Software Construction

RWTH AACHEN UNIVERSITY

The present work was submitted to
the Research Group
Software Construction

of the Faculty of Mathematics,
Computer Science, and
Natural Sciences

Bachelor Thesis

Rule-Based Assessment of Solution Architecture Alignment

Rule-Based Assessment of Solution Architecture Alignment

presented by

Jonah Schüller

Aachen, January 5, 2023

Examiner

Prof. Dr. rer. nat. Horst Lichter

Prof. Dr. rer. nat. Bernhard Rumpe

Supervisor

Peter Alexander

Acknowledgment

I would like to thank my supervisor Peter Alexander for his support and guidance throughout the thesis. I thank him for providing & teaching me the necessary knowledge to complete this thesis and introducing me to the field of enterprise architecture. I would also like to thank Prof. Dr. rer. nat. Horst Lichter for allowing me to write this thesis in his research group. Special thanks go to my family and friends for their support and encouragement throughout my entire studies.

Jonah Schüller

Abstract

Nowadays, enterprise architectures consist of numerous systems of various applications. With the increasing complexities of architectures, demand for new requirements and constraints on enterprise architecture is growing. In order to ensure that the solution architecture of each application complies with changing requirements and constraints of the enterprise architecture, a solution architecture must be continuously assessed. These assessments should enable early identification of potential deviations from the enterprise architecture and thereby provide a basis for actions or decisions. Due to the complexity of the enterprise architecture, the assessment of the solution architecture is a challenging task. Personal experience and knowledge influence the results and the quality of the assessment. In this thesis, we propose a rule-based approach to assess the solution architecture. The rules aim to improve the reproducibility of the assessment by making the assessment process less dependent on personal experience and knowledge.

Contents

1 Introduction **1**
- 1.1 Problem Statement . . . 1
- 1.2 Research Goal . . . 2
- 1.3 Research Questions . . . 2

2 Foundation **5**
- 2.1 Enterprise Architecture . . . 5
- 2.2 Enterprise Architecture Debt . . . 6
- 2.3 Enterprise Architecture Debt Management . . . 8
- 2.4 Compliance Framework . . . 9

3 Related Work **11**
- 3.1 Automated Compliance Checking . . . 11
- 3.2 Enterprise Architecture Compliance . . . 12

4 Solution Concepts **17**
- 4.1 Scope of the Framework . . . 17
- 4.2 Compliance Model . . . 17
- 4.3 Compliance Process . . . 22

5 Realization **29**
- 5.1 Architecture . . . 29
- 5.2 Implementation . . . 32

6 Evaluation **45**
- 6.1 Setup . . . 45
- 6.2 Evaluation Results . . . 48

7 Discussion **51**
- 7.1 Discussion of the Results . . . 51
- 7.2 Implication . . . 53
- 7.3 Threats to Validity . . . 54

8 Conclusion **55**
- 8.1 Summary . . . 55
- 8.2 Future Work . . . 55

Bibliography **57**

List of Tables

4.1 Example of an assessment rule. 21

6.1 Evaluation Results 49

List of Figures

2.1 EADM framework overview [Ale+20] 8

4.1 Compliance Framework Model 18
4.2 Project Compliance Lifecycle 23
4.3 Project Compliance Assessment Process 26

5.1 Prototype architecture 30
5.2 Standard Overview 33
5.3 Standard creation 34
5.4 Standard filter 34
5.5 Rule Overview 35
5.6 Rule creation 36
5.7 Rule filter 37
5.8 Workspace 37
5.9 Project contract overview 38
5.10 Project budget 39
5.11 Project assessment overview 40
5.12 Project assessment overview 40
5.13 Compliance assessment component 40
5.14 Compliance check component 41
5.15 Analytics component 42

1 Introduction

Contents

1.1 Problem Statement 1
1.2 Research Goal 2
1.3 Research Questions 2

1.1 Problem Statement

One of the main functions of enterprise architecture (EA) is to provide a high-level comprehensive descriptive overview of the organization. These insights into the architecture of the organization are used to make management decisions. As an example, they can be used to initiate a project to improve the architecture of the organization, by optimizing structure and processes.

Furthermore, the insights into the architecture of the organization can drive the focus on suboptimal parts of the current *as-is* architecture. Such insights can provide a basis for the design of the *to-be* architecture and can be used to make decisions about future projects and investments of the organization [Joh+04] [Foo+12].

A second function of EA is to provide a prescriptive framework that guides the development of the organization's IT systems. This framework keeps the focus on the *to-be* architecture of the enterprise. Local project initiatives (that implement the EA artifacts) might not be aware of the goals of the enterprise architecture and therefore might not comply with or contribute to them. This second aspect of EA will be the focus of this thesis.

A common technique to ensure the consistency of the strategy is to use a set of prescriptions, which are used as high-level constraints for the organization. These prescriptions can be standards or regulations, which are enforced by the organization. Usually, local projects are the ones that are affected by these prescriptions, as they are the ones that implement the EA artifacts. Examples of how prescriptions can be used to guide the development of the organization are shown in [Bru+10]. The first example is a national statistics agency that employs over 200 people. The agency produces and publishes statistical data on the country. The agency uses prescriptive EA to reduce the cost of the statistical processes and to redesign the products to be more agile. Another example is a manufacturing company that uses prescriptive EA to ensure consistency between different domains of the company.

According to [BY06a], EA standards, which is one type of prescription, can help organizations manage their IT resources more effectively. Standardization includes the

definition of a set of policies, rules and guidelines that are used to ensure the quality of the architecture. They help coordinate the development process across local projects. Even if standardization does not always lead to optimal local solutions, it helps to ensure that the local solutions are aligned with the goals and vision of the enterprise.

Using prescriptive EA in an organization alone is not sufficient to ensure that the organization's IT systems comply with the EA. The compliance of the IT systems with the EA must be checked. [BY06a] states that conformance to EA standards does not occur automatically. This is especially true in larger multi-unit organizations, where these standards have to be communicated across the organization. This rises the need for a compliance assessment of the solution architecture with the enterprise architecture.

Compliance assessments are tests, performed to ensure that the solution architecture complies with all required prescriptions. It is important to note that the compliance assessments are performed at the level where the prescriptions are enforced, which is usually the local project. There have been frameworks proposed to perform compliance assessments, such as the framework by Foorthuis et al. However, as the authors state, compliance assessments involve a lot of personal influence based on the experience and knowledge of the expert [Foo+12]. This causes the compliance assessments to be inconsistent and non-reproducible. This thesis will show how to address this issue by developing a rule-based approach for the assessment of solution architecture alignment.

1.2 Research Goal

The goal of this thesis is to develop a rule-based framework to assess the compliance of solution architectures to EA prescriptions. The framework includes the management of both the rules and prescriptions and the assessment process. The process will primarily focus on the *Identification and Collection* phase of the Enterprise Architecture Debt Management Framework [Ale+20].

1.3 Research Questions

Section 1.1 has introduced the problems that appear when performing compliance assessments and the need for further research in this field. To guide the development towards the goal defined in 1.2, research questions will help to understand which parts of the problem need to be addressed. The research questions that will be addressed in this thesis are:

Which data points are relevant during the compliance assessment process of the solution architecture?

Data points are the information that is used to perform the assessment. The research question will help to understand which information is required to perform the assessment. Therefore, it helps to develop the general structure of the rule-based approach.

How can the assessment of solution architecture alignment with the enterprise architecture be supported by a tool?

As one part of the thesis is focusing on the development of a support tool, one of the research questions is to understand how a software tool can support the enterprise architect in the assessment process.

How can the assessment of solution architecture alignment with the enterprise architecture be made more consistent and reproducible?

The typical assessment process is lacking consistency and reproducibility. The focus of this thesis will be to improve this. Therefore, the third research question will help to understand which parts of the assessment process can be made more consistent and reproducible.

Which insights can be gained from the assessment of solution architecture alignment with the enterprise architecture?

Once the assessment process is improved, the next step is to analyze the results of the assessment. Part of this thesis will also be to study how reproducible assessments can be used to gain insights into the architecture of the organization.

2 Foundation

Contents

2.1 Enterprise Architecture 5
2.2 Enterprise Architecture Debt 6
2.3 Enterprise Architecture Debt Management 8
2.4 Compliance Framework 9

This chapter provides an overview of the foundational concepts of Enterprise Architecture (EA), Enterprise Architecture Debt (EAD), Enterprise Architecture Debt Management (EADM) and Compliance Frameworks.

2.1 Enterprise Architecture

In modern organizations, complex IT landscapes are used to support the business processes of the organization. These IT landscapes consist of numerous applications and systems. With increasing complexity, management of these landscapes becomes more and more challenging, posing risks to the effective execution of business processes. Therefore, the effective management of these landscapes is crucial to ensure the success of the organization. Enterprise Architecture (EA) is a management discipline that supports the management of IT landscapes. EA combines the practice of managing principles, models and methods that support both design and implementation not only of IT systems but also on the business and organizational level. The goal is to collect essential information about the organization and its IT landscape. To structure the collected information, an architecture is used. EA can be seen as a "blueprint" of the organization either of the current state or of the desired future state. This blueprint is used to support, guide and optimize the IT investments of the organization and to ensure that the IT-landscape is aligned with the business goals of the organization [Jon+06].

According to [Jon+06], one of the most important aspects of EA is that it provides a holistic view of the organization. A common issue is that domains within the organization perform their own architectural practices. These local architectural practices might be optimal within the local domain. However, these local architectures are not aligned with the architecture of the organization. This leads to a lack of alignment between the different domains and a lack of a holistic view of the organization due to the heterogeneity of the different architectural practices.

An example of local architectural practices could be the optimization of the IT architecture of a domain-specific application. The IT systems used to run this application might be optimized to the needs of the domain IT landscape. However, on a global or-

ganizational level, the IT systems might not be aligned with the overall architecture and IT landscape of the organization. For example, the IT systems might be too rigid and constrained to support potential needs, such as scalability or elasticity, of the overlying business processes.

Important to note is that EA is not only about the architectural side of the organization. The principles and guidelines, which are provided by EA, are used to develop and design the business processes and the underlying applications and systems [PG10]. The architecture does not serve as something static which restricts the development of the organization, but rather as a guideline that supports architects of different domains across the organization to drive the development of the organization in the right direction. Furthermore, the unified view of the organization can be used as a discussion basis and serve as a common ground for the different domains of the organization.

In the field of EA, there are different approaches and frameworks which can be used to structure the information about the organization. The framework of TOGAF is one of the most popular frameworks in the field of EA. The framework was developed by the Open Group and is used to structure the information about the organization. According to [Kot18] 60% of the Fortune 500 companies use TOGAF as their EA framework. There are around 75,000 certified TOGAF professionals worldwide [Gro11].

2.2 Enterprise Architecture Debt

In Software Engineering, the term *technical debt* is used to describe the shortcuts and compromises that are made during the development of a software system. Often, these shortcuts are made to meet a deadline or a certain requirement. However, these shortcuts might lead to problems in the future, like higher costs or a lack of maintainability of the software system. In general technical debt reflects the compromises that provide a short-term benefit but lead to a long-term cost [Kru+13]. The term technical debt does not only apply to software systems, it can also be applied to other domains like *database debt* [FFB19] or *documentation debt* [Rio+20].

The concept of debt being taken during a development process is also applicable to the development of an EA. The development of an EA is a complex process [ST06], where many different aspects of the organization are considered. In section 2.1, the importance of EA was discussed. It was mentioned that EA can be applied to get a holistic view of the current state of the organization but also on the desired future state of the organization.

Due to the complexity of organizations, the *as-is* situation (the current state of the enterprise) of the organization often deviates from the *to-be* situation (the desired future state) of the organization. An example of a situation where the *as-is* situation differs from the *to-be* situation is the following: The *as-is* situation of an organization is that the IT landscape is designed in a monolithic way. This monolithic design no longer holds up to the scaling requirements of the organization. Therefore, the *to-be* situation

of the organization is that the IT landscape is designed in a microservice-based way. This microservice-based design is more flexible and scalable than the monolithic design. However, the transition from the monolithic design to the microservice-based design is not trivial. The transition requires a lot of effort and time and cannot be accomplished in a one-step migration. This imaginary example shows how the *as-is* situation can differ from the ideal, future *to-be* situation of the organization. The deviation between the *as-is* situation and the *to-be* situation is referred to as *Enterprise Architecture Debt* (EAD).

[Hac+19] defines EAD as follows:

> *Enterprise Architecture Debt is a metric that depicts the deviation of the currently present state of an enterprise from a hypothetical ideal state.*

EAD arises when EA artifacts are not implemented according to their ideal design. The ideal design of an EA artifact is the design that is described in the *to-be* EA blueprint.

