation can lead to inefficiencies and operational challenges. EA helps identify and visualize these dependencies, ensuring a holistic view of the systems and their interconnectedness.

Interviewee 2 discusses the integration between Jira and [redacted] This integration between a PM tool and an EA management tool enables stakeholders to assess the impact of delays on the overall transformation and make informed decisions about resource allocation and timeline adjustments.

Interviewee 3 emphasizes the importance of understanding the current system's interfaces, data flows, and processes through EA. This knowledge allows for the identification and management of dependencies between different components, interfaces, and processes within the as-is state. By documenting and analyzing these dependencies, EA provides insights into what needs to be implemented in the new system to ensure a seamless transition and avoid disruption due to unaddressed dependencies.

In summary, EA *enables* dependency management by providing transparency, integrating information from various sources, and facilitating a comprehensive understanding of system interfaces, data flows, and processes. These capabilities support effective decision-making, risk mitigation, and successful implementation of digital transformation initiatives.

### 5.3.8 Incremental development

Interviewee 1 believes that adopting a smaller architecture increment approach is definitely better than having a big transformation project. They argue that planning too far ahead, such as three or five years, is not practical because demands and technology will change over time. Instead, they advocate for setting an overall goal and starting with an agile approach. By regularly reviewing progress and determining the next steps, they believe this approach will be more successful than a rigid waterfall approach. In their professional services work, they follow a similar methodology, implementing changes gradually in phases, starting with a subset of applications or focusing on specific locations or business units. They prioritize use cases and gradually expand based on successful drafts and customer needs. Overall, they emphasize the importance of an agile approach in their projects.

Interviewee 2 acknowledges that enterprise architecture has traditionally been approached as a big project rather than addressed step by step. However, they believe that performing smaller architecture increments is a better solution. They describe the current state of enterprise architecture as primarily descriptive, where efforts are focused on bringing the existing IT and business landscape into alignment. They argue that enterprise architecture needs to transition into a more prescriptive dimension, where it proactively identifies areas for improvement and drives change. The interviewee suggests that enterprise architecture should take the lead in identifying dependencies and initiating necessary actions, rather than being reactive to application owners' decisions. They believe that an incremental approach is more suitable because it allows for gathering information, improving data quality, and adapting to the fluid nature of enterprise architecture. They emphasize the importance of detailed enterprise architecture that guides decision-making and suggest breaking down work packages into smaller steps, such as identifying dependencies, proposing solutions, and executing them. In conclusion, the interviewee expresses a preference for incremental approaches in enterprise architecture.

Interviewee 3 suggests that it is important to have a target state and a roadmap in mind when approaching enterprise architecture. They emphasize the need to define the objectives and desired outcomes clearly to avoid potential issues. While the implementation process can be agile and adaptable, the overall solution should be well-defined. The interviewee recommends taking an incremental approach, gradually rolling out the solution rather than pursuing a full-scale implementation all at once. This allows for flexibility in implementing process changes and ensures a more controlled and manageable transition.

Interviewee 4 states that the approach of performing smaller architecture increments or a big transformation depends on the scope of the project. In the case of a large program like an ERP transformation, comprehensive planning is necessary to consider the future and changing circumstances. However, even within a big plan, there are numerous small increments that can be implemented. The interviewee mentions the use of milestone planning, where shifts in one milestone can impact the entire plan. This highlights the need for adaptability and flexibility in managing enterprise architecture projects.

Overall, the interviews **stress the advantages of an agile and incremental approach in enterprise architecture**, while also considering the scope and adaptability required for successful implementation.

### 5.3.9 Other noteworthy takeaways

#### Visualization

All the interviewees note that visualization of the enterprise architecture is a given and do not even question it. However, the organization, [redacted] moves further on from a model-based development into a model-based, data-driven development of enterprise architecture. Modern EA tools [redacted] are advancing the field of EA by allowing Enterprise Architects to make visualize the architecture via the use of data in the form of various reports.

### Demand Prioritization

All the interviewees believe that demand prioritization is a key process in the methodology. The second interviewee even suggested that it is the most key process in the entire methodology.

### Demand Classification

Some of the interviewees believe that the demand classification may be unnecessary, but some of the interviewees believe that any information to provide additional context and transparency is worth including.

### EA Evaluation

The interview participants offered comparable suggestions for assessing the increase in Enterprise Architecture (EA). These suggestions include validating the implementation of use cases following the completion of the design. Two interviewees proposed ensuring adequate representation of testing requirements, while also recommending an additional method of verification through stakeholder feedback. This entails verifying with the "demand requester" whether their problem has been effectively addressed or not.

### Proto-type

[redacted]

## 5.4 Summary

For the purposes of the data analysis of the interview, a thematic analysis process was followed. The main RQ that needed an answer for this section was : *"Does the developed methodology provide a viable approach for designing and implementing enterprise architecture?.* This was split into 2 main objectives and they were achieved:

- to confirm the prerequisites of the AEA-DM and its constructs
- to obtain feedback on improving the methodology.

The first theme discussed was the importance of establishing a representation of a single source of truth, which includes transparency, defining the current state and future goals, data quality, and ownership. Enterprise architecture plays a crucial role in creating transparency, collecting data from subject matter experts, and providing insights into the IT landscape.

The second theme focused on the necessity of assigning ownership to enterprise architecture elements. Clear roles and responsibilities, modernization efforts, transparency, data quality, and governance were highlighted as crucial aspects of effective enterprise architecture.

The third theme emphasized the need for high-level strategic objectives. The interviewees stressed the significance of aligning enterprise architecture with the overall business strategy, setting clear goals, and avoiding a narrow focus on technology alone.

The fourth theme discussed effective stakeholder management, including employee engagement, clear communication, addressing resistance to change, and obtaining strategic support. These factors were identified as critical for the success of digital transformation initiatives.

The fifth theme addressed demand management, including the need for effective feedback mechanisms, a primary business domain with dedicated demand owners, strategic support, and alignment of organizational practices.

Finally, the sixth theme touched on service orientation, highlighting the integration of business processes, planning the future landscape, and embracing service-orientation principles within enterprise architecture.

Overall, the interviews provided valuable insights into the prerequisites, methodology constructs, and improvement opportunities related to enterprise architecture. The themes discussed underscored the importance of transparency, ownership, strategic alignment, stakeholder management, demand management, and service orientation in optimizing the IT landscape and driving successful digital transformation initiatives.

After evaluating the methodology through interviews, a modified Agile Enterprise Architecture Development Method (AEA-DM) was developed to address the feedback from the interviews. Two observations led to the modifications:

- the need for nuanced demand management
- the complexity of demand classification

In larger organizations, a demand management team is necessary, consisting of representatives from various business units, an overall demand owner, and a strategic-level representative. This allows for agile decision-making and strategic insights. The classification of demands as intentional or emergent is replaced with a simpler approach based on the duration of solution design. If it exceeds 1-2 sprints, the team has the flexibility to choose the number of demands to address concurrently. If it can be completed within 1-2 sprints, it is recommended to select 1-2 demands per sprint.

# Chapter 6

# Conclusions

The study's primary focus was the design of a comprehensive but flexible methodology to build an agile enterprise architecture. For this purpose, a research question was designed:

> ***How to design a methodology that satisfies agile practices or principles so that companies can better transform digitally using enterprise architecture?***

This RQ was split into 3 further RQs to split the design problem into manageable endeavors

## 6.1 Agile practices that benefit EA

The objective of the literature review is to explore the development of an agile enterprise architecture methodology that can effectively support an organization's digital transformation and answer the first sub-RQ. The review specifically focuses on addressing the following research question:

> *What are the agile practices that contribute to the management of enterprise architecture during the digital transformation of an organization?*

TABLE 6.1: Agile practices that benefit EA

| | **Description** | **From studies** |
|---|---|---|
| 1 | *Incremental Development* | [35], [39], [42], [15] |
| 2 | *Collaboration* | [59], [15], [47] |
| 3 | *Customer/Stakeholder Representation* | [10], [15], [39], [35] |
| 4 | *Continuous Optimization* | [59], [42] |
| 5 | *Test-Driven Design* | [47], [39] |

To answer the first sub-research question, the literature review explored agile practices that benefit enterprise architecture management during digital transformation. These practices can be seen in the table 6.1.

Furthermore, it has been determined that the agility aspects must be incorporated as essential criteria for *evaluating* the effectiveness of an enterprise architecture in the development of the methodology.

Incremental development is a crucial to achieve agility within enterprise architecture. It refers to the iterative and incremental approach of building and improving systems, software, or solutions in small, manageable steps rather than attempting to deliver a complete and final product all at once. It allows for faster Time-to-Value & ability to adapt to changing demands which are both hugely important for improved EA delivery.

Collaboration is vital for agility in enterprise architecture as it brings together diverse perspectives and expertise to make informed decisions quickly. By involving stakeholders, architects, and developers in open communication, shared understanding of goals and challenges is achieved, leading to faster problem-solving and adaptive solutions. This collaborative approach fosters alignment, reduces misunderstandings, and enables the rapid adjustment of architectural strategies to meet evolving business needs.

Stakeholder representation is crucial for agility in enterprise architecture as it ensures that the architectural decisions align with the diverse needs and priorities of different business units and users. Involving stakeholders provides valuable insights, enabling the architecture to adapt rapidly to changing requirements. This approach fosters buy-in, minimizes potential conflicts, and facilitates the creation of a more responsive and effective architectural framework.

Continuous optimization of enterprise architecture is essential for agility as it enables the architecture to evolve in response to changing business needs, technological advancements, and market dynamics. By regularly reviewing and refining architecture components, processes, and technologies, organizations can ensure that their systems remain efficient, adaptable, and aligned with strategic goals. This iterative optimization approach enhances agility by facilitating quick adjustments and minimizing the risk of architectural stagnation.

Test-driven design of an enterprise architecture is important for agility as it promotes the creation of well-defined and quality assured EA deliverables. By focusing on writing design use-cases and by verifying architecture quality standards before implementing solutions, test driven design of an EA *ensures* that architecture meets business requirements and remains adaptable to changes.

The overall goal of this research being the design of a methodology for the development of an agile EA, the principles in the table 6.1 must be catered to in the design of the method. By considering these agile principles, the resulting methodology can better align with the requirements and challenges of digital transformation within an organization.

## 6.2 A methodology for agile enterprise architecture

The second sub-research question is addressed by the design of the Agile Enterprise Architecture - Development Method (AEA-DM) in the figure 4.2 and its pre-requisites in figure 4.1, which serves as a systematic approach for developing enterprise architecture in an agile manner.

*How can the design or engineering of an enterprise architecture be made agile?*

This method employs a process flow diagram that offers a clear and structured representation of the methodology, facilitating comprehension of the various steps involved. Within the methodology, different team roles are assigned, including the Demand Owner, Architects, and testers responsible for agile testing.

The workflow of the methodology initiates with demand collection, where stakeholders' demands from different business units of the organization are registered in the demand change backlog. This workflow is used to show the *representation of stakeholders* in the method. The demands that successfully pass the assessment are recorded in the Architecture Change Backlog, which serves as input for the Architecture Development/Design Sprint. This sprint encompasses several activities such as Architecture Change Analysis, where demands are analyzed, prioritized, and classified as intentional or emergent changes. Following this, the Design of EA Increment phase focuses on designing the enterprise architecture increment based on the selected demands. Concurrently, *test-driven design* takes place, incorporating activities such as design testing, architecture evaluation, defect management, and deployment management. The sprint culminates in a retrospective and review meeting to discuss achievements and identify areas for improvement. The cycle then restarts with the addition of new demands and defects to the demand change backlog which represents the *incremental development* and *continuous optimization* of an EA. This can only occur if the ownership of EA elements is assigned. Therefore, without the *collaboration* between the architecture development team the owners of different EA elements, this method will not be successful.

By following the Agile Enterprise Architecture - Development Method (AEA-DM), it is hypothesized that organizations can effectively manage and address the evolving demands of stakeholders in an agile manner, fostering continuous improvement and adaptation within the enterprise architecture development process.

## 6.3 Viability of the methodology

*Does the developed methodology provide a viable approach for designing and implementing enterprise architecture?*

To answer the third sub-research question, an evaluation of the AEA-DM was conducted. The evaluation of the methodology was conducted through thematic analysis of interview data, aiming to confirm its viability. The constructs of the methodology are verified by asking the interviewees for their opinion and feedback for improvement. This meant the addressing of two objectives:

- Confirming the prerequisites and constructs of the AEA-DM.
- Obtaining feedback for improving the methodology and build an improved method.

This feedback therefore gives us the answer to the last sub-research question.

Is the version of the AEA-DM shown in 4.2, a viable approach for designing or building an enterprise architecture? **No**. *But, it can be with a few modifications.*

Main takeaways from the interviews and the answer to the last research question are as follows:

- All the interviewees confirmed (directly or indirectly) the importance of the need for the methodology prerequisites, namely, single source of truth representation, assigned ownerships, high-level strategic objectives & stakeholder management. These prerequisites are outlined more in depth in chapter 4
- All the interviewees believe and stress on the importance of an incremental method of architecture development.
- All the interviewees liked the idea of maintaining a "backlog" for the development of an architecture increment. However, the overall aspect of demand management was not adequately addressed in the first iteration of the AEA-DM.
- All the interviewees confirmed (directly or indirectly) the importance of the idea of service orientation in the development of an architecture increment
- All interviewees also confirmed that visualization of an Enterprise Architecture is important
- All interviewees confirm the importance of the prioritization of demands while some considering it the most important step in the method.
- All the interview participants offered comparable suggestions for assessing the architecture increment, namely whether the use-case has been satisfied and by verification with the "demand requester".
- All the interviewees seemed to think that the classification of demands was not important as the prioritization and was needlessly complex.

The analysis confirmed the inclusion of all prerequisites and identified themes that validated the methodology. Valuable feedback from the interviewees was incorporated into the final modified AEA-DM, thereby *hypothesizing* that the developed methodology is now a viable approach for designing and implementing enterprise architecture.

The word "hypothesizing" is used since further verification is needed for practical implementation of the methodology due to the limited sample size of the interviews and lack of empirical data. Observation and long-term empirical testing are necessary to assess effectiveness, adaptability, and address potential limitations.

## 6.4 Improved Agile Enterprise Architecture - Development Method

The feedback generated through the interviews was later incorporated into the new Modified AEA-DM presented in the figure 6.1 thereby answering the main research question. Therefore, with the combination of the 3 sub-research questions, in this subsection, the answer to the main research question mentioned at the beginning of this chapter is answered. For easy readability, here is this research question:

> ***How to design a methodology that satisfies agile practices or principles so that companies can better transform digitally using enterprise architecture?***

In this thesis, a methodology (the modified AEA-DM) is designed through the use of the DSRM which satisfies agile principles, namely, incremental development, continuous optimization (working architecture increments, frequent delivery of these increments), customer/stakeholder representation, test-driven design to ensure quality & collaboration between architecture teams and other stakeholders within the organization and self reflection and continuous improvement. This methodology can be seen in the figure 6.1.