It is important to note that not all EAD is inherently bad, as it can be useful or even necessary to take debt in a certain situation. It is more about the tradeoff between the short-term benefit and the long-term cost. The example above is such a situation where it can be useful to take EAD during the transformation process to the *to-be* architecture. Depending on the situation, it might even be necessary to take EAD in some cases. For example, if applications of the architecture have interdependencies (e.g. in their communication interfaces) and one of these applications cannot immediately be migrated to the new architecture and the other application has to be implemented not according to the ideal design in order to be able to communicate with the first application. In case no feasible alternative exists, the extra effort is justified and the EAD has to be taken. A similar situation is described in [Hac+19]. However, there are also situations where EAD is not useful and unnecessary. For example, if an application has to be migrated to a new technology stack, but the IT department decides not to migrate the application, as it requires a lot of effort. In such a situation, postponing the migration for no good reason is not useful and leads to unnecessary EAD.

It can be compared to a situation where a company is taking financial debt to invest in a new project. Hacks et al. states that EAD can be used as a tool. The debt is useful if it helps the organization to grow and achieve its goals. However, if the organization does not properly pay back the EAD, taking the debt does not pay off [Hac+19].

A very important aspect of EAD is that it needs to be monitored and managed. Losing track of the present EAD might introduce new EAD in the future. Imagine a situation where a system is not implemented according to the ideal design and therefore EAD is taken. If the EAD is not properly monitored and managed and other systems are implemented on top or depending on the system, the EAD might increase over time [Hac+19]. Not keeping track of the EAD might become more costly to pay back in the future. Furthermore, losing track of the EAD might lead to stagnating or a less efficient transformation process.

2.3 Enterprise Architecture Debt Management

In the previous section, EAD was explained along with its implications. As explained, taking EAD might be beneficial in some situations, but only if the debt is taken in a controlled way. This leads to the question of how EAD can be managed in a controlled way. The management of EAD is called *Enterprise Architecture Debt Management* (EADM).

Keeping track of the EAD that is present in the organization is important since it increases awareness of suboptimal parts of the organization. Furthermore, a clear management process can help prevent the accumulation of EAD in the first place. The management of EAD can be done in different ways.

Figure 2.1 proposes a framework for EADM which divides the management into nine activities. *Identification & Collection, Assessment & Prioritization, Monitoring, Repayment & Prevention* combined with a continuous *Documentation & Communication* process.

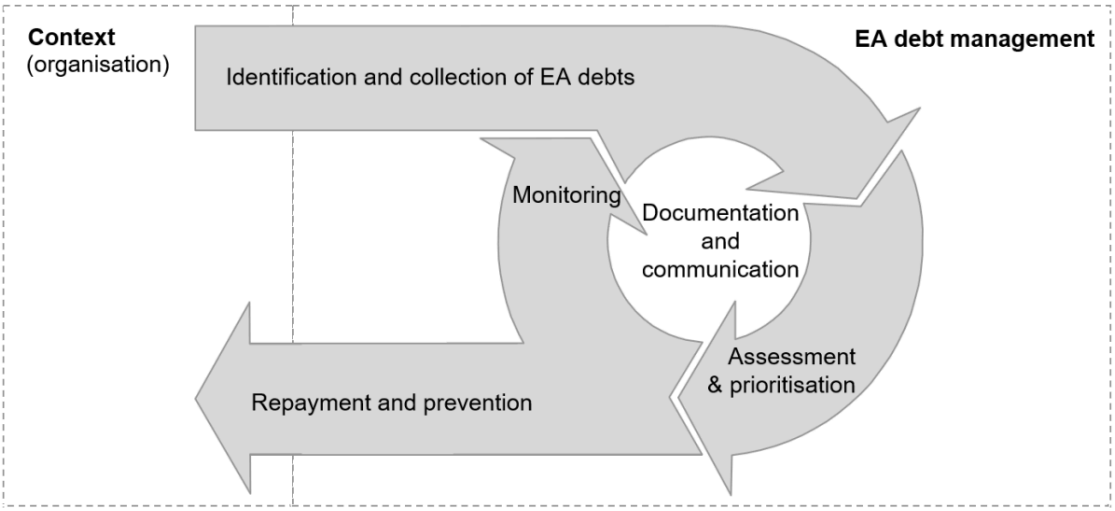

Figure 2.1: EADM framework overview [Ale+20]

Identification: The first step of EADM is the identification of EAD. The identification focuses on identifying signs of potential EAD. **Collection:** The second step of EADM is the collection activity. During that activity, further evidence of potential EAD is collected by observing the development of EA artifacts. In the third step, the identified EAD is **assessed and prioritized**. The assessments can be based on the cost/ benefit of mitigation of the EAD. The results of the assessment serve as a foundation for the prioritization of the EAD. Based on the business goals and plans of the organization the EAD is ranked and prioritized. The prioritization can help to create a mitigation strategy for the EAD.

Monitoring: The monitoring activity focuses on the continuous monitoring of the EA due to changing business requirements, etc. This is important since in rapidly changing

environments, the EAD views can become outdated if they are not constantly monitored. Outdated EAD views can lead to wrong decisions and wrong investments.

Mitigating EAD can be done in two ways: **Repayment** and **Prevention**. **Repayment:** Repayment is the process of mitigating EAD by investing to reduce or eliminate the EAD. **Prevention:** Prevention is the process of mitigating EAD by avoiding the EAD in the first place. The essence of the EADM process is the continuous **Documentation & Communication** of the EAD. The documentation and communication ensure the flow of EA knowledge across the stakeholders of the organization [Ale+20].

2.4 Compliance Framework

As stated in the previous sections, part of EA is to manage the transition from the current organization structure to the desired target structure. Usually, the transformation is carried out step-by-step in a set of projects [YNH08]. To ensure that the initiated projects contribute to the EA transformation, each project needs to be controlled and guided. This is considered as *projects need to be compliant* with the EA transformation. One issue with the compliance of projects is that the definition of compliance is not always clear. According to Ylimäki, Niemi, and Hämäläinen *compliance* mostly refers to conformance with *rules, standards, regulations, laws or contracts* [YNH08]. However, a clear definition of compliance is not always given. According to [Groa], a reason for this is that the term *compliance* or *conformance* differs between organizations.

Instead of defining the term, compliance (in the context of EA) focuses on two perspectives, as proposed by the TOGAF framework [Groa]:

1. **Architecture Compliance Reviews:** An architecture compliance review is a review of a project architecture against an *"established architectural criteria, spirit, and business objectives"*.

2. **Project Impact Assessments:** A project-specific sub-set of the EA to illustrate how the EA impacts projects.

During these architecture reviews, the EA architects look for portions of the solution architecture (SA) that are not compliant with the standards or guidelines of the organization's EA blueprint. The identified non-compliant parts of the SA can be considered as a deviation from the target architecture. Usually, such a review is divided into three phases: Preparation, Review and Discussion. Exemplary processes that follow this structure are the process proposed by [Foo+12] or [Groa].

In general, one goal of compliance assessments is to make sure that both the organization and local project initiatives contribute to the transformation strategy [YNH08]. To achieve this goal, compliance assessments should not be performed in a one-time manner. Instead, the compliance assessments should be performed continuously throughout the lifecycle of the project [Foo+12]. In the following chapter 3, the mentioned compliance frameworks are discussed in more detail.

As stated in section 2.3, the first part of the EADM process is the identification & collection of EAD. The identification & collection of EAD is done by observing the

development of EA artifacts. The development of EA artifacts can be observed using various methods. One of these methods is the use of a *compliance framework*. The non-compliant parts of the SA, identified by the compliance framework, can be considered to be EAD. Therefore, the compliance framework can be used to identify EAD.

When looking at the available literature, it can be seen that there are different approaches to compliance frameworks. However, generally, they can be grouped into two categories: *automatic compliance assessments* and *human-based compliance assessments*. The differences will be explained in the following chapter 3.

3 Related Work

Contents

3.1 Automated Compliance Checking 11
3.2 Enterprise Architecture Compliance 12

When studying Enterprise Architecture Compliance literature, it can be seen that there are two main approaches to the problem. The first approach aims to automate the compliance checking process, while the second approach aims to provide a framework to support the compliance checking process done by human experts. In this chapter, we will look into some of the research that has been done in the field of EA compliance.

3.1 Automated Compliance Checking

An automatic compliance check framework that is used to automatically check the alignment of EA artifacts and the *to-be* EA blueprint. Usually, these frameworks try to formalize EA compliance by translating it into some form of a *formal system* (e.g. [Dei+09]). As stated in section 2.3, a continuous process of EA compliance checking is necessary to ensure that the EA blueprint is aligned with the current EA. Highly complex EA blueprints can be difficult to check manually on a regular basis. Therefore, an automated compliance-checking framework can be used to check the alignment of the EA artifacts and the EA blueprint [Dei+09].

Examples of such frameworks are [FWJ20] or [Dei+09]. Both frameworks use a formal language to construct *formal rules* that are used to automatically check the compliance of the solution architecture with the enterprise architecture. The rules are then used to formulate the compliance criteria. To give an example of such a rule, using the approach proposed by Deiters et al., a rule could express that a certain component is only allowed to have a given set of dependencies. The rule would then be used to check if the component has the correct dependencies [Dei+09]. Another example is the HUSACCT tool proposed by Pruijt et al. It uses a similar rule-based approach, that enables automation when it comes to compliance checks.

To clarify, the rule-based approach developed in this thesis is not the same as the rule-based approach discussed in this section. The rule-based approach discussed in this section is used to automate the compliance checking process. The rule-based approach developed in this thesis is used to make enterprise compliance assessments more reproducible.

Generally, when looking at the literature, it can be seen that the focus of these approaches presented in this section is on the analysis of static software architecture arti-

facts. All have in common that they do not focus on assessing domain knowledge-specific compliance criteria. Foorthuis et al. states that EA might not be suited to automatic compliance tests [Foo+12]. Due to the natural language nature of EA prescriptions and that testing the compliance of EA artifacts often requires in-depth domain knowledge, it might not be possible to fully automate the compliance process or not be worth the effort. As an example, one could take the TOGAF example architecture principles [Groe]. The principles are written in natural language and express the high-level goals of the architecture. Foorthuis et al. states that tests that are done automatically are likely to yield irrelevant outcomes [Foo+12]. However, when it comes to compliance assessments in general, it is more about a combination of multiple approaches rather than a single approach that tackles the problem from all angles. Each approach has its own strengths and weaknesses and can be used to solve different dimensions of the problem [ČR12].

3.2 Enterprise Architecture Compliance

As stated in the previous section, automatic compliance checks lack in-depth domain knowledge and are likely to yield irrelevant outcomes when it comes to assessing enterprise architecture artifacts. Therefore, further related work will be discussed. The focus will be on work that is less related to software architecture compliance but focuses more on the broader field of Enterprise Architecture Compliance. General attempts to formalize EA compliance along with proposed frameworks that support the compliance checking process will be discussed.

Formulating the enterprise architecture compliance problem

The paper [ČR12] published by Čyras and Riedl aims to formulate the enterprise architecture compliance problem, by reflecting on different aspects of enterprise architecture compliance. The authors primarily focus on enterprise architecture compliance with the law. Apart from presenting the problem, the authors also raise complexity issues when attempting to formalize the enterprise architecture compliance problem. Examples of difficulties are the abstractness of norms, a large number of regulatory requirements, or heuristics that translate high-level concepts into low-level ones. After reflecting on different aspects of enterprise architecture, e.g. elements of EA and a high-level Enterprise Architecture Compliance Process, the authors drive towards the methodology of compliance with the law. Different resources, i.e internal arrangement of transparency, methods for legal architecture view and design methods for law-triggered changes, that can help to shape a framework are presented. The authors continue by taking existing frameworks and adding a legal perspective to them. Finally, the authors conclude that positioning the problem, including the legal perspective, into existing work is challenging due to the complexity and extent of the enterprise architecture compliance problem.

TOGAF Architecture Compliance Reviews

The TOGAF framework defines a process for reviewing the compliance of projects to the Technical Architecture [Grob]. In the compliance review definition, they define the goals of the compliance review. One of these goals is to catch errors in the project architecture and to reduce the cost and risk of the project. Other goals are to ensure that the project architecture adheres to enterprise standards or to identify required changes in enterprise standards. Again other goals are to communicate the general technical readiness and significant architectural gaps of the project to the management. In addition to the goals of the compliance review, the TOGAF framework also discusses the timing, of when the compliance review should be performed. They claim that the compliance review should be performed as soon as practical when the project is still able to correct major errors.

Enterprise Architecture Compliance Review Process

The authors of [Uni14] propose a procedure for reviewing the compliance of projects to the enterprise architecture. The procedure is intended to protect the stability of the enterprise architecture at Plymouth University. Furthermore, it controls unauthorized changes or deviations from the enterprise architecture. The procedure defines an architecture review process that involves three stakeholders: the IT Architect, the Enterprise Architect and the IT Management Team. Additionally, the authors provide different areas that will be taken into account during the review process, e.g. Documentation viewpoints (including potential missing information), Assessment areas and Potential outcomes.

Compliance Assessments of Projects Adhering to Enterprise Architecture

The authors of [Foo+12] propose a very detailed framework for assessing the compliance of projects. The framework includes a compliance model and a compliance process. As the framework proposed in the paper is very similar to what will be proposed in this thesis, the compliance model will be discussed in more detail in this section.

It is important to note that the compliance model which will be discussed in the next section is part of a larger framework. Working with EA is a multi-step process. Each step passes the result onto the next step. Foorthuis et al. divides the process into four layers. The first layer is the *Management layer*, which develops the visions and goals of the organization. Based on these visions and goals, constraints are formulated in the *EA layer*. The result is the *Enterprise architecture*. The resulting EA prescriptions are then used as input for the *Project layer*, where local solutions are created and designed. These local solutions serve as input for the *Production layer* [Foo+12].

Each layer contains its separate process and should be seen as distinct processes which are connected by their input and output. Since the framework deals with the conformance of local project solutions, the focus lies on the *Project layer*. This implies that certain information, e.g. the enterprise prescriptions, which are part of the models and processes will be expected to be given.

Compliance Model

Foorthuis et al. divides the compliance model into four main parts: *Compliance norms*, *Assessment item*, *Compliance test* and *Business*. The *Compliance norms* contain the *Prescriptions* which local projects are required to comply with. Everything starts with a *Strategy* which serves as the foundation to construct the Enterprise Architecture along with its transformation plans including the *Prescriptions*. The prescriptions can be of different types, e.g. *principles*, *models* or *policy statements*. However, in general, prescriptions are some form of fundamental norm or rule projects have to comply with. Furthermore, Foorthuis et al. differentiates between *Enterprise prescriptions* and *Project prescriptions*. Enterprise prescriptions are used to provide generic constraints and directions for the enterprise. They guide the development of local solutions across projects. Project prescriptions, however, are used to provide constraints and directions on a local project level. Along with the project prescriptions, Local Acceptance Criteria might be defined. The Local Acceptance Criteria is a local project-specific variation of the prescriptions.

These prescriptions drive the compliance tests of local project solution architectures in combination with a *Baseline*. The Baseline is a set of artifacts that are tested against the prescriptions. It can be seen as a snapshot of the current state of the solution architecture, which allows for interventions in case the proposed solution does not comply with the prescriptions.

Based on the prescriptions and the Baseline, the *Compliance test* is performed. The baseline will be checked with each relevant prescription to determine if the solution architecture is compliant. The result of the compliance test is the Compliance Check results [Foo+12]. Foorthuis et al. differentiates between four different types of compliance checks, which will be discussed later in this chapter.

Compliance Test Process

In this section, the compliance test process proposed by Foorthuis et al. will be explained. Foorthuis et al. presents a few requirements before compliance tests should be performed. One of these requirements is that the proposed solution architectures should not be tested for compliance at the end or later stages of the project. At this stage, it might be too late to make drastic changes to the solution architecture. Rather, the compliance tests should be performed during the lifecycle of the project. This allows for non-compliant solutions to be identified early on and for the solution architecture to be adapted accordingly [Foo+12].

Another requirement is that the compliance test of a project should be part of a larger compliance initiative. This means that the enterprise stimulates the projects to comply with the prescriptions right from the beginning and integrate the compliance tests into the project lifecycle.

The compliance test process is divided into seven phases: *Prepare Compliance Test*, *Review Artefacts*, *Assess EA conformance*, *Create EA Conformance Report*, *Discuss EA Conformance*, *Create EA Feedback Report* and *Distribute Reports* [Foo+12]. The

process starts with the preparation of the compliance test. After the preparation, the project artifacts are reviewed and assessed. Based on the results, a report is created and discussed with the project team and all relevant stakeholders.

Types of Compliance Checks

As mentioned in section 3.2, Foorthuis et al. differentiates between four different types of compliance checks. These types are *Correctness Check*, *Justification Check*, *Consistency Check* and *Completeness Check*.

The *Correctness Check* is used to determine if the solution architecture applies the prescriptions correctly. The check verifies that the way the solution architecture was designed does not violate the way the prescription is formulated.

The *Justification Check* is used to determine if (the lack of) compliance with the prescriptions is justified. For example, if an application solution deviates from the description of the prescription (which is determined by the correctness check), it needs to be justified why this deviation is acceptable. If the prescription is not applied, it needs to be justified why the prescription is not applied in this situation. Lastly, if the prescription is applied correctly, it needs to be justified whether it is indeed justified to apply the prescription.

In the concept of Foorthuis et al., prescriptions can have prescriptions that are related to each other. The *Consistency Check* is used to determine if the relations between the prescriptions are consistent. In other words, if one prescription is applied, the Consistency Check determines if the other related prescriptions are also applied.

Finally, the *Completeness Check* is used to check if all prescriptions are applied. This may only focus on the prescriptions that are relevant to the solution architecture.

All of these checks are categorized into three outcomes: *Passes*, *Failed* and *Needs attention*. The *Passes* outcome indicates that the prescription is applied correctly. The *Failed* outcome indicates that the prescription is not applied correctly. The *Needs attention* outcome indicates that the prescription might become compliant. However, it might only be partially compliant or not sufficiently implemented.

4 Solution Concepts

Contents

4.1 Scope of the Framework 17
4.2 Compliance Model 17
4.2.1 Compliance Norms 18
4.2.2 Local Projects 19
4.2.3 Assessment Rules 20
4.2.4 Compliance Assessment 21
4.3 Compliance Process 22
4.3.1 Prerequisites 22
4.3.2 Project Compliance Lifecycle 24
4.3.3 Project Compliance Assessment Process 25

In this chapter, the solution concepts of the *Compliance Framework* for the thesis are introduced. As mentioned previously, the goal of the thesis is to develop a compliance framework to align project solution architectures with enterprise architecture. The framework should support enterprise architects in the process of aligning the solution architecture. This chapter will prepare the conceptual foundation of the framework. The following chapter 5 will discuss the prototype implementation of the framework.

4.1 Scope of the Framework

Before the framework can be explained in detail, it is important to understand the scope of the framework, defining its boundaries and the requirements that are expected to be fulfilled by the enterprise. Part of working with EA is that prescriptions are defined and used to guide the development of solution architectures. As the management of the EA is not part of the scope of the framework, the framework will expect these prescriptions to be present. Furthermore, the project management and planning process is also not part of the framework. However, as the goal is to align the solution architecture with the enterprise architecture, these prescriptions and projects are essential parts of the compliance process and will be mentioned throughout this chapter.

4.2 Compliance Model

The compliance model (CM) is the core of the compliance framework. It describes the core elements of the framework and how they are related to each other. The model is shown in figure 4.1. Generally, the CM can be divided into four core parts:

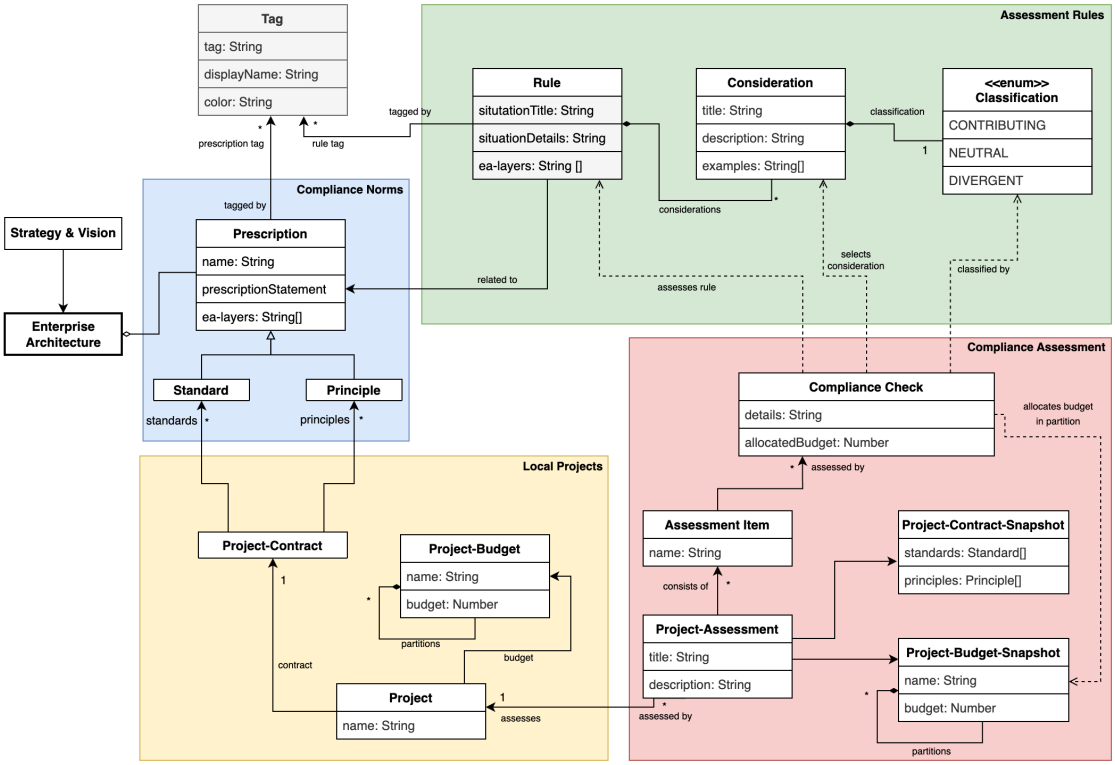

Figure 4.1: Compliance Framework Model

- **Compliance Norms**: The compliance norms are the foundation of the compliance tests. They define the constraints and guidelines in form of EA-Prescriptions, which should be followed when developing a solution architecture.
- **Local Projects**: Local projects are the project initiatives within the enterprise that are used to carry out transformation initiatives.
- **Assessment Rules**: The assessment rules are used to formalize commonly occurring situations into *rules* to perform the compliance tests in a reproducible way.
- **Compliance Assessment**: The compliance assessment part uses the assessment rules to assess local project initiatives against the compliance norms.

4.2.1 Compliance Norms

Starting with the Strategy & Vision of the enterprise, which serves as the foundation of the enterprise architecture. Within the enterprise architecture, the EA-Prescriptions are defined in form of **Standards** and **Principles**. The EA-Prescriptions are the guidelines and constraints that should be followed when developing solution architectures. As Boh and Yellin states in [BY06b], well-defined EA-Standards guide the enterprise to choose

technologies that help the enterprise to move towards the goals of the business strategy. They help achieve greater compatibility between the IT components and the integration of applications. Additionally, Principles are defined to provide rules and guidelines that inform and support the development of the organization. They provide a basis for decision making [Groe].

Both principles and standards are defined in the same way. The *name* of the prescription serves as the identifier and should be unique. The most important component of the prescription is the *prescription statement*. The prescription statement consists of multiple parts that explain and justify the prescription. It can be documented in a variety of ways, e.g. in form of plain text, a diagram, pictures, etc. [BY06b]. Regardless of the format, the prescription statement should include information about a formal definition, the rationale, and potential implications. The formal definition is a formal description of the prescription. It should be as precise as possible to avoid ambiguity. The rationale is the reason why the prescription is defined. It should explain the reason why the prescription is defined and what it is supposed to achieve. The potential implications are the consequences of the prescription. They should explain the potential consequences when implementing the prescription [Foo+12]. Boh and Yellin states that standards can be related to a certain EA-Layer. This is the case for both standards and principles. The EA-Layer is the layer in the enterprise architecture that the prescription is related to. Supported layers are the *Business Layer*, *Application Layer*, *Data Layer*, and *Infrastructure Layer*, *Strategy Layer* and *Implementation & Migration Layer*. The prescription can be related to multiple layers. The EA-Layers refer to the *TOGAF Architecture Domains* [Grod]. It should be noted that the domains are not a fixed definition. Enterprises can consider other domains as well [Groc]. Furthermore, prescriptions are tagged using *Tags*. They are used to attach metadata to the prescription about the content, context or purpose of the prescription. This helps to group and filter prescriptions. The tags are defined by the user and can be freely chosen. Each tag has a *name* and a *display name*. The name is used to identify the tag and should be unique. The name is inferred from the display name by removing all non-alphanumeric characters and converting the display name to lowercase. This is done to ensure that different spellings result in the same tag name (e.g. *MongoDB* and *Mongo-Db* result in the same tag name *mongodb*).

4.2.2 Local Projects

Another important component of the CM is the **Local Project**. Local projects are the projects within the enterprise that are used to carry out transformation initiatives. This means that local projects are the initiatives that need to comply with the EA-Prescriptions. To manage the scope of the local project, the local projects have a **Project-Contract**. A project contract is a contract between the local project and the enterprise. It contains the agreed-upon principles and standards that should be followed or carried out by the local project. The principles and standards of the contract are a subset of the EA-Prescriptions. The project contract will come into play when assessing the local project against the EA-Prescriptions. Another part of the local project is the **Project Budget**. The project budget is the budget which is allocated by the enterprise

for the local project. The budget follows the composition pattern and recursively can consists of multiple **Budget Partitions**. These budget partitions will be used in the compliance assessment. Assessed parts of the solution architecture can be allocated to a budget partition. This allocation allows us to track the progress of the local project and to see how much of the budget has been used. Furthermore, the budget allocation can visualize how the investment is contributing to the transformation of the enterprise.