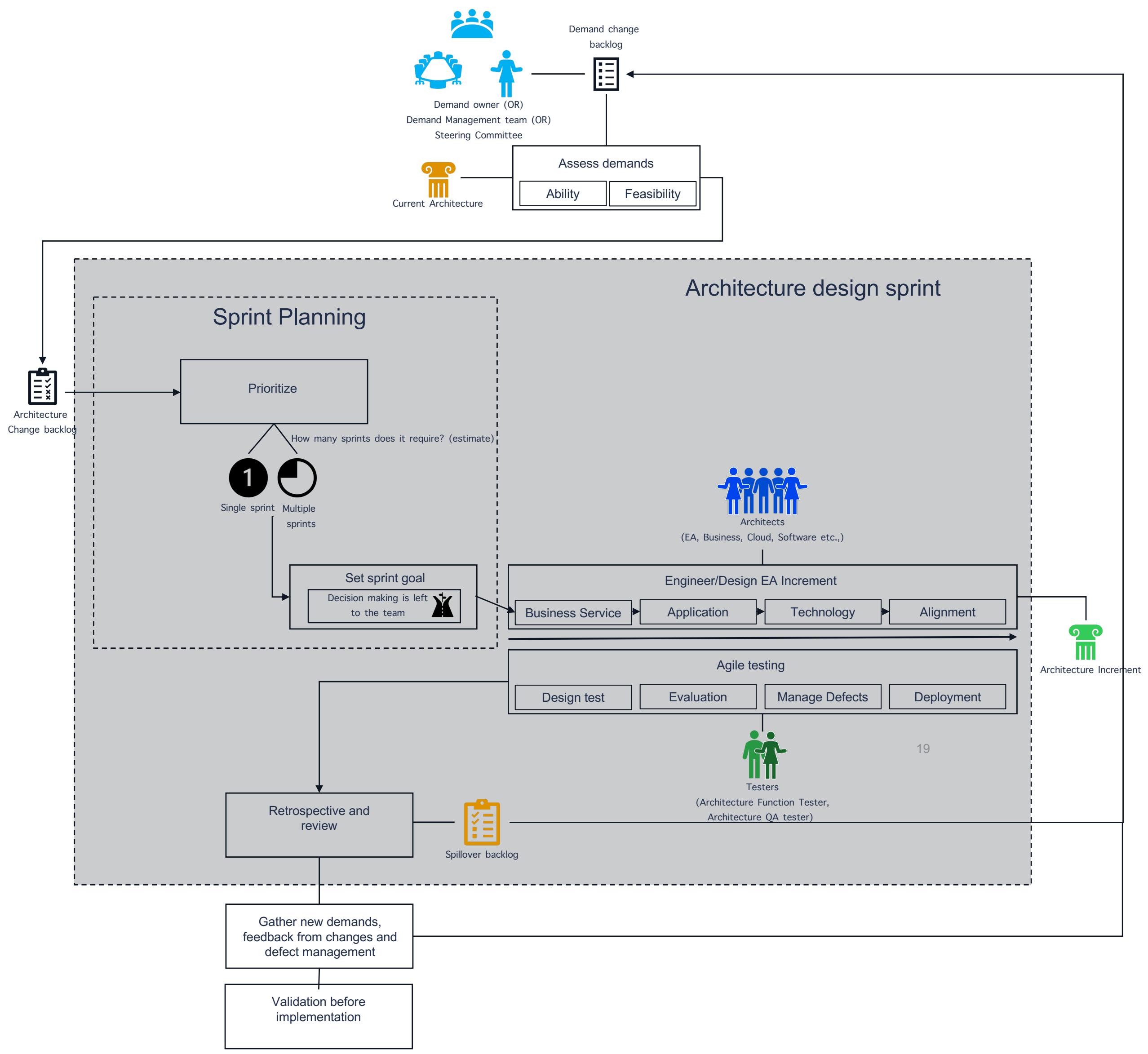


FIGURE 6.1: Modified Agile Enterprise Architecture - Development Method

The modified AEA-DM 6.1 was developed based on two observations:

1. The demand management aspect of the methodology requires nuance.
2. The use of the classification of demands is unclear and is needlessly complex

Each of these changes can be seen in the figure 6.1 and in the figures 6.3 and 6.2.