4.2.3 Assessment Rules

As Foorthuis et al. states in the discussion of their research results in [Foo+12], compliance assessments are a highly subjective process. EA prescriptions are often written in a highly abstract language and leave a lot of room for interpretation. Projecting the EA prescriptions onto the solution architecture to check for compliance opens up the possibility for different interpretations. Furthermore, the personal experience and preferences of the assessor can influence the assessment. To address these issues, the CM uses **Assessment Rules**. Assessment rules provide an additional layer of abstraction to formalize commonly occurring situations into *rules* to perform the compliance tests in a reproducible way. This builds upon the work of Foorthuis et al. who proposed that prescriptions should be operationalized in order to make them less prone to interpretation. Each rule consists of a **Situation** and related **Considerations**. The situation refers to a specific situation that might arise when assessing a local project against the EA-Prescriptions, e.g. the situation *The local project uses a technology that is not supported by the enterprise* might arise when assessing a local project. Along with the situation, the rule also contains considerations. The considerations are the factors that should be considered when assessing the situation. The considerations are used to classify the situation present in the solution architecture into different *compliance classifications*. Similar to [Foo+12], which uses the terms *Passes*, *Fails*, and *Needs attention*, the CM uses the terms *Contributing*, *Divergent*, and *Neutral*. The *Contributing* outcome means that the proposed solution architecture contributes to the transformation of the enterprise with respect to the situation described in the rule. The *Divergent* outcome means that the proposed solution architecture diverges from the transformation of the enterprise with respect to the situation described in the rule. This is similar to the *Passes* and *Fails* outcomes of [Foo+12] respectively. However, one difference to [Foo+12] is that the *Neutral* outcome is not simply used to indicate partial compliance. The *Neutral* outcome is used to indicate in-between cases where the situation is neither fully contributing nor fully divergent. These cases cover situations where the solution architecture violates prescriptions for a reason that is due to outside factors of the local project. For example, the situation *The local project uses a technology that is not supported by the enterprise* might be neutral when the standard that prohibits the use of the technology was not defined at the time the local project was started. An example of a rule is shown in table 4.1. The rule describes the situation that the local project uses a technology that is not supported by the enterprise.

Similar to the EA-Prescriptions, the assessment rules allow metadata attached to them. Similar to the EA-Prescriptions, the metadata consists of *tags* and *ea-layers*. The

Situation	Consideration	Classification
The local project uses a technology that is not supported by the enterprise.	The standard that prohibits the technology did not exist at the time the project was started.	Neutral
	The project uses the technology without a valid reason.	Divergent

Table 4.1: Example of an assessment rule.

functionality of the tags and ea-layers is the same as for the EA-Prescriptions. Attaching metadata to the rules helps to deal with scalability issues of the CM when dealing with a large number of rules. As Foorthuis et al. states, a large number of such rules probably leads to testers not being able to keep track of them or not reading them at all. The metadata can be used to filter the rules that are relevant to a specific local project. This gives testers the ability to scope the rules that are relevant for the currently assessed item of the solution architecture, e.g. by filtering for the corresponding ea-layer or tag.

Rules can also have references to the principles and standards of the enterprise. These references can provide further contextual information about the rule. During the compliance assessment, the references can be used to further limit the set of rules that are relevant to the local project by filtering out rules that do not have a reference to a principle or standard of the project contract. The combination of the metadata and the references enables a rich filtering mechanism to search for a relevant set of rules during the compliance assessment.

4.2.4 Compliance Assessment

The final part of the CM is the **Compliance Assessment**. The compliance assessment is the part of the model that is used to assess the local project against the EA-Prescriptions. One key requirement of the compliance model proposed by Foorthuis et al. is that the compliance assessment should not be carried out at later stages of project initiatives but rather at multiple stages of the project lifecycle [Foo+12]. Therefore, the CM provides the functionality to assess projects multiple times during the project lifecycle. This is accomplished by the **Project Assessment**. An assessment instance is a snapshot of the solution architecture at a specific point in time. The assessment instance is used to assess the solution architecture against the EA-Prescriptions and the assessment rules. As mentioned before, one project can have multiple assessment instances. To keep track of the progress of the local projects, assessments need to freeze the project contract and the project budget at the time of the assessment. This is done by creating a **Project Contract Snapshot** and a **Project Budget Snapshot** for each assessment instance. The project contract snapshot contains all the agreed-upon principles and standards of the project contract at the time of the assessment. The project budget

snapshot contains the budget tree of the project at the time of the assessment. These information are snapshot to ensure that the assessment is not influenced by changes to the project contract or the budget during the project lifecycle. For example, if the project contract is extended during the project lifecycle by adding new principles and standards, the assessments that were carried out before the extension should not be influenced by the new principles and standards. These changes should only be considered in assessments that are carried out after the extension of the project contract.

Each assessment instance contains a set of **Assessment Items**. Every assessment item represents the assessment of a specific item or part of the solution architecture. The assessment items are used to group the assessment into smaller parts. For example, in case the project spans across multiple departments or applications the assessment items can be used to group the assessment by department or application. Assessment items internally contain the compliance checks, in other words, the compliance assessment of the assessment item. A rule is assessed by selecting a consideration of the rule, describing the reason for the selection, and allocating some budget for this assessment. Similar to the contract and budget snapshots, rule assessments contain a snapshot of the rule at the time of the assessment. This is done to ensure that the assessment is not influenced by changes to the rule at later stages.

4.3 Compliance Process

In the previous section 4.2 the compliance model with all its components was presented. This section describes how the components presented are used in the context of the compliance process and project lifecycle. This is done by presenting two process models, first the *project compliance lifecycle model* and second the *project compliance assessment process model*. The project compliance lifecycle covers the entire project lifecycle and describes the context of the compliance assessments. The project compliance assessment process model describes the process of carrying out a compliance assessment. It is important to note that the project compliance assessment process is not a separate process, but rather part of the project compliance lifecycle.

4.3.1 Prerequisites

Before explaining the process models, a few requirements & prerequisites are presented.

The first prerequisite is that it is recommended to carry out compliance assessments at multiple stages of the project lifecycle [Foo+12]. This means that the compliance assessment is not only carried out once at the beginning or end of the project but rather as a repetitive part of the project lifecycle. It should be noted that this practice differs between organizations. External factors outside the compliance scope such as the project management methodology and the project size can influence when project assessments are carried out. Therefore, the compliance model proposed in this paper is designed to support multiple assessments during the project lifecycle. However, it is not mandatory to carry out multiple assessments. The compliance model can also be used to carry out

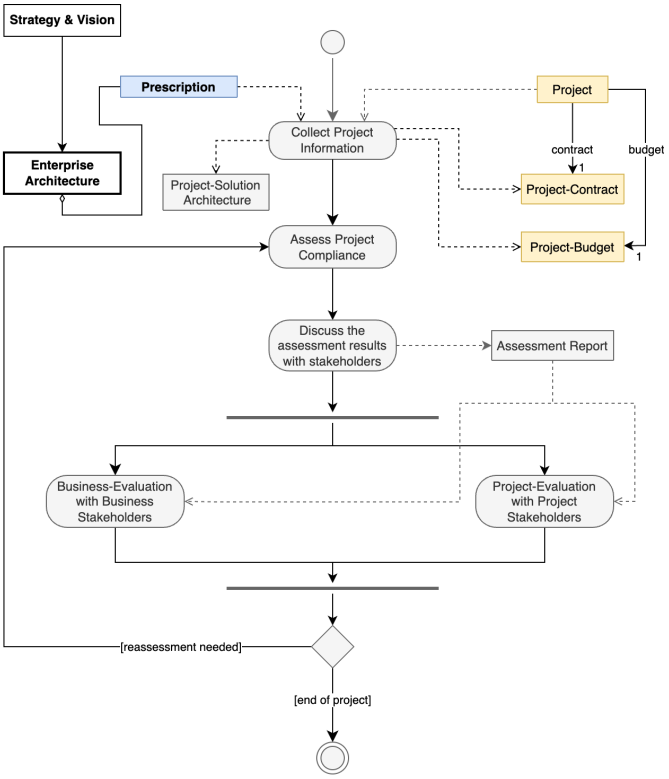

Figure 4.2: Project Compliance Lifecycle

a single assessment.

In section 4.1 it was mentioned that portions of the information that is needed to carry out the compliance assessment are expected to be given by the enterprise. Expecting the enterprise to provide the information is a second prerequisite. The following information is expected to be given by the enterprise: *EA-Prescriptions*, *active projects*, *project contract* and *project budget*. However, only a part of this information is mandatory to carry out the compliance assessment. The EA-Prescriptions and the active projects are mandatory to carry out the compliance assessment. Information that is not mandatory means that the compliance model can still be used to perform the compliance assessment, but it might hurt the quality of the assessment. The project contract and the project budget are not mandatory to carry out the compliance assessment. The project contract improves the effectiveness of the compliance assessment by providing contextual information about the project. The project budget improves the informational value of the compliance results by providing information about the budget that is allocated to the project.

4.3.2 Project Compliance Lifecycle

Figure 4.2 shows the project compliance lifecycle. The model can be divided into three phases, namely the *Collection*, *Assessment*, and the *Communication* phase.

1. **Collect Project Information**: The *Collection* phase is the first step of the project lifecycle. In this phase, the enterprise architect collects information about the local project. This collection can be separated into three parts. One part is the collection of the financial information of the project. This information might come from the project stakeholders or other departments of the enterprise. Based on the financial information the enterprise architect can create the project budget. The second part of the collection is the collection of the solution architecture of the project. The project stakeholders are expected to provide the solution architecture. The solution architecture is the proposed solution of the local project. It contains various aspects of the project, such as goals and constraints, strategy, views and risks. If the solution architecture is detailed enough, it can be used to create the project contract, which is the third part of the collection. Otherwise, the enterprise architect can create the project contract together with the project stakeholders. It is important to note that the collection of these three parts does not have to be done in a specific order. It depends on other processes of the enterprise how the information is collected or can be gathered.

2. **Assess Project Compliance**: Once all the information is collected, the first draft of the solution architecture of the project can be assessed against the EA-Prescriptions. This is done by the *Assess Project Compliance* phase. The assessment is carried out by the enterprise architect. The *Assess Project Compliance* is more detailed described in section 4.3.3.

3. **Discuss the assessment results with stakeholders**: The next phase is the *Discuss the assessment results with stakeholders* phase. As multiple stakeholders of different domains might be involved in the assessment process, the assessment results should be discussed with the stakeholders. The result of the discussion will be an *Assessment Report*. The assessment report contains the results of the assessment along with the justifications and recommendations. The assessment report is used to initiate the *Communication* phase. The results of the assessments are communicated both with the project stakeholders and the business stakeholders. Therefore the process is split into a parallel process where the results are communicated with both of the mentioned parties.

4. **Business-Evaluation with Business Stakeholders**: The first part of the *Communication* phase is the *Business-Evaluation with Business Stakeholders* phase. The assessment report is discussed with the business stakeholders to evaluate the business impact of the identified non-compliance issues.

5. **Project-Evaluation with Project Stakeholders**: The second part of the *Communication* phase is the *EA-Debt-Evaluation with Project Stakeholders* phase. The

assessment report is discussed with the project stakeholders to evaluate the project impact of the identified non-compliance issues. The enterprise architect communicates the discovered non-compliance issues along with potential recommendations to the project stakeholders.

As mentioned before, the compliance assessment is a continuous process. This means that after the results of the assessment are communicated to the stakeholders, the assessment process is repeated as soon as the project needs to be assessed again. This can be due to changes in the project contract, the project budget, the project solution architecture, or simply on a timed basis. The project lifecycle is repeated until the project is finished.

4.3.3 Project Compliance Assessment Process

Figure 4.3 shows the compliance process. As mentioned in the previous section, the compliance process is a part of the project lifecycle. In detail, the compliance process is carried out in the *Assess Project Compliance* phase of the project lifecycle. The information collected by the enterprise architect in the *Collection* phase is used as input for the compliance process. In section 3.2, the concept of budget snapshots and contract snapshots was mentioned. Before starting the assessment of a project, the current project budget and project contract are used to create a budget snapshot and a contract snapshot. These snapshots will be used during the assessment process. The snapshots ensure that the assessment carried out will not be affected by changes in the project budget or contract.

1. **Identify Assessment Items**: The first step of the compliance process is to identify the assessment items. The assessment items will be identified based on the solution architecture of the project. Each assessment item represents a discrete part of the solution architecture that is assessed against the EA-Prescriptions.

2. **Select Prescriptions to assess**: The next step is to select the EA-Prescriptions that are relevant to the assessment items. This is done by the enterprise architect. The project contract can be used to identify the relevant EA-Prescriptions since the project contract contains the relevant prescriptions. Furthermore, the attached metadata of the EA-Prescriptions can be used to identify the relevant EA-Prescriptions.

3. **Find applicable rules**: As mentioned in the previous sections, one of the goals of the compliance model is to make the compliance assessment more reproducible. Therefore, the prescriptions are not directly assessed against the solution architecture. Instead, the assessment rules are used to assess the solution architecture. It is the responsibility of the enterprise architect to find the applicable rules for the assessment items and the selected EA-Prescriptions. If specific rules are missing in the collection, the enterprise architect can even create new rules.

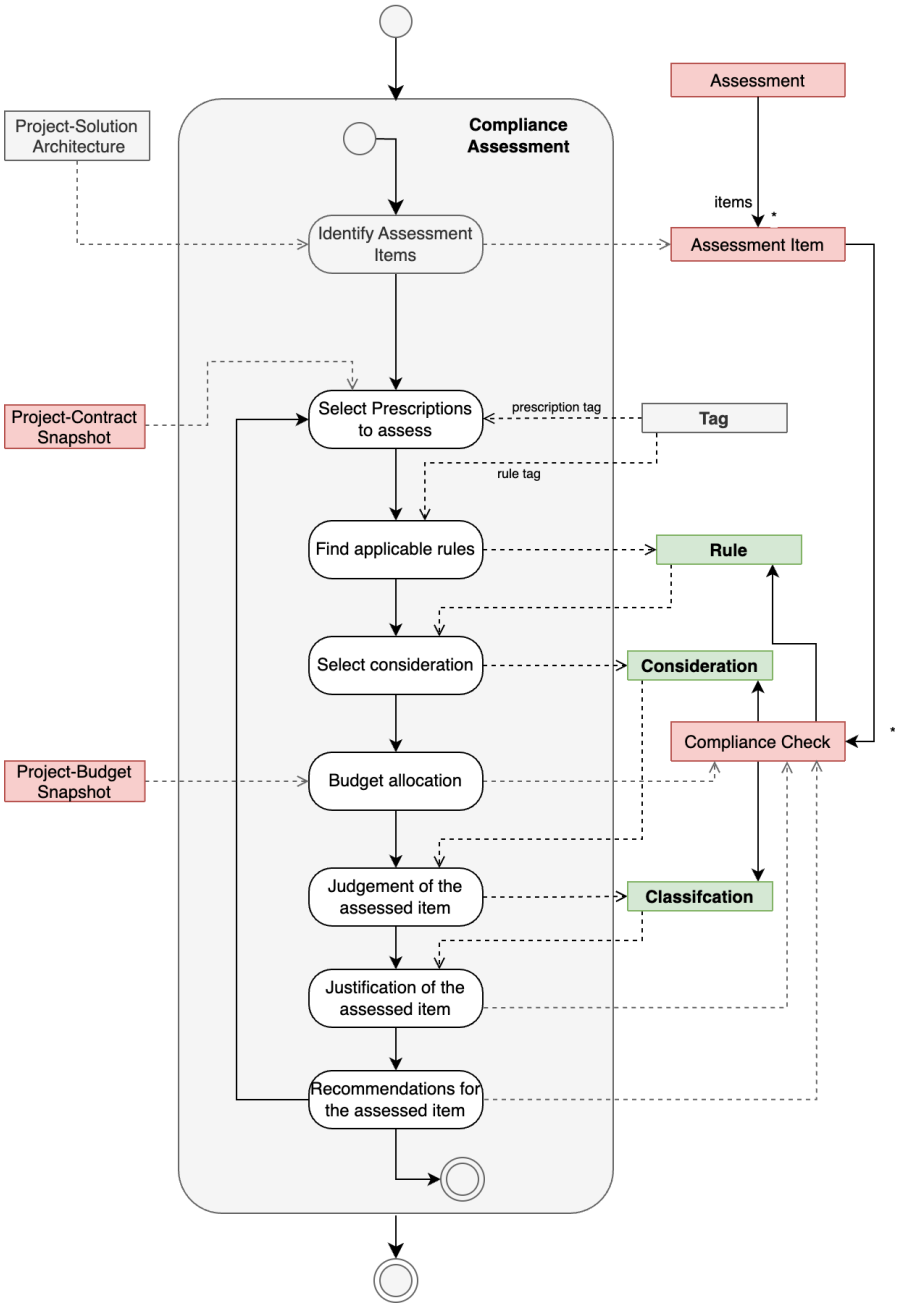

Figure 4.3: Project Compliance Assessment Process

4. **Select consideration**: The next step is to select the consideration of the assessment rules, which corresponds to the assessment item. The consideration is the part of the rule that will judge the assessment item. Similar to the previous step, the enterprise architect may have to create new considerations for the assessment rules if the existing considerations are not applicable.

5. **Budget allocation**: The enterprise architect can allocate the investment costs of the assessed item from a specific project-budget partition and attach them to the compliance check. This is where the budget snapshot is used as a reference for the current project budget.

6. **Judgment of the assessed item**: Additionally to the selected rule and consideration, the enterprise architect judges the compliance of the assessment item. The judgment is based on the consideration (which provides a classification). The judgment includes whether the assessment item is compliant or non-compliant.

7. **Justification of the assessed item**: Besides the judgment, the enterprise architect should also justify the judgment. The justification is used to explain the judgment and to provide additional information about the judgment.

8. **Recommendation for the assessed item**: Finally, after the judgment and justification of the assessment item, the enterprise architect can provide a recommendation. Recommendations can be used to provide additional feedback for the project stakeholders, e.g. how to improve the compliance of the assessment item or how to mitigate the non-compliance of the assessment item.

Based on steps 3 to 7, a *Compliance Check* is created for each assessment item. The compliance check contains the information, in detail:

- A reference to the used rule
- The selected consideration
- Potentially, the allocated budget
- The judgment of the assessment item
- The justification of the judgment
- The recommendation for the assessment item

Once all the steps are completed, the enterprise architect either starts to select the next prescription which should be assessed or starts the assessment of the next assessment item. This process is repeated until all the assessment items are fully assessed. After all the assessment items are assessed, the enterprise architect can start the *Discuss the assessment results with stakeholders* phase.

5 Realization

Contents

5.1 Architecture 29
5.1.1 Architectural foundation 29
5.1.2 Technical decisions 31
5.1.3 Data model 32
5.2 Implementation 32
5.2.1 Prescription management 32
5.2.2 Rules management 35
5.2.3 Project & assessment management 37
5.2.4 Analytics 42

The previous chapter explained the theoretical solution concept of the compliance model. This chapter will be built upon this concept and demonstrate how it can be implemented in a prototype application. As mentioned before, the goal of this thesis is to create a software prototype that implements the compliance model. This chapter will explain how the prototype was created and how it can be used.

5.1 Architecture

The objective is to create a prototype application for the rule-based compliance assessment of software architectures. After the theoretical solution concept was defined in the previous chapter, the architectural foundation and design decisions will be explained in this chapter. The architectural foundation covers the components of the prototype and their interactions. Furthermore, the technical decisions, which are the decisions that were made to implement the prototype, will be explained.

5.1.1 Architectural foundation

As it can be seen in figure 5.1, the prototype is divided into three layers: the Presentation layer, the Application layer and the Persistence layer. The presentation and application layers communicate with each other through an *API* (Application Programming Interface). The application layer communicates with the persistence layer through an *ODM* (Object Data Modeling) library. The technical decisions will be explained in more detail in section 5.1.2.

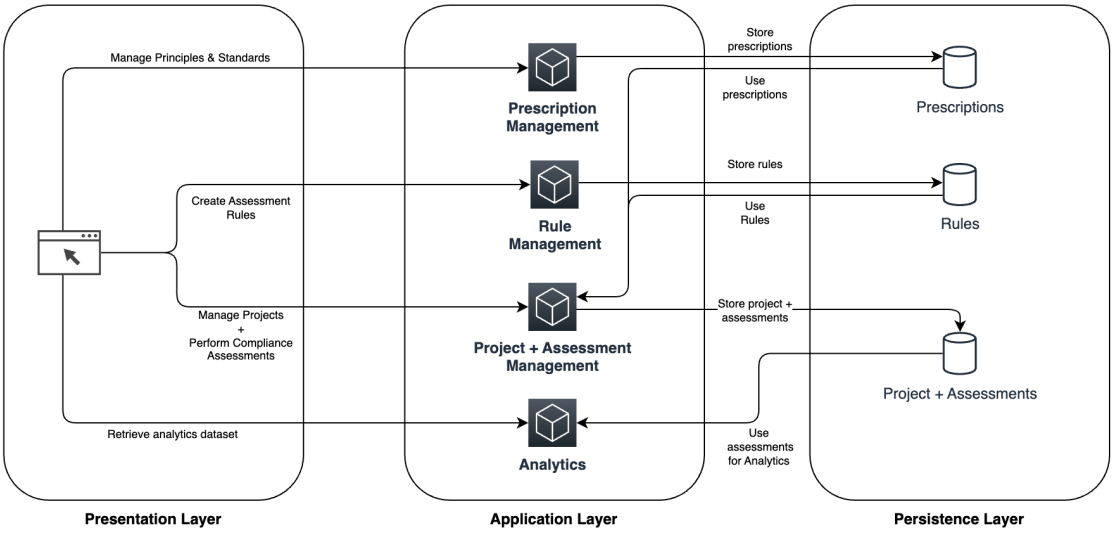

Figure 5.1: Prototype architecture

Presentation layer

The presentation layer represents the user interface (UI) of the prototype, which is implemented as a web application. The application provides a UI for the management of the principles, standards, rules, projects and assessments. Furthermore, the application provides a UI for compliance checks and analytics. The web application will use the APIs exposed by the application layer to interact with the application layer. All involved stakeholders will be able to interact with all components of the prototype through a web browser.

Application layer

The application layer is the core of the prototype, as it implements the business logic. The application layer is responsible for taking the input, that comes from the user interface in the presentation layer, and transforming it into the data model of the prototype. The data models will be forwarded to the persistence layer to store them in the database. The application layer is subdivided into four components:

- **Prescription management**: Prescription management is responsible for the management of the principles and standards. The API of the prescription management is used by the presentation layer to create, read, update and delete (CRUD operations) principles and standards. Furthermore, the API allows searching for principles and standards by using a search query.
- **Rules management**: The rules management is responsible for the management of the rules. The API of the rules management is used by the presentation layer

to perform CRUD operations on the rules. Additionally, the API allows searching for rules by using a search query.

- **Project + assessment management**: The project + assessment management is responsible for the management of the projects and their assessments. The API of the project + assessment management exposes endpoints to create, read, update and delete projects and assessments. In addition to the CRUD operations, the API also exposes endpoints to perform compliance checks. The project + assessment management uses the rules and prescriptions data to perform the compliance checks.

- **Analytics**: The analytics is responsible for generating the analytics for the compliance checks of the projects. The API of the analytics exposes read-only endpoints to retrieve the data for the analytics. However, the analytics are not generated by the API, but by the presentation layer based on the user configuration. The purpose of the analytics component is to prepare the data that is retrieved from the project + assessment management component's database and to transform the data into a format that is suitable for generating analytical charts. More information about the analytics can be found in section 5.2.4.

Persistence layer

The persistence layer is responsible for retrieving and storing the data of the prototype such as the principles, standards, rules, projects and assessments. The persistence layer is implemented using a document-based database and the communication between the application layer and the persistence layer is done through an ODM (Object Data Modeling) library.

5.1.2 Technical decisions

This chapter will explain the technical decisions that were made to implement each layer or component of the prototype. The presentation layer is implemented as a web application. The web application is implemented using the Next.js framework [Nex]. Next.js is a framework that is built on top of React.js [Rea]. React.js is a JavaScript library (with Typescript support [Typ]) for building user interfaces [Rea].

An additional feature of Next.js is the ability to implement API endpoints with server-side code. This feature is used to implement the application layer of the prototype. These API endpoints are used by the presentation layer to interact with the application layer. Each component of the application layer exposes REST API endpoints which are used by the presentation layer to interact with the application layer. The API communication will be in JSON format.

Apart from the communication between the presentation layer and the application layer, the application layer also needs to communicate with the persistence layer. The persistence layer is implemented using a MongoDB database [Mona]. MongoDB is a

document-oriented database that is used to store the data of the prototype. The advantage of using a document-oriented database is that the data can be stored in a flexible schemaless way. Relation databases require a predefined schema, which is not always the best solution for storing data in the prototype. The reason for this is, to generate the snapshots mentioned in the 4.2 chapter, the application layer needs to be able to simply duplicate entire data structures. Using a document-oriented database simplifies this process by allowing the mentioned schemaless data storage. The communication between the application layer and the persistence layer is done through an ODM (Object Data Modeling) library. The ODM library that is used is Mongoose [Monb]. Mongoose is an object modeling tool for MongoDB which helps to ensure data consistency and validation.

5.1.3 Data model

The data model of the prototype is based on the compliance model presented in section 4.2. As mentioned in the previous section, the data is stored in a document-oriented database. This allows to store data in a dynamic format. Therefore the data model shown in section 4.2 - figure 4.1 can be used as the data model of the prototype. However, there are some minor differences between the compliance model and the data model of the prototype which will be explained in this section. These differences primarily prevent previous assessments from being affected by changes to the rules.