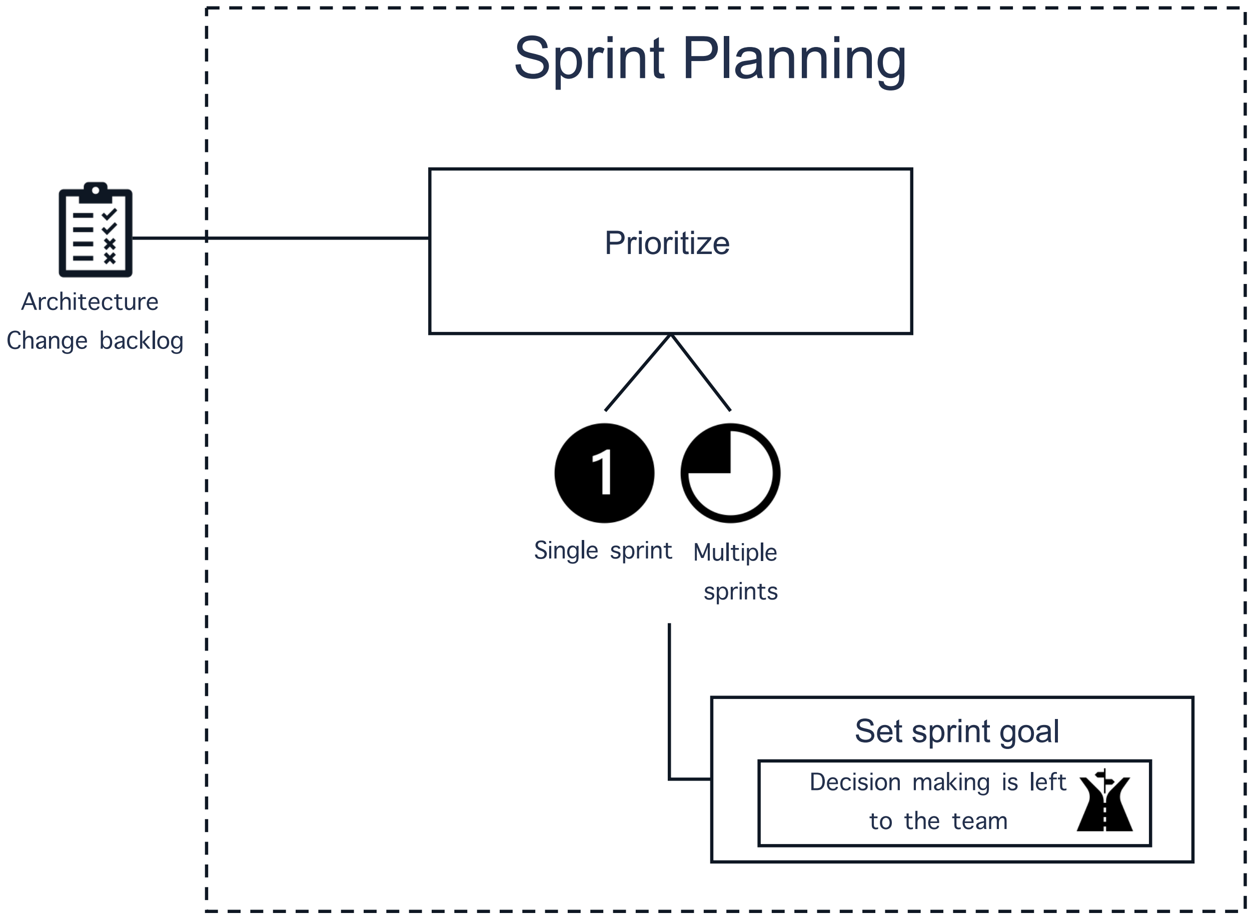


FIGURE 6.2: Sprint planning in the modified AEA-DM

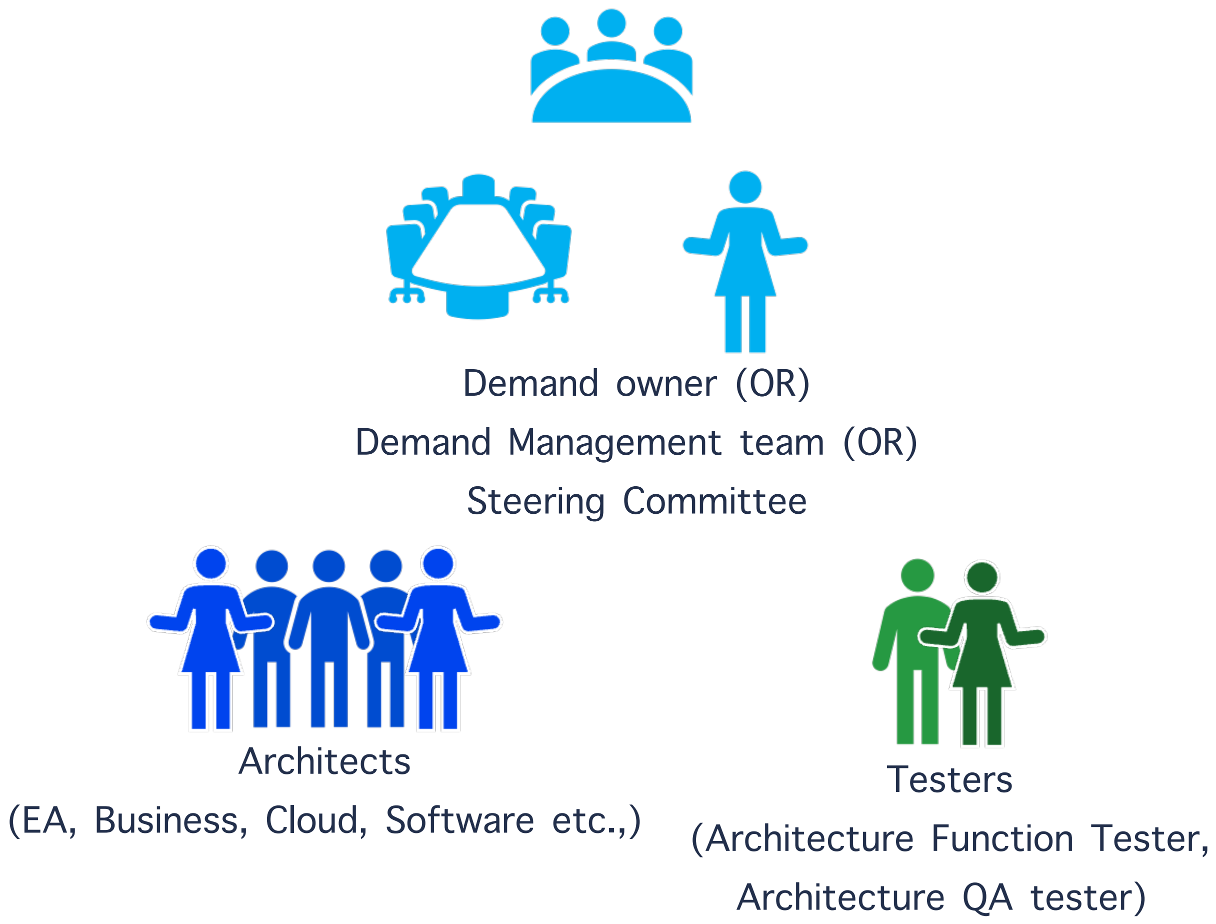
### 6.4.1 Demand Management

The management of demands varies across organizations. While a single demand owner may suffice for smaller, closely-knit organizations, it is inadequate for larger entities. Therefore, the introduction of a **demand management team** becomes necessary. This team comprises individuals from all business units or owners of business functions within the enterprise, an overall demand owner to talk to and communicate with the architecture development team, and a representative from the strategic level of the organization.

The inclusion of a strategic-level representative was deemed essential based on feedback from interviewee 3. However, seeking permission from a strategic level for demands can be perceived as too rigid and waterfall-like. To maintain agility, having a team member from the strategic level who is considered an equal member of the demand management team enables prompt decision-making and provides valuable strategic insights to the methodology. The new team structure can be viewed in the figure 6.3 in the Figures section of the Appendix.

### 6.4.2 Sprint Planning

The classification of demands into intentional and emergent categories is considered excessively intricate and is therefore discarded in favor of a simpler decision-making approach. The simplified decision is



Demand owner (OR)
Demand Management team (OR)
Steering Committee
Architects
(EA, Business, Cloud, Software etc.,)
Testers
(Architecture Function Tester,
Architecture QA tester)

FIGURE 6.3: Team structure in the modified AEA-DM

based on the following question: "Does the design of the solution to address a demand require more than 1-2 sprints?" (assuming a sprint duration of 2 weeks to a month). If yes, the choice to develop more than one demand is left to the team's discretion. This approach is taken under the assumption that the team's expertise is sufficient to gauge the effort needed to address a demand or multiple demands. The team may opt to focus on a single demand for as long as necessary or concurrently undertake solution design for multiple demands. However, if the design can be completed within 1-2 sprints, it is *recommended* (not mandated) to select only 1-2 demands to be addressed in each sprint. The modified sprint planning stage can be viewed in the figure 6.2 in the Figures section of the Appendix.

### 6.4.3 Demonstration of the modified AEA-DM

[redacted]

[redacted]

## 6.5 Limitations

While this research has yielded valuable insights into the design and development of an agile enterprise architecture methodology, certain limitations should be acknowledged.

Firstly, the research artifact underwent qualitative review through a limited number of interviews, and the time-frame was constrained to a few months, preventing extensive real-life implementation and validation via case studies.

Secondly, the *intentional generality* of the methodology may require customization for specific industry types to enhance its applicability.

Thirdly, the methodology lacks a specific focus on the implementation of architecture increments in organizations, potentially introducing complexities during adoption and execution.

Furthermore, transitioning organizations with traditional enterprise architecture mindsets to embrace the new methodology requires a deliberate effort. Promoting an agile mindset throughout the organization and making stakeholders aware of the demand management pipeline are essential steps. Demonstrating the benefits of the new architecture development approach for their work and overall advantage is crucial. Adoption of modern enterprise architecture tools may facilitate this endeavor.

## 6.6 Future Work

Moving forward, future work should focus on addressing these limitations and further improving the methodology. To strengthen its validity, conducting both *quantitative* and *qualitative* reviews involving a larger sample size and incorporating observation as a research method can provide deeper insights into its practical implementation and effectiveness within organizations.

Additionally, conducting empirical testing over an extended period through multiple iterations of the design cycle will help evaluate the long-term adaptability and efficacy of the methodology.

Furthermore, exploring the *implementation* of agile enterprise architecture (EA) within organizations is crucial. Understanding the adoption, success factors, and barriers of agile EA can provide valuable insights for organizations seeking to embrace digital transformation.

A promising area for future research is delving into agile testing within the context of enterprise architecture. While quality attributes of EA have received attention in previous studies, agile testing in EA is still in its nascent stages and has not been extensively explored. Investigating the integration of agile testing practices into the EA development process can enhance the overall quality and reliability of enterprise architecture.

Another area of future research could investigate how IT procurement can leverage the new agile enterprise architecture -development method. This integration between procurement and the EA function of a company is crucial for ensuring that technology purchases align with the overall business strategy and technical roadmap. This alignment could help in optimizing IT investments, reducing redundancies, and enhancing the overall efficiency and effectiveness of IT operations.

In summary, future work should focus on quantitatively and qualitatively reviewing the methodology through a larger sample size and observation. Long-term empirical testing and validation are essential to evaluate the effectiveness of the methodology. Additionally, investigating the implementation of agile EA within organizations and exploring agile testing practices in the context of EA are promising areas for further research. These efforts will enhance the practical applicability and understanding of agile enterprise architecture and contribute to the advancement of digital transformation practices in organizations.

# Bibliography

[1] *16th state of agile report.* 2022. URL: https://digital.ai/resource-center/analyst-reports/state-of-agile-report/.

[2] J.F. Abrantes and G.H. Travassos. “Common agile practices in software processes”. In: *International Symposium on Empirical Software Engineering and Measurement* (2011). Included, pp. 355–358. DOI: 10.1109/esem.2011.47.

[3] F. Ahlemann et al. *Strategic Enterprise Architecture Management.* Jan. 2012. ISBN: 978-3-642-24222-9. DOI: 10.1007/978-3-642-24223-6.

[4] B. Anthony Jnr et al. “Digital transformation with enterprise architecture for smarter cities: a qualitative research approach”. In: *Digital Policy, Regulation and Governance* 23 (4 2021), pp. 355–376. DOI: 10.1108/DPRG-04-2020-0044.

[5] *Applying the TOGAF® ADM using Agile Sprints.* Tech. rep. Apr. 2022. URL: https://pubs.opengroup.org/togaf-standard/guides/agile-sprints.html.

[6] A. Azanha et al. “Agile project management with Scrum: A case study of a Brazilian pharmaceutical company IT project”. In: *International Journal of Managing Projects in Business* 10 (1 2017), pp. 121–142. DOI: 10.1108/IJMPB-06-2016-0054.

[7] O. Al-Baik and J. Miller. “The kanban approach, between agility and leanness: a systematic review”. In: *Empirical Software Engineering* 20 (6 2015), pp. 1861–1897. DOI: 10.1007/s10664-014-9340-x.

[8] O.E. Balcicek, M. Gundebahar, and S. Cekerekli. “An agile approach for converting enterprise architectures”. In: *2013 The International Conference on Technological Advances in Electrical, Electronics and Computer Engineering, TAEECE 2013* (2013). This paper is bullshit, can take some statements from it, but most of them are not backed by any data and are just plain statements., pp. 380–386. DOI: 10.1109/TAEECE.2013.6557305.

[9] H. Barbazange et al. *Agile Enterprise Architecture in the Digital age.* July 2018. URL: https://www2.mega.com/en/lp/agile-enterprise-architecture-in-digital-age.

[10] K. Beck and C. Andres. *Extreme Programming Explained: Embrace Change (2nd Edition).* Addison-Wesley Professional, 2004. ISBN: 0321278658.

[11] K. L. Beck et al. *Manifesto for Agile Software Development.* 2001.