- **Compliance Checks**: In the compliance model a compliance check has a reference to the rule, the selected consideration and a classification. In the prototype, this reference is replaced by a copy of the rule. Using a copy of the rule prevents changes to the rule from affecting the compliance check.
- **Project Contract Snapshots + Project Budget Snapshots**: It has been mentioned before, but to clarify: When snapshots are created, the entire data structure is copied. In the case of the project contract, this means that the list of principles and standards is copied. In the case of the project budget, this means that the nested project budget partitions are copied.

5.2 Implementation

The previous section discussed the architectural side of the prototype. This section will build upon the architectural foundation and explain how the different components of the prototype are implemented. Each part will be explained as a whole, covering the presentation layer by showing screenshots of the web application, the application layer and the persistence layer.

5.2.1 Prescription management

The first component of the application layer is the prescription management. The component is subdivided into two parts: the principle management and standard management.

The explanation of prescription management will be done by using the standard management as an example. The principle management will be omitted, as it follows the same structure and functionality. The prescription management is responsible for the management of the principles and standards of the enterprise. Stakeholders can create, manage and delete principles and standards.

Overview

Figure 5.2 shows the *standards* overview page of the application, which is the entry point for the standards management. The page allows the user to create new standards, view, search and manage existing standards. By clicking on a standard, the user is redirected to the standard detail page.

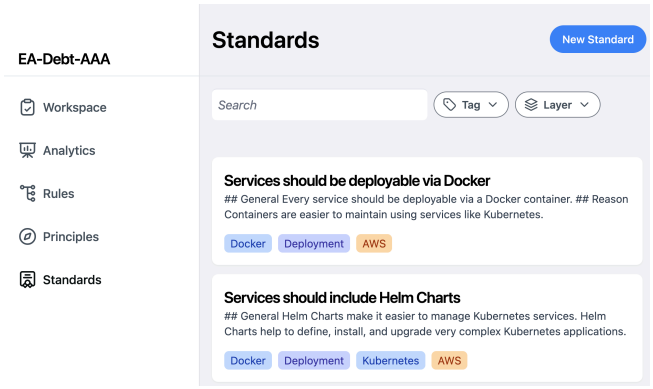

Figure 5.2: Standard Overview

Creating new prescriptions

In section 4.2 the prescription model was explained. It was mentioned that the prescription statement is composed of multiple parts, such as the formal definition, rationale, and potential implications. To give the user the ability to manage complex prescription statements, the prescription management uses markdown format. Markdown is a lightweight markup language that can be used to create formatted text, embed images, create tables and add links [Mar].

Figure 5.3 shows the creation of a standard. As seen in the figure the user can enter a title and a standard statement in markdown format. Additionally, there is a section to attach the metadata to the standard. When the user clicks on the *EA-Layer* row, a dropdown menu will appear, which enables the user to select the EA-Layer of the standard. The same applies to the *Tags* row. The text editor shown in the figure enables the user to enter the standard statement in markdown format. By clicking on the *Create Standard* button, the entered information, including title, prescription statement, EA-Layer and tags, will be transferred to the application layer via the REST API of the prescriptions management component and stored in the database.

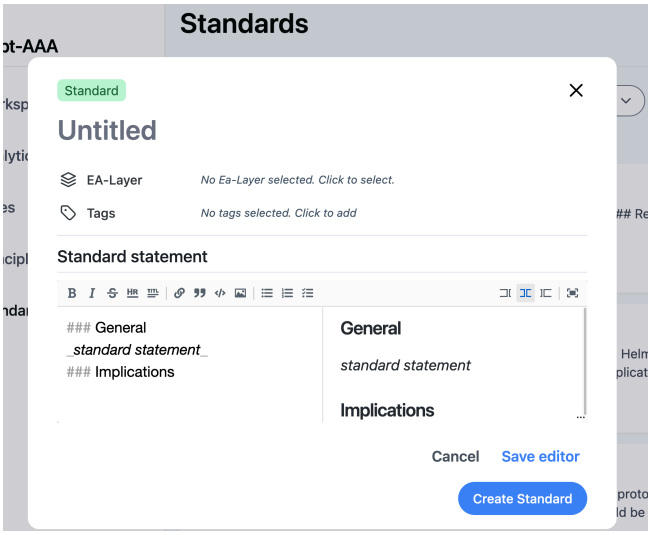

Figure 5.3: Standard creation

Filtering & searching for prescriptions

The standards overview page can be used to search and filter for standards. As mentioned in the previous chapter, the metadata of the prescriptions can be used to filter and search for prescriptions.

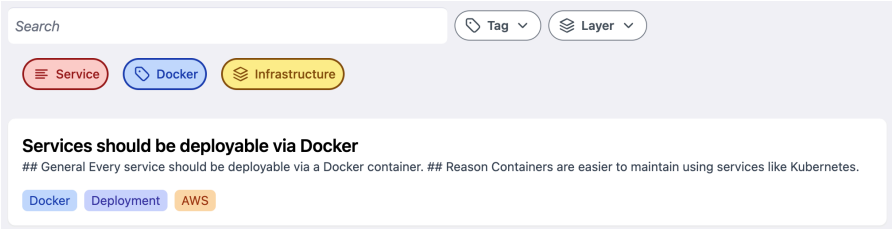

Figure 5.4: Standard filter

Figure 5.4 shows the filter of the standards. The application supports a filtering mechanism, which enables the user to build a filter query by combining different types of filters. The user can filter by the title, the EA-Layer and the tags. The example filter in the figure shows a *keyword filter* which searches for the keyword *Service* in the title of the standards, a *tag filter* which filters for the tag *Docker* and a *EA-Layer filter* which filters for the EA-Layer *Infrastructure*. Using the filtering mechanism, users can easily find the standards they are looking for, even if there is a large number of standards in the database.

Editing prescriptions

The user can edit or delete a standard by opening a standard by clicking on the standard. A detail modal will be opened, where the title, statement and metadata of the standard can be edited. The modal looks like the creation modal shown in figure 5.3, but all information is already filled with the current values of the standard. Additionally, the standard can be removed from the database. When a prescription is removed from the database, it will only be removed from the prescription management, not from the project contract snapshots. Once the user is done editing the standard, the updates will be sent to the application layer and persisted in the persistence layer.

Principle management

The principle management follows the same structure and functionality as the standard management, but instead of managing standards, it manages principles.

5.2.2 Rules management

The rules management is responsible for managing the assessment rules of the enterprise. It is similarly structured as the prescription management. Stakeholders can create, manage and delete rules.

Overview

Figure 5.5 shows the rule overview page of the application, which is the entry point for the rules management. It is similar to the standard overview page, but instead of showing the standards, it shows the rules. The user interface of the rule overview enables the user to create new rules, view, search and manage existing rules. By clicking on a rule, the user is redirected to the rule detail page.

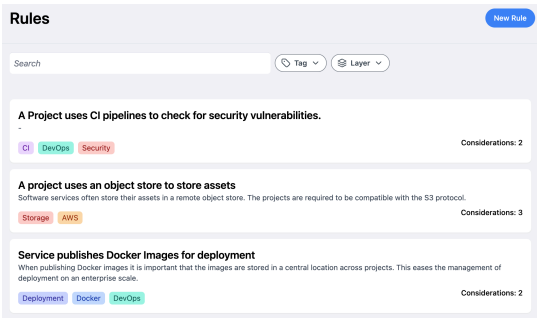

Figure 5.5: Rule Overview

Creating new rules

As explained in section 4.2, the rules consist of a situation and considerations. The modal component shown in figure 5.6 provides the user with an interface to enter the situation of the rule and the considerations. Additionally to the situation and considerations, the user can attach the EA-layer and tag metadata to the rule. This is done by clicking on the *EA-Layer* row and the *Tags* row. A dropdown menu will appear, which enables the user to select the EA-Layer and the tags of the rule. Apart from the tag and layer metadata, rules have relationships to principles and standards. Therefore, the *Rules Creation Modal* offers additional UI components to select the related principles and standards. By clicking on the *Related Principles* or the *Related Standards* row, a dropdown menu will appear, which enables the user to search & select the related principles or standards.

Once the user is done entering the information, the rule can be created by clicking on the *Create Rule* button. The entered information, including situation, considerations, EA-Layer, tags, related principles and related standards, will be transferred to the application layer via the REST API of the rules management component and persisted in the persistence layer.

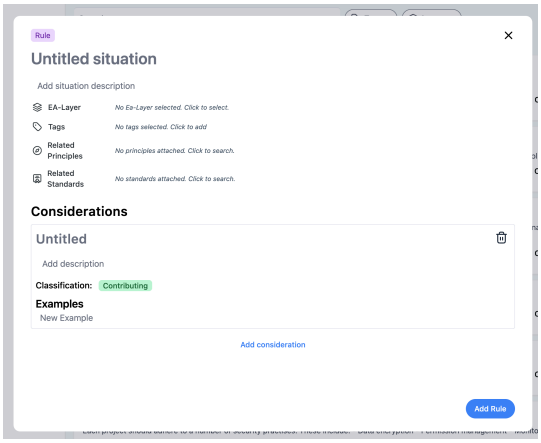

Figure 5.6: Rule creation

Filtering & searching for rules

Likewise the prescription management, the rules management supports the same filtering mechanism, as shown in Figure 5.7. The user can filter by the situation, the EA-Layer and the tags. The example filter in the figure shows a *keyword filter* which searches for the keyword *CI* in the title of the rules, a *tag filter* which filters for the tag *Docker* and a *EA-Layer filter* which filters for the EA-Layer *Implementation & Migration*.

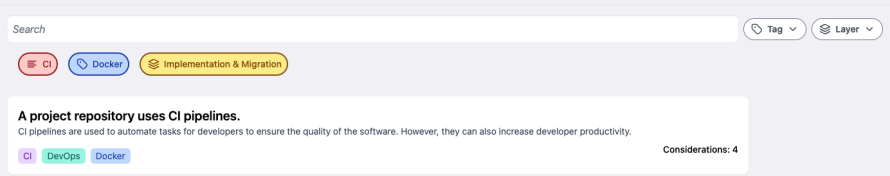

Figure 5.7: Rule filter

Editing rules

The user can edit or delete a rule by opening the detail modal of the rule by clicking on the rule. A view is opened, where the situation, considerations, EA-Layer, tags, related principles and related standards of the rule can be edited. The modal follows the same structure as the creation modal shown in figure 5.6, but all information is already filled with the current values of the rule. Additionally, the rule can be removed from the database. When a rule is removed from the database, it will only be removed from the rules management, not from compliance checks. Once the user is done editing the rule, the updates will be transferred to the application layer and persisted in the database. Updated rules will not influence existing compliance checks, as the compliance checks keep a copy of the rules at the time of the compliance check.

5.2.3 Project & assessment management

The project management is responsible for managing the projects of the enterprise. Each project contains three sub-components: the project contract management, the project assessment management and the project budget management. The application features a *workspace* section (figure 5.8), where all projects can be managed. New projects can be created and by clicking on an existing project, the project management can be accessed.

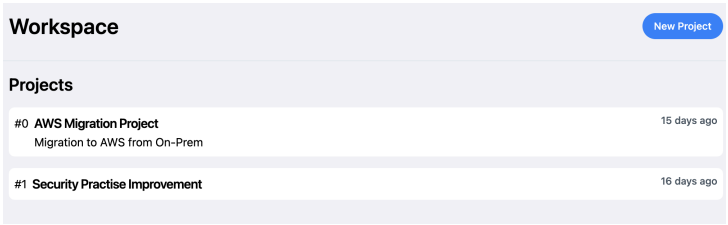

Figure 5.8: Workspace

Project contract management

The project contract management component is responsible for managing the contract of the project which will be used for the assessment. An example of a contract is shown in figure 5.9. The contract is a list of principles and standards which are relevant to

the project. The left side of the component shows the current contract of the project and the right side provides a search section to find new principles and standards which can be added to the contract. The search section uses the principles and standards from the persistence layer to search for principles and standards. It features the same filtering mechanism as in the prescription management, which enables the user to search for principles and standards by their title, EA-Layer or tags. In the example shown, the user searches for prescriptions related to the tag *Docker* and the EA-Layer *Infrastructure*. When the user clicks on a principle or standard from the search result, it will open a detail screen where the prescription can be added to the contract. The user can remove a principle or standard from the contract by clicking on the *Remove* button in the detail screen of the contract. Contract changes are saved by using the API endpoints which are provided by the project + assessment management component. In section 3.2 the concept of a contract snapshot was introduced. When a contract is changed, it will not influence existing contract snapshots of assessments. By this, the contract of an assessment is always the same as the contract of the project at the time of the assessment.