[12] J. Bloomberg. *The Agile Architecture Revolution: How Cloud Computing, REST-Based SOA, and Mobile Computing Are Changing Enterprise IT.* 1st. Wiley Publishing, 2013. ISBN: 1118409779. DOI: 110.5555/2829282.

[13] B. Boehm. “Get ready for agile methods, with care”. In: *Computer* 35.1 (2002), pp. 64–69. DOI: 10.1109/2.976920.

[14] S. Bondar et al. “Agile Digitale Transformation of Enterprise Architecture Models in Engineering Collaboration”. In: *Procedia Manufacturing* 11 (2017), pp. 1343–1350. DOI: 10.1016/j.promfg.2017.07.263.

[15] S. Buckl et al. “Towards an agile design of the enterprise architecture management function”. In: *Proceedings - IEEE International Enterprise Distributed Object Computing Workshop, EDOC* (2011), pp. 322–329. DOI: 10.1109/EDOCW.2011.33.

[16] M. Buschle et al. "Automating Enterprise Architecture Documentation using an Enterprise Service Bus". In: *Americas Conference on Information Systems*. 2012.

[17] G. Chin. "Agile project management". In: *AMACOM, New York* (2004).

[18] Luisanna Cocco et al. "Simulating kanban and scrum vs. waterfall with system dynamics". In: *Agile Processes in Software Engineering and Extreme Programming: 12th International Conference, XP 2011, Madrid, Spain, May 10-13, 2011. Proceedings 12*. Springer. 2011, pp. 117–131.

[19] T. M. Czerepak, T. Świerszcz, and A. Sobczak. *Lean Architecture Framework, version 1.4*. 2020. URL: https://lafinstitute.org/.

[20] W. Daoudi, K. Doumi, and L. Kjiri. "Adaptive Enterprise Architecture: Towards a model". In: *ACM International Conference Proceeding Series* (2020). DOI: 10.1145/3447568.3448539.

[21] P. Diebold and M. Dahlem. "Agile practices in practice - A mapping study". In: *ACM International Conference Proceeding Series* (2014). Included. DOI: 10.1145/2601248.2601254.

[22] Z. Dragičević and S. Bošnjak. "Agile architecture in the digital era: Trends and practices". In: *Strategic Management* 24 (Jan. 2019), pp. 12–33. DOI: 10.5937/StraMan1902011D.

[23] T. Dybå and T. Dingsøyr. "Empirical studies of agile software development: A systematic review". In: *Information and Software Technology* 50 (9-10 2008). Included, pp. 833–859. DOI: 10.1016/j.infsof.2008.01.006.

[24] J. Erickson, K. Lyytinen, and K. Siau. "Agile Modeling, Agile Software Development, and Extreme Programming: The State of Research". In: *J. Database Manag.* 16 (Oct. 2005), pp. 88–99. DOI: 10.4018/jdm.2005100105.

[25] H. C. Esfahani, E. Yu, and M.C. Annosi. "Capitalizing on empirical evidence during agile adoption". In: *Proceedings - 2010 Agile Conference, AGILE 2010* (2010). Included, pp. 21–24. DOI: 10.1109/AGILE.2010.7.

[26] H.C. Esfahani and E. Yu. "A repository of agile method fragments". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 6195 LNCS (2010). Included, pp. 163–174. DOI: 10.1007/978-3-642-14347-2_15.

[27] Inc. Gartner. *Gartner glossary - Enterprise architecture (EA)*. URL: https://www.gartner.com/en/information-technology/glossary/enterprise-architecture-ea.

[28] Inc. Gartner. *Gartner glossary - IT governance*. URL: https://www.gartner.com/en/information-technology/glossary/it-governance.

[29] LeanIX GmbH. *Enterprise Architecture Management - Manage the transformation and risk of your IT landscape*. https://www.leanix.net/en/products/enterprise-architecture-management. Accessed: 2023-06-23.

[30] D. Gonçalves, L. Ferreira, and N. Campos. "Enterprise architecture for high flexible and agile company in automotive industry". In: *Procedia Computer Science* 181 (Jan. 2021), pp. 1077–1082. DOI: 10.1016/j.procs.2021.01.303.

[31] J. Gothelf. *Safe? agile? safe is not agile. what you need to know*. July 2022. URL: https://jeffgothelf.com/blog/safe-is-not-agile/.

[32] D. Greefhorst and E. Proper. *Architecture Principles: The Cornerstones of Enterprise Architecture*. The Enterprise Engineering Series. Springer, Jan. 2011. ISBN: 978-3-642-20278-0. DOI: 10.1007/978-3-642-20279-7.

[33] The Open Group. *The TOGAF® Standard, 10th Edition*. The Open Group, Apr. 2022. URL: https://publications.opengroup.org/c220.

[34] The Open Group. *The TOGAF® Standard, Version 9.2*. The Open Group, 2018. URL: https://pubs.opengroup.org/architecture/togaf9-doc/arch/.

[35] S. Hanschke, J. Ernsting, and H. Kuchen. "Integrating agile software development and enterprise architecture management". In: *Proceedings of the Annual Hawaii International Conference on System Sciences* 2015-March (2015), pp. 4099–4108. DOI: 10.1109/HICSS.2015.492.

[36] E. Hasnain. "An overview of published agile studies: A systematic literature review". In: *ACM International Conference Proceeding Series* Par F12882 (2010). Included. DOI: 10.1145/1890810.1890813.

[37] M. Hauder et al. "Agile Enterprise Architecture Management: An Analysis on the Application of Agile Principles". In: *Software Engineering for Business Information Systems* (Jan. 2014), pp. 38–46. DOI: 10.5220/0005424100380046.

[38] A. Henriksen and S. Pedersen. "A qualitative case study on agile practices and project success in agile software projects". In: *The Journal of Modern Project Management* 5 (May 2017), pp. 62–73. DOI: 10.19255/JMPM01306.

[39] E. Hosiaisluoma et al. "Lean enterprise architecture method for value chain based development in public sector". In: *Proceedings of the European Conference on e-Government, ECEG* 2018-Octob (2018), pp. 86–94.

[40] M. Hron and N. Obwegeser. "Scrum in practice: An overview of Scrum adaptations". In: *Proceedings of the Annual Hawaii International Conference on System Sciences* 2018-Janua (2018), pp. 5445–5454.

[41] ISO Central Secretary. *Software, systems and enterprise — Architecture description*. en. Standard ISO/IEC/IEEE 42010:2022. Geneva, CH: International Organization for Standardization, Nov. 2022. URL: https://www.iso.org/standard/74393.html.

[42] T. Kaddoumi and M. Watfa. "A foundational framework for agile enterprise architecture". In: *International Journal of Lean Six Sigma* ahead-of-print (June 2021). DOI: 10.1108/IJLSS-03-2021-0057.

[43] T. Kaddoumi and M. Watfa. "A proposed agile enterprise architecture framework". In: *2016 6th International Conference on Innovative Computing Technology, INTECH 2016* (2017), pp. 52–57. DOI: 10.1109/INTECH.2016.7845126.

[44] W.H. Kee. "Future implementation and integration of Agile methods in software development and testing". In: *2006 Innovations in Information Technology, IIT* (2006). DOI: 10.1109/INNOVATIONS.2006.301945.

[45] B. Kitchenham and S. Charters. "Guidelines for performing Systematic Literature Reviews in Software Engineering". In: 2 (Jan. 2007).

[46] E. Lander and J. Liker. "The Toyota Production System and art: Making highly customized and creative products the Toyota way". In: *International Journal of Production Research - INT J PROD RES* 45 (Aug. 2007), pp. 3681–3698. DOI: 10.1080/00207540701223519.

[47] M. Lankhorst and H. Proper. "Agile Architecture". In: Mar. 2012, pp. 41–57. ISBN: 978-3-642-28187-7. DOI: 10.1007/978-3-642-28188-4_3.

[48] Y. Masuda et al. "An adaptive enterprise architecture framework and implementation: Towards global enterprises in the era of cloud/mobile IT/digital IT". In: *International Journal of Enterprise Information Systems* 13 (3 2017), pp. 1–22. DOI: 10.4018/ijeis.2017070101.

[49] P. Medeiros et al. "An Agile Approach for Modeling Enterprise Architectures". In: *International Conference on Enterprise Information Systems, ICEIS - Proceedings* 2 (2021), pp. 659–670.