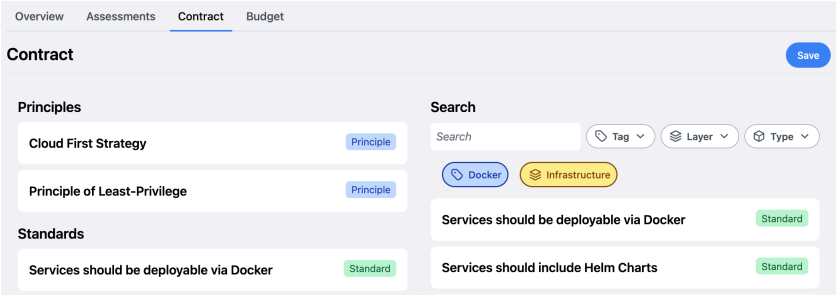

Figure 5.9: Project contract overview

Project budget management

The project budget management component is responsible for managing the budget of the project which will be used for the assessment. As explained in the section 4.2 the budget is a nested budget tree model which consists of subpartitions. The project budget management component provides the required UI components to manage the budget tree. Additionally to the management UI, the component provides an interactive *Sunburst Chart* to visualize the budget tree, as shown in figure 5.10. The user interface can be used to create, edit and delete budget partitions. As with the contract management, budget changes are saved by using the API endpoints which are provided by the project + assessment management component. Changes to the budget will not influence existing budget snapshots of assessments.

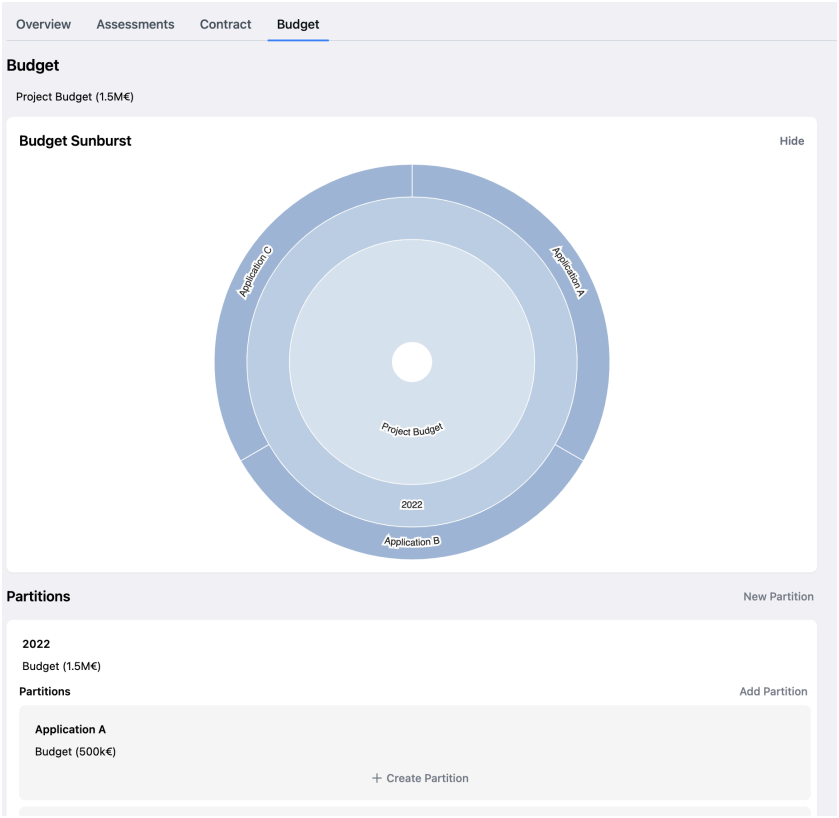

Figure 5.10: Project budget

Project assessment management

The next, and probably most important component of the project management is the project assessment management. The project assessment management component is responsible for managing the compliance assessments of the project. The component provides the required functionality to perform compliance assessments.

As mentioned in the section 4.2, projects should not be assessed at once, but rather at different stages of the project lifecycle. Therefore, as figure 5.11 shows, the project assessment management component provides the required functionality to manage multiple assessments of a project. When the user creates a new assessment, optionally all assessed items from the previous assessment can be copied to the new assessment. This feature prevents the user from having to assess the same items again and again. In case the previous assessment is copied, the application layer will fetch the previous assessment from the persistence layer and copy the assessment items to the new assessment.

Referring to the section 4.2, each project assessment consists of three parts: the assessment items, the project contract snapshot and the project budget snapshot. At the time a new assessment is created, the project contract snapshot and the project budget snapshot are created automatically by the application layer by copying the current

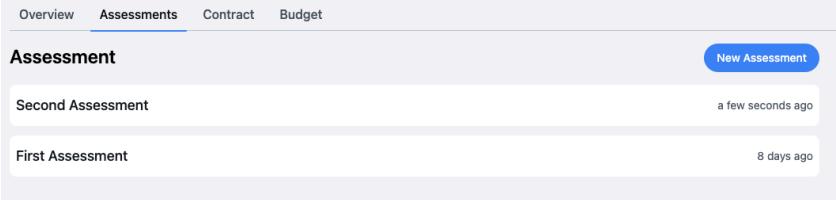

Figure 5.11: Project assessment overview

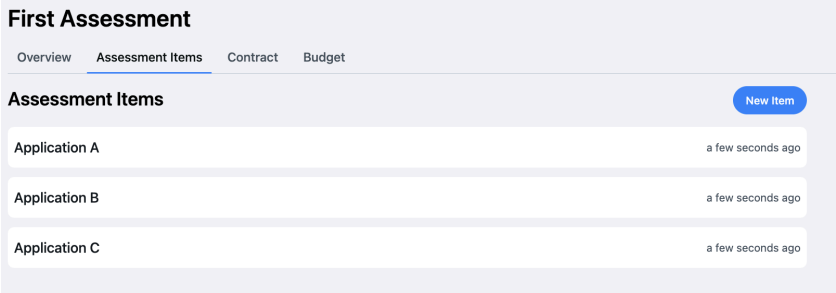

Figure 5.12: Project assessment overview

contract and budget of the project. The assessment items need to be entered manually by the user. This corresponds to the first step in the *Compliance Assessment Process* (section 4.3). Within the assessment item, the actual compliance checks are performed.

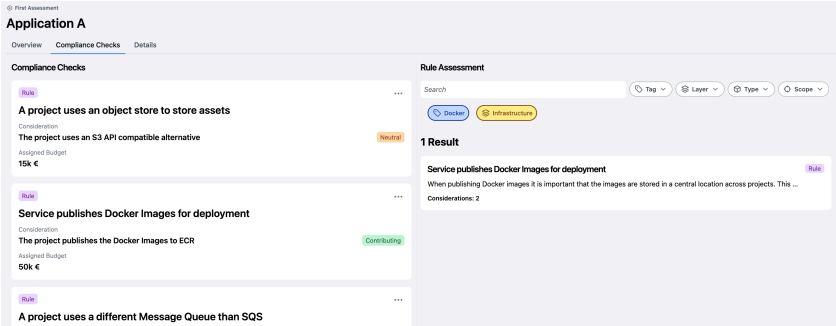

Figure 5.13: Compliance assessment component

Figure 5.13 shows a component of the application where the compliance checks are performed. The next step in the *Compliance Assessment Process* (section 4.3) is to select the prescription which should be checked. This step is not actually performed within the application, but rather by the user. However, the application can support the enterprise architect with the project contract management component (figure 5.9) to select the correct prescription.

The left side of the compliance assessment component (5.13) shows the already performed compliance checks of the assessment item. The right side provides the UI compo-

nents to perform a new compliance check. Referring back to the section 4.3, the next and third step in the *Compliance Assessment Process* is to find an applicable rule. This step is performed by using the right side of the compliance assessment component. The user can search for rules by using the same filtering mechanism as in the rules management component (figure 5.7). Once an applicable rule is found, the user can start to create the compliance check using the rule.

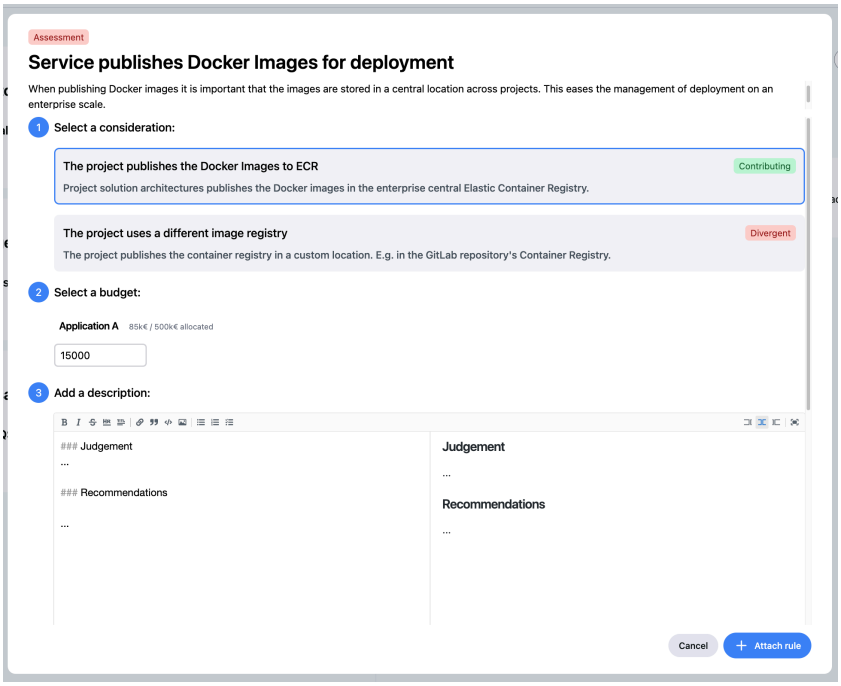

Figure 5.14: Compliance check component

Assessing a rule is divided into a three-step process in the application. The first step is to select the consideration of the rule which is applicable to the current situation. In the example shown in Figure 5.14, this step can be seen in the *Select a consideration* section. The second step is to select the budget partition and allocate the investment costs. This step can be seen in the *Select a budget* section. The application provides the required UI components to select the budget partition and allocate the investment costs. It also shows the already allocated budget, to prevent the user from allocating more budget than is available. The third and last step in the application combines the last three steps in the *Compliance Assessment Process* (section 4.3). The application provides a markdown editor to enter the judgment, the justification and further recommendations for the project stakeholders. Once the user has finished the three steps, the compliance check is saved and can be viewed on the left side of the compliance assessment component.

5.2.4 Analytics

The last component of the application is the analytics component. Once the enterprise architect has performed the compliance assessments, the analytics component can be used to analyze the results. The analytics component uses the compliance checks created by the projects & assessments component to generate the required data for the analytics. This is done by fetching the compliance checks from the persistence layer. The analytics API is not used to generate the data for the analytics. Instead, the analytics API is used to collect a dataset of compliance checks which is then used in the presentation layer to visualize the analytics.

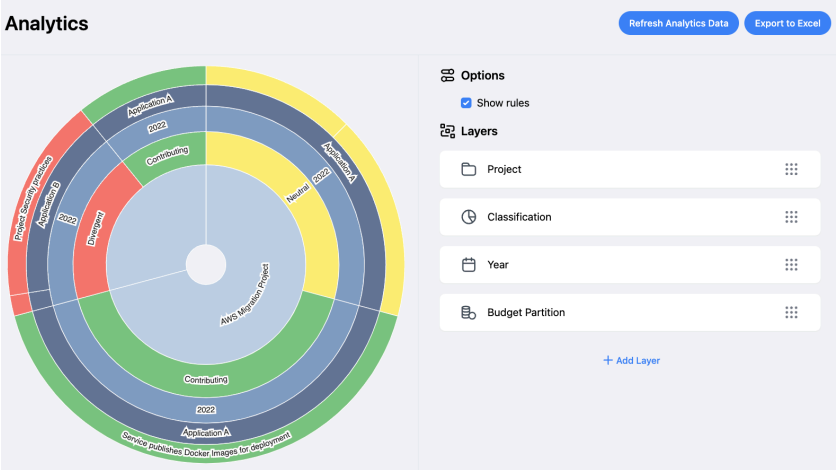

Figure 5.15: Analytics component

In general, the analytics are visualized in an interactive *Sunburst Chart* as shown in figure 5.15. The foundation for the analytical data are all compliance checks in the database. Each layer of the sunburst chart groups the set of compliance checks by a specific attribute. The following *Group by* layers are available:

- *Project*: Groups the compliance checks by the project.
- *Year*: Groups the compliance checks by the year.
- *Month*: Groups the compliance checks by the month.
- *Classification*: Groups the compliance checks by the classification of the selected consideration.
- *Budget Partition*: Groups the compliance checks by the budget partition associated with the selected consideration.

All layers can filter the compliance checks by excluding certain values. This is done by opening the layer and deselecting the values that should be excluded. The relative

size of the sunburst chart segments is determined by the amount of budget allocated to the compliance checks.

A powerful feature of the analytics component is the ability to reorder the layers of the sunburst chart. This means that users can change the way the data is presented. For example, the enterprise architect can group the compliance checks by the project and then by their classification to see how the project is contributing or the enterprise architect can group the compliance checks by the classification and then by the project to see how the overall compliance of the enterprise is distributed.

Using the analytics component

Important to note that the analytics tool, as it is implemented in this prototype, it does not have a specific audience in mind. The goal is to provide a tool that can be used by different stakeholders to analyze and visualize the compliance results according to their needs. Therefore, the analytics component is not limited to a specific set of users. As an example, a stakeholder who is responsible for a specific project can use the *Project* layer to filter only the compliance checks of their project. This way, the stakeholder can see how the project is contributing to the overall compliance of the enterprise. Another example would be a stakeholder who is responsible for financial planning and budgeting. This stakeholder can use the *Year* and *Classification* layers, for example, to analyze sunk costs of specific years.