[50] I. Mergel, N. Edelmann, and N. Haug. "Defining digital transformation: Results from expert interviews". In: *Government Information Quarterly* 36 (June 2019), p. 101385. DOI: 10.1016/j.giq.2019.06.002.

[51] M. Mirsalari Se. R.and Ranjbarfard. "A model for evaluation of enterprise architecture quality". In: *Evaluation and Program Planning* 83 (2020), p. 101853.

[52] M.E. Moreira. “Being agile: Your roadmap to successful adoption of agile”. In: *Being Agile: Your Roadmap to Successful Adoption of Agile* 9781430258 (2013), pp. 1–255. DOI: 10.1007/978-1-4302-5840-7.

[53] S. Nadhamuni et al. “Driving digital transformation of comprehensive primary health services at scale in India: an enterprise architecture framework”. In: *BMJ Global Health* 6 (July 2021). DOI: 10.1136/bmjgh-2021-005242.

[54] M. Negreiro and T. Madiega. June 2019. URL: https://www.europarl.europa.eu/RegData/etudes/BRIE/2019/633171/EPRS_BRI(2019)633171_EN.pdf.

[55] S. Nerur, R. Mahapatra, and G. Mangalaraj. “Challenges of Migrating to Agile Methodologies”. In: *Commun. ACM* 48.5 (May 2005), pp. 72–78. ISSN: 0001-0782. DOI: 10.1145/1060710.1060712. URL: https://doi.org/10.1145/1060710.1060712.

[56] K. Peffers et al. “A design science research methodology for information systems research”. In: *Journal of Management Information Systems* 24 (Jan. 2007), pp. 45–77.

[57] M. Rimol. *Enterprise architecture enables digital innovation.* Tech. rep. Gartner, Inc., Feb. 2021.

[58] J. W. Ross. “Enterprise architecture: Depicting a vision of a firm”. In: (Mar. 2004).

[59] B.D. Rouhani et al. “Presenting a framework for agile enterprise architecture”. In: *Proceedings of the 2008 1st International Conference on Information Technology, IT 2008* (2008). DOI: 10.1109/INFTECH.2008.4621684.

[60] O. Salo et al. *How to create an agile organization.* Tech. rep. McKinsey & Company, Oct. 2017. URL: https://www.mckinsey.com/capabilities/people-and-organizational-performance/our-insights/how-to-create-an-agile-organization#/.

[61] S. Sararuch, P. Wannapiroon, and P. Nilsook. “Dimensions of Agile Enterprise Architecture”. In: *Proceedings - 2022 Research, Invention, and Innovation Congress: Innovative Electricals and Electronics, RI2C 2022* (2022), pp. 304–309. DOI: 10.1109/RI2C56397.2022.9910275.

[62] B. Schiano. “Agile and DevOps in the context of enterprise architecture and IT architecture”. In: July 2020, pp. 76–90. ISBN: 9781351037785. DOI: 10.4324/9781351037785-5.

[63] K. Schwaber. “SCRUM Development Process”. In: *Business Object Design and Implementation.* Springer London, 1997. ISBN: 978-1-4471-0947-1.

[64] C.B. Seaman. “Qualitative methods in empirical studies of software engineering”. In: *IEEE Transactions on Software Engineering* 25.4 (1999), pp. 557–572. DOI: 10.1109/32.799955.

[65] L. Sebola and L. Khoza. “THE BENEFITS AND CHALLENGES OF SCALED AGILE FRAMEWORK IN FINANCIAL INSTITUTIONS”. In: 10 (Sept. 2022), pp. 30–43. URL: https://journalmodernpm.com/manuscript/index.php/jmpm/article/view/502.

[66] A. Shrivastava et al. “A Systematic Review on Extreme Programming”. In: *Journal of Physics: Conference Series* 1969 (1 2021). Not included because this focuses on Extreme Programming only and not on other available agile methods. i.e., It is not a systematic review of agile methods. DOI: 10.1088/1742-6596/1969/1/012046.

[67] C.G. Silva, E.L.F. Filho, and L.M. Fontoura. “Software processes used in university, government, and industry: A systematic review”. In: *ICEIS 2020 - Proceedings of the 22nd International Conference on Enterprise Information Systems* 2 (2020), pp. 314–321.

[68] M. Steinhorst. “iARIS - Supporting Enterprise Transformation Using An Iterative ISD Method”. In: *European Conference on Information Systems.* 2013.

[69] *The Digital Practitioner Body of Knowledge™ Standard (the DPBoK™ Standard).* The Open Group, Jan. 2020. URL: https://publications.opengroup.org/c196.

[70] *The safe delusion.* Mar. 2023. URL: https://safedelusion.com/.

[71] Theo Thesing, Carsten Feldmann, and Martin Burchardt. “Agile versus Waterfall Project Management: Decision Model for Selecting the Appropriate Approach to a Project”. In: *Procedia Computer Science* 181 (Jan. 2021), pp. 746–756. DOI: 10.1016/j.procs.2021.01.227.

[72] B.V Thummadi, V.D. Khapre, and R.J Ocker. “Unpacking Agile Enterprise Architecture Innovation work practices: A Qualitative Case Study of a Railroad Company”. In: *Americas Conference on Information Systems*. 2017.

[73] J. R. Venable, J. Pries-Heje, and R. Baskerville. “Choosing a Design Science Research Methodology”. In: *ACIS*. 2017.

[74] R.M.v Wessel, P. Kroon, and H.J. de Vries. “Scaling Agile Company-Wide: The Organizational Challenge of Combining Agile-Scaling Frameworks and Enterprise Architecture in Service Companies”. In: *IEEE Transactions on Engineering Management* (Dec. 2021), pp. 1–14. DOI: 10.1109/TEM.2021.3128278.

[75] *What is SOA?* URL: https://www.ibm.com/topics/soa.

[76] R.J Wieringa. *Design science methodology for information systems and software engineering*. Undefined. 10.1007/978-3-662-43839-8. Springer, 2014. ISBN: 978-3-662-43838-1. DOI: 10.1007/978-3-662-43839-8.

[77] J. A. Zachmann. “A framework for information systems architecture”. In: *IBM Systems Journal* 26.3 (1987), pp. 276–292. DOI: 10.1147/sj.263.0276.

[78] X. Zhu, S. Ge, and N. Wang. “Digital transformation: A systematic literature review”. In: *Computers and Industrial Engineering* 162 (2021). DOI: 10.1016/j.cie.2021.107774.

# Appendix A

# Tables

Table 1: Agile practices from [2] & [21]

| | Agile practice | Definition |
|---|---|---|
| 1 | *Whole team / Unattached communicative teams / Small cross-functional teams* | To ensure the success of a team, it is important to incorporate all essential skills and perspectives, foster a strong sense of teamwork, and promote a shared purpose among all team members. This can be achieved by involving customers, end-users, and other business stakeholders directly in the project. |
| 2 | *Project Visibility / Common knowledge* | Efforts should be made to promptly inform the teams of the project's current status and offer feedback. This can be achieved by setting up web-based project dashboards that display real-time updates and metrics on the project's progress. These dashboards should also feature the project's progress in relation to the user stories that the teams have agreed to deliver at the end of each iteration. |
| 3 | *Continuous integration* | It is advisable for team members to frequently integrate their work upon completing new tasks or changes to identify integration issues and detect system failures at the earliest possible stage. |
| 4 | *Test-Driven Development / Quality check* | Before writing production code, programmers should write test cases. Developers should write unit tests before coding and encourage customers to write acceptance test cases. Initially, the test case will fail as there is no corresponding code that implements the particular feature or condition being tested. Then, the developer writes just enough code to pass the test or satisfy the current goal, followed by refactoring to improve readability and remove duplications. |
| 5 | *Refactoring* | Improve an existing code base by making it more comprehensible, maintainable and enhancing its design without changing its external behavior or functionality. Refactoring comes in various forms such as simplifying complex statements, creating reusable code by abstracting common solutions, and eliminating duplicate code. During the refactoring process, it is important to ensure that all unit test cases in the code base are still passed. |
| 6 | *Pair programming* | This practice requires that two individuals collaborate on writing all source code by working simultaneously on the same computer. The programmers take turns as the driver and navigator throughout the day. |

| | | |
|---|---|---|
| 7 | *Planning game / Validation* | A collaborative process between developers and customers, where the latter selects the essential User Stories for a brief, gradual delivery. After the customer has accepted and tested each short implementation increment, the remaining User Stories are reviewed for potential changes in requirements or priorities. The Planning Game is then repeated for the following implementation increment. This planning process is ongoing and cumulative. |
| 8 | *Simple design* | Creating a solution that is uncomplicated and can be executed at the present moment. It is important to eliminate any unnecessary complexity or superfluous code as soon as possible. Additional features should only be added if they are justified. Predicting future needs is discouraged, and programmers should concentrate on solving present-day problems using the most cost-effective development approach. |
| 9 | *Small releases / Frequent releases* | The practice of frequent software releases aims to shorten the feedback loop from customers by reducing release cycles. Since requirements can change, it is important to keep release cycles brief and ensure that each release provides valuable business solutions to the customer. To achieve this, an initial version of the system is deployed quickly, followed by iterative releases. At the end of each release, the customer reviews the product, identifies defects, and adjusts requirements for future releases. |
| 10 | *Metaphor* | The aim of this practice is to establish a clear, shared story that provides developers and customers with a common understanding of the system's core functionality. This story serves as a high-level software architecture, in a way. When choosing a suitable metaphor, developers must broaden their perspective and analysis of the application under development. |
| 11 | *Collective Code Ownership* | The code repository should be open to all programmers with no restrictions on making changes to any part of the code at any time without seeking permission from anyone. Any pair of programmers can add value to any portion of the code whenever they see an opportunity. |
| 12 | *On-site customer Customer involvement* | To foster effective collaboration and ensure accurate and prioritized requirements, this practice involves having a customer representative as a member of the development team. |
| 13 | *Sustainable Pace / Time boxing* | The focus of this practice is on working efficiently and sustainably by limiting the number of work hours. To avoid the need for overtime, requirements should be selected for each iteration that can be accomplished within the regular working hours. |
| 14 | *Open workspace* | The practice requires developers to work in a shared workspace, preferably a large room with small cubicles, and to place pair programmers in the center of the space. The workplace should also have common areas that encourage open communication among team members. |
| 15 | *Coding standards* | To ensure consistency and readability among team members, coding standards which are easy to follow and collaboratively decided are essential as developers code different system parts with various team members. |

|  |  |  |
|---|---|---|
| 16 | *Product Backlog / Product vision* | Create and manage a constantly updated and prioritized list of business and technical requirements for the system being built or improved that guides development based on current understanding. |
| 17 | *Stand-up meetings / Planning meetings / Daily discussions* | These are brief daily meetings, usually lasting around 15 minutes, that are held to monitor project progress, address any issues, and coordinate daily tasks. During the meeting, each team member gives a quick update on their recent work and progress made. |

TABLE 2: Kanban elements from [7]

|  | **Agile practice** | **Definition** |
|---|---|---|
| 1 | *Kanban method* | Applied to current work-flow. It suggests an incremental and evolutionary approach, and emphasizes the need for ongoing review and improvement of the method. |
| 2 | *Kanban board* | The Kanban board visualizes the workflow of creating and delivering a product/service. Scholars suggest different board designs, including a design board based on existing activities, the traditional waterfall approach or a simpler design with three queues - To Do, In Progress, and Done. |
| 3 | *Pull system* | It is the development process that begins with a customer request. This approach also requires setting a limit on work in progress (WIP) to ensure that work items are pulled only if the WIP limit has not been reached. It is also recommended to pull the highest priority items whenever possible. |
| 4 | *Prioirtized queue* | Scholars have recommended various methods for setting priority criteria in the prioritized queue of the Kanban approach. For instance, some suggest prioritizing work items based on their value, urgency, and importance, while others prioritize based on the cost of delay or resource availability. Some also suggest having separate queues for each process on the Kanban board. |
| 5 | *Inclusion criteria* | This element is one of the main criteria of the Kanban approach, ensuring that work added to the board will provide value to the customer. |
| 6 | *Work-in-progress (WIP)* | The maximum number of work items that can be concurrently processed in each stage of the workflow is known as WIP. This is a key feature of the Kanban method and serves as its primary objective. |
| 7 | *Done item* | A work item that is completed |
| 8 | *Reverse item* | A work item that is being moved backward on the Kanban board, instead of moving forward. This decisioin must be made after consultation with customers as this is a de-prioritization of an item |
| 9 | *Validated learning* | The process of validating a feature's value from a business perspective after it has been completed. |

| | | |
|---|---|---|
| 10 | *Cycle-time/Lead-time* | The determinant for the process efficiency and effectiveness in measuring the performance of Kanban. It is the time elapsed from work start to end to produce a feature or the time between deliveries of items. |
| 11 | *Performance measurement tool* | Measuring the performance of Kanban involves using a cumulative flow diagram based on cycle time and WIP according to some studies, while others recommend daily tracking of progress using burn-down charts. |
| 12 | *Bottleneck* | A bottleneck refers to a point of congestion where the processing of work items cannot keep up with the pace of production from the preceding point. These bottlenecks can be identified through workflow visualization or by establishing a measure adopted from the theory of constraints. |
| 13 | *Slack / Buffer* | The buffer serves several purposes, such as determining the waiting time for work items in the queue before processing them, handling the variation in a process's cycle time, and minimizing bottlenecks. |
| 14 | *Waste identification* | In summary, waste is anything that does not contribute to the valuecreation for the customer; hence, it should be removed. Kanban reduces waste and improves responsiveness to customer requests by identifying and eliminating wasteful activities. |
| 15 | *Team collaboration* | Kanban facilitates communication between team members as well as between team members and senior management. |
| 16 | *Meeting structure* | There is a debate between **Researcher**s of whether meetings are a good use of time or not. However, 15-min daily stand-up meetings are recommended. It is important to note that some **Researcher**s advocate of the meeting length and scope to be decided on the fly by team members. |
| 17 | *Avatars* | They are visual representations of the task owner who is responsible for the respective task |
| 18 | *Planning and estimation* | Planning and retrospective activities are considered important, as they provide an opportunity for improvement through continuous feedback from team members. |
| 19 | *Policies* | Governing the Kanban approach. They are the dominant guidelines that outline what work should be done and how it should be done. |
| 20 | *Feedback loop* | The main objective of delivering a product through short cycles and frequent builds is to receive customer feedback. While **Researcher**s emphasized the importance of a feedback loop, they did not sufficiently address how Kanban could facilitate customer involvement in providing feedback. |

Table 3: Construct definition as seen in [59]

| Construct | Description |
|---|---|
| *Business Projections, Resources, and Supporting Technologies* | Responsible for creating the future operating system. This model is crucial for company projects and allows for **reusing** systems. Its main benefit is supporting system development with available resources. |
| *The Technical and Functional Anticipation* | Anticipates emergent customer requirements and technologies that are likely to impact the firm in the near future, and feeds this information to the "Business Projection, Resources and Supporting Technologies" construct. |
| *The Process Monitoring and Continuous Optimization* | Detects divergences and measures performance components in organizational structures. Its two main missions are to improve processes and feed information to the "Business Projection, Resources and Supporting Technologies Model" construct. |
| *Adaptation of Competencies and Collaboration Types* | This construct implements an agile framework for training and communication, improving HR competencies, motivation, and collective intelligence. It supports the objectives of the "Process Monitoring and Continuous Optimization" model. |
| *Information Systems and Technological Systems* | Covers the operation and control of IT systems using standardized models such as ITIL and Cobit, as well as new technologies for information processing and communication, industrial data processing, and other forms of automation. |
| *Processes Operation* | Covers rules for current process operations and their execution structure, whether manual or automated through the "Information Systems and Technological Systems Model" construct. |
| *Design Rudimentary Model* | Creating basic designs using prognoses and viewpoints, presenting them in meetings to formalize plans, sending them to process operation models, and reducing project risks. The stakeholder agreement is necessary to implement the plan, which helps formalize future activities. The input for this model includes resource project prognoses and protector technological models. |

TABLE 4: Flow description as seen in [59]

| Flow | Description |
|---|---|
| 1 | Feedback on functional/technical incidents during process operation is directed to resources responsible for corrective maintenance of data processing and information systems, using a light "anomaly correction" work-flow |
| 2 | If there are differences between what was planned for a process and what is actually happening during the process, this information is sent to the people responsible for predicting how the process will evolve over time. This helps them to prepare for any changes or challenges that may arise. The information is shared in a special online space where everyone can work together to predict and plan for the future. |
| 3 | Sharing information about technical and functional trends that have been confirmed and are important to consider for future information systems development. This information is shared through a module called "rational Anticipation." The support for this flow can be in the form of an email notification that directs people to the collaborative space where this information is shared. |
| 4 | When there is new information about the skills and knowledge needed for the installation of a new production system, this information is used to create new job descriptions for the people who will be responsible for that work |
| 5 | Upon the designation of the role and responsibility of a new system, the information about it is transferred to the people who will work on updating the system. They use this information to test how well the system works and make sure it's doing what it's supposed to do. This process is called "integration testing." It's an important step in making sure the system is working correctly. |
| 6 | Technical transition and installation of the new process, i.e., implementing a new system or process that has been developed and tested, and replacing the old system or process with it. |
| 7 | Transition of organizational directives, it involves updating the instructions that employees follow to match the new process. The company may offer training to help employees learn the new instructions. |
| 8 | If there are any problems or differences between what was planned and what is actually happening in the process, there must be a way to report them in real time. This is done by using either a document workflow system or an automated monitoring system to keep track of the issues and their solutions. |
| 9 | If there are mistakes or problems in the manual part of a process, steps must be taken to quickly fix them or find a temporary solution while working on an automated process. This is maybe done by changing the instructions for how the process should be done and possibly getting more people to help |
| 10 | Sending predictions or forecasts, and also sharing resources and technologies that have been received. |
| 11 | Process of changing from old models to new ones, putting the changes into effect, keeping track of the progress and making sure everything is formalized properly |

Table 5: Construct definition as seen in [74]

| **Construct** | **Description** |
|---|---|
| *ASF characteristics* | Characteristics of the agile framework, such as its intended usage (project, program, enterprise), integration with EA thinking, team composition, and available configuration options are among its descriptive features. |
| *Scope of ASF applications* | The scope and extent of the agile framework can be categorized into two aspects: *Reach* and *Range*. Reach pertains to the extent of the framework's application within the company, ranging from a single department to the entire enterprise. On the other hand, range refers to the breadth of functionality enabled by the framework, such as in the case of a bank, from mortgage services to all services provided. |
| *Attention paid to EA* | The organization's level of consideration towards Enterprise Architecture practices, including processes, methods, and tools, with an intention to take action. One could also argue whether an organization has specific roles dedicated to EA |
| *Waterfall retainment* | The level of stage-gate methods and practices implemented after adopting the agile framework in the organization. This could include the use of sequential processes, distinct roles in subsequent development phases, detailed front-end plans, extensive documentation, and guidelines and milestones to be followed throughout the development projects. |
| *Type of agile governance* | The decision-making authority and accountability structure to promote favorable actions when implementing and utilizing Agile and EA within an organization |
| *Process/Service innovation* | Service Innovation can be defined as a new service or a renewal of an existing service, which provides benefits to the organization that developed it. Similarly, process innovation is defined as a new or significantly enhanced method of production or delivery |

# Appendix B

# Figures

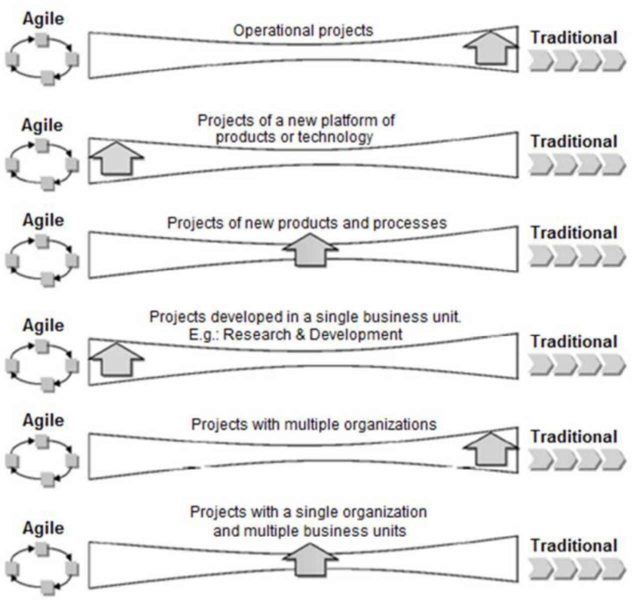

Agile
Operational projects
Traditional
Agile
Projects of a new platform of products or technology
Traditional
Agile
Projects of new products and processes
Traditional
Agile
Projects developed in a single business unit. E.g.: Research & Development
Traditional
Agile
Projects with multiple organizations
Traditional
Agile
Projects with a single organization and multiple business units
Traditional

FIGURE 1: Application of agile vs traditional methods in various project types by [17]

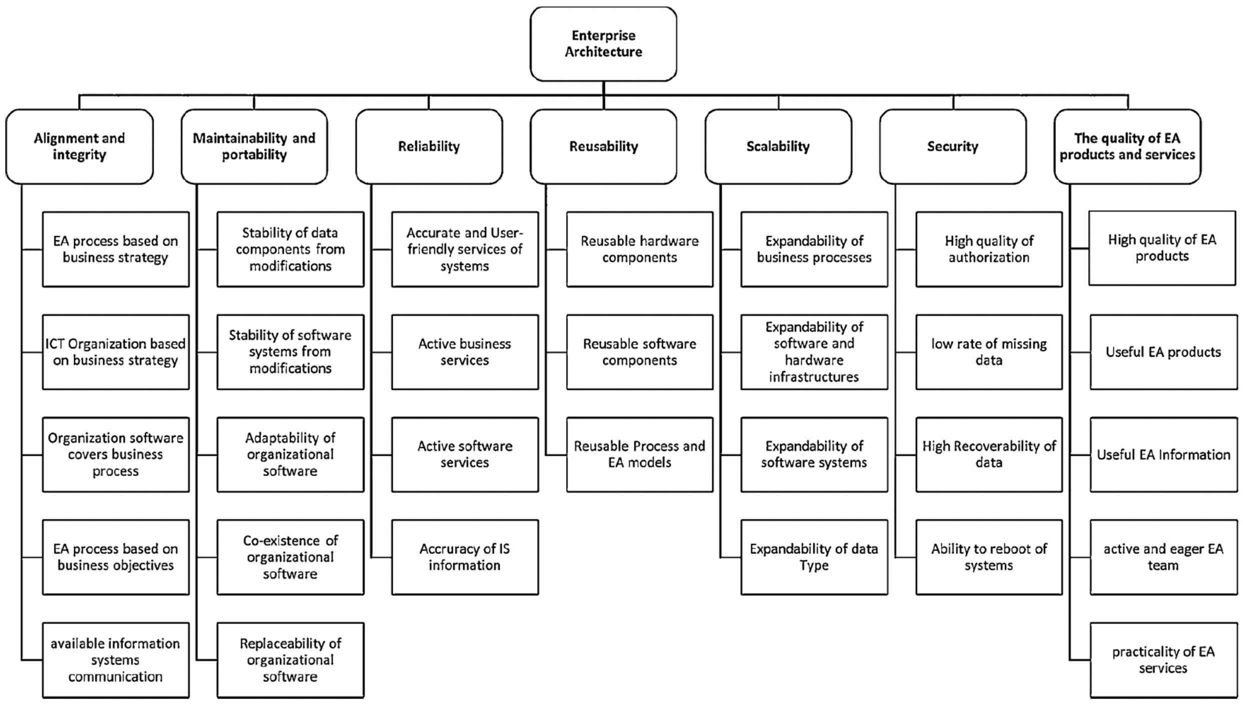

Enterprise Architecture
Alignment and integrity
EA process based on business strategy
ICT Organization based on business strategy
Organization software covers business process
EA process based on business objectives
available information systems communication
Maintainability and portability
Stability of data components from modifications
Stability of software systems from modifications
Adaptability of organizational software
Co-existence of organizational software
Replaceability of organizational software
Reliability
Accurate and User-friendly services of systems
Active business services
Active software services
Accrurracy of IS information
Reusability
Reusable hardware components
Reusable software components
Reusable Process and EA models
Scalability
Expandability of business processes
Expandability of software and hardware infrastructures
Expandability of software systems
Expandability of data Type
Security
High quality of authorization
low rate of missing data
High Recoverability of data
Ability to reboot of systems
The quality of EA products and services
High quality of EA products
Useful EA products
Useful EA Information
active and eager EA team
practicality of EA services

FIGURE 2: EA evaluation model by [51]

# Appendix C

# Interview Transcripts

This section requires permission to view from the author.