6 Evaluation

Contents

6.1 Setup 45
6.2 Evaluation Results 48

6.1 Setup

To evaluate the Rule-based compliance assessment tool, we have used an interactive video format with embedded questions into the video. The video was forwarded to the participants and they were asked to answer the questions or statements while watching the video.

In general, the video is about 30 minutes long. It covers every component that was presented in chapter 5. The video can be found using the following link: `https://app.screencast.com/MHcC8jsPwYFVj` (accessed on 2023-01-03). The interactive video is separated into five sections followed by a survey to collect feedback after each section.

The questions in the video can be answered on a scale from 1 to 5, where 1 means "strongly disagree" and 5 means "strongly agree". The questions are about the usability of the software prototype and usefulness of either the software prototype or concepts of the compliance framework.

Principles & Standards

The first section is about the principles and standards component of the software prototype. This section introduces the management of principles and standards and the way they are used in the software prototype. Furthermore, it explains the way metadata is attached to the principles and standards. The questions after this section are the following:

- Attaching EA-Layers to Principles and Standards helps to improve the management of prescriptions.
- Tagging prescriptions with user-defined tags helps to improve the management of prescriptions, when dealing with a large number of prescriptions?

Both questions aim to find out if the metadata serves its purpose, which is to improve the management of prescriptions, especially when dealing with a large number of prescriptions.

Local Projects

The second section is about the local projects component of the software prototype. In this section project management is presented, along with the creation of the project contract and the project budget. The statements after this section are the following:

- Contracts help enterprise architects to better manage the principles and standards a project should conform with.
- A Nested Budget Partition Model is a good way to manage the project budget.

The first statement helps to find out if the project contract is a good way to manage the principles and standards that are relevant to the project. The second statement helps to find out if organizing the project budget in a nested budget partition model is a good way to manage the project budget.

Rules Management

The third section is about the rules management component of the software prototype. The concept of rules is presented in this section and how they are created and managed in the software prototype.

- Formalizing commonly occurring situations into Assessment Rules can help to increase the reproducibility of compliance assessments.
- Adding related EA-Layers to Rules as a filtering mechanism helps to manage a large number of rules.
- Tagging rules with user-defined tags helps to manage a large number of rules.
- Classifying rule considerations into contributing, neutral, and divergent is sufficient to perform compliance assessments.

The first statement helps to find out if the idea of formalizing commonly occurring situations into rules is a good way to increase the reproducibility of compliance assessments. The second and third statement aims to find out if the attached metadata is a good way to manage a large number of rules. The fourth statement helps to find out if the three classifications of considerations are sufficient to perform compliance assessments or if there are more/ other classifications that are needed.

Assessment

The fourth section is about the assessment component of the software prototype. This section explains and presents the compliance assessment process. It presents how rules can be used to assess the assessment items and how the metadata can be used to improve the assessment process. The statements after this section are the following:

- Dividing an assessment into multiple assessment items is useful to manage multiple scopes of the assessment.
- Using the attached metadata is a good mechanism to search and filter for rules to assess.
- Scoping the search results to "Contract" helps to efficiently find rules that are relevant to the scope of the project.
- Allocating the investment cost to the compliance checks helps to rank the compliance checks by their importance.

The first statement helps to find out if the general model of an assessment consisting of multiple assessment items serves its purpose to manage multiple scopes of the assessment. The second statement helps to find out if the metadata of the rules, including EA-Layers and tags, can be used to efficiently find rules that are relevant to the scope of the assessment. The third statement aims to find out if the concept of the project contract helps to further improve the search results by only showing rules that are relevant to the scope of the project. The fourth statement tries to find you if the allocation of the investment cost to the compliance checks helps to generate a ranking of the compliance checks by their importance.

Analytics

The last section is about the analytics component of the software prototype. Once the assessment process is introduced in the previous section, the analytics component is presented. This section explains how the analytics component can be used to generate insights about the performed assessments.

- Visualizing the conformance results in a sunburst chart provides highly informational valued analytical insights.
- The supported "Group By" layers are sufficient enough to generate in-depth analytical data.
- Being able to reorder the layers of the sunburst chart improves the informational value of the generated insights.

The first statement aims to find out if the general concept of visualizing the conformance results in a sunburst chart provides a good way to generate analytical insights. The second and third statement tries to find out if the configuration options that are available in the software prototype improve the analytical insights that can be generated.

6.2 Evaluation Results

The results of the evaluation are presented in the following table 6.1. Unfortunately, due to technical issues, the results of the evaluation could not be collected properly. Therefore, the results are based on the feedback of one participant. Even though this participant had experience in both research and practice in the field of enterprise architecture, the representativeness of the results is limited due to the small sample size.

Question/ Statement	Results	Remarks
Attaching EA-Layer metadata to Principles and Standards helps to improve the management of the prescriptions?	Disagree	It is not clear, why the EA Layer needs to be a separate kind of tags.
Adding user-defined tags to Principles & Standards helps to manage the prescriptions, especially when dealing with a large number of prescriptions?	Strongly Agree	
Contracts help enterprise architects to manage the principles and standards a project should conform with.	Disagree	Entering the information causes effort. Collecting this information might also be an issue.
A Nested Budget Partition Model is a good way to manage the project budget.	Disagree	
Formalising commonly occurring situations into Assessment Rules can help to increase the reproducibility of compliance assessments.	Neutral	The tags in general are useful.
Adding related EA-Layers to Rules as a filtering mechanism helps to manage a large amount of Rules.	Strongly Disagree	
Tagging Rules with user-defined tags help to manage a large amount of Rules.	Strongly Agree	
Classifying Considerations into Contributing, Neutral and Divergent is sufficient to perform compliance assessments.	Disagree	
Dividing an assessment into multiple assessment items is useful to manage multiple scopes of an assessment.	Neutral	Regarding the costs, it is great to have that kind of information. However, it might be hard to gather this kind of information in a real-world project.
Using the attached metadata is a good mechanism to search and filter for relevant rules to assess.	Agree	
Scoping the search results to *Contract* helps to efficiently find rules that are relevant in the scope of the project.	Strongly Disagree	
Allocating the investment cost to the compliance checks helps to rank the compliance check by their importance.	Neutral	
Visualising the conformance results in a sunburst chart provides high informational valued analytical insights.	Strongly Disagree	The visualisation needs a purpose and a target audience.
The supported *Group By* layers are sufficient enough to generate in-depth analytical data.	Strongly Disagree	
Being able to reorder layers of the sunburst chart improves the informational value of the generated insights.	Strongly Disagree	

Table 6.1: Evaluation Results

7 Discussion

Contents

7.1 Discussion of the Results 51
7.2 Implication 53
7.2.1 Implications for Researchers 53
7.2.2 Implications for Practitioners 53
7.3 Threats to Validity 54
7.3.1 Threats to Internal Validity 54
7.3.2 Threats to External Validity 54

In this chapter, the results of the evaluation are discussed.

7.1 Discussion of the Results

Principles & Standards

The first section primarily focused on metadata management and the principles and standards management. As mentioned in chapter 4 prescriptions have metadata attached to them. The metadata consists of EA-Layers or user-defined tags.

- The feedback raises the question if separating the metadata into EA-Layers and user-defined tags are necessary. It was stated that the user-defined tags are sufficient enough to manage the prescriptions since they can be used to group the prescriptions into domains, products or divisions. I agree that the difference between the EA-Layers and the user-defined tags is not that big. However, I think that the EA-Layers are not useless as they can be used to group the prescription by their domain.

Local projects

The second section focused on two components of the software prototype. The first component is the project contract and the second component is the project budget.

- The first feedback was about the project contract. It was stated that the context of the project contract is not clear. It was not clear who is involved in the project contract.
- The second feedback was about the project budget. The nested project budget seems to be useful. However, it was stated that the project budget might be

hard to collect from the financial department. Collecting the budget might be a problem since the finance department might not be familiar with the software prototype. Since I am not an enterprise practitioner, I cannot judge if collecting this information is a problem or not. However, I can imagine that this might become an issue when using this software prototype in a real-world scenario.

Rules management

The third section focused on the rules management.

- The first feedback was about the general concept of rules including the *Considerations* concept. The results of the evaluation show that classifying the considerations into *Contributing*, *Neutral* and *Divergent* might not be sufficient enough. One reason for this might be that *Neutral* covers a wide range of considerations. I agree with the feedback that the *Neutral* category might be too broad. This might lead to inconsistencies between the classifications of the considerations.
- Another part of the feedback was about the metadata of the rules. Similar to the feedback about the metadata of the prescriptions, it was stated that the EA-Layers metadata might not be necessary. User-defined tags might be sufficient enough to manage the rules. The same argument as in the feedback about the metadata of the prescriptions applies here. The EA-Layers metadata can be used to group the rules by their domain.

Assessments

The fourth section focused on the compliance assessments of solution architectures.

- The general concept of dividing the assessment into multiple assessment items was rated as neutral. However, I think that the division of the assessment into multiple assessment items definitely has advantages. The division of the assessment into multiple assessment items allows the practitioner to focus on a specific part of the assessment.
- The second feedback was about the usage of rules metadata in the assessment. It was stated that the usage of the rules metadata in the assessment helps to search and filter for relevant rules. However, the usefulness of the contract scope was doubted. The contract scope is used to filter the rules by the project contract. I do not agree with the feedback that the contract scope is not useful. In case the project contract is available it can be used to efficiently bring down the search space of the rules to only take the rules into account that are relevant to the project. In case the project contract is not available, the filter cannot be applied and therefore doesn't have any effect.
- An additional discussion point is about the allocated budget for the compliance check. Here the same issue appears as in the project budget. Attaching this

financial information is good, but collecting them from the finance department might be a problem. Again, as mentioned before, I do not have enough experience to judge if this is a problem or not.

Analytics

The fifth and last section focused on the analytics of the compliance assessments.

- It was stated that the sunburst chart might not meet the target audience's needs. Unfortunately, I think that there was a misunderstanding in the evaluation. The analytics component was specially designed not to have a target audience and to be as generic as possible, to be able to generate insights for various stakeholders. As briefly shown in chapter 5.2.4, different stakeholders can configure the tool to generate insight for their needs.

7.2 Implication

In this section, the implications of the results are discussed. We will differentiate between implications for researchers and implications for practitioners.

7.2.1 Implications for Researchers

This thesis focused on the *Identification & Collection* phase of the EADM framework presented by [Ale+20]. It presented a concept for identifying EAD in a more reproducible way by using a rule-based approach that aims to reduce personal influences during the identification of potential EAD. This leads to a more consistent output of the *Identification & Collection* phase, which is used as input for the following phases of the EADM framework. However, there is room for improvement. The presented solution still has a certain amount of subjectivity in the assessment rules. This subjectivity opens up the possibility for inconsistencies in the compliance assessments. Researchers could research how this subjectivity can be reduced to further increase the reproducibility and consistency of the compliance assessments.

7.2.2 Implications for Practitioners

The compliance model presented in this thesis allows practitioners to identify EAD in a more reproducible way. It allows practitioners to manage the principles and standards, projects and rules in a software prototype. Furthermore, it has support to manage a large number of prescriptions and rules. The generic analytics tool allows practitioners to generate insights for various stakeholders. The insights can be used to communicate the compliance status of the solution architecture to the stakeholders.

7.3 Threats to Validity

This section discusses the threats to validity. The threats to validity are divided into threats to internal validity and threats to external validity.

7.3.1 Threats to Internal Validity

The goal of the thesis was to develop a solution concept to make compliance assessments more reproducible. The presented solution concept was developed on a rule-based approach. A potential threat to the internal validity of the solution concept is that the rule-based approach still has some subjectivity. The rules including the situation, considerations and classifications have to be entered by the practitioners. This might lead to inconsistencies between classifications of considerations, as the classification is again influenced by the personal experience & knowledge of the practitioner. As an example, a practitioner might classify a certain consideration as *Neutral* while another practitioner might classify the same consideration as *Contributing*. This inconsistency might lead to inconsistent compliance assessments.

7.3.2 Threats to External Validity

Looking back on how the solution concept and the prototype were developed, we have based the solution on a framework used by one organization. As it has been raised multiple times in this thesis, a common problem is that processes and terms vary between organizations (e.g. the definition of *compliant* [Groa]). This is a potential threat to the external validity of the solution, as different organizations might have different requirements and resources available for the compliance management. Therefore, the solution might not apply to the same extent in other organizations.

8 Conclusion

Contents

8.1 Summary 55
8.2 Future Work 55

In this final chapter, the thesis will be summarized and after that future work will be presented which can be done in the field of compliance assessments.

8.1 Summary

In this thesis, a rule-based approach for compliance assessment framework for solution architectures was developed. The goal was to support enterprise architects in the assessment of solution architecture alignment. The thesis proposes the use of assessment rules which can be used to improve the reproducibility of compliance assessments. The compliance framework that was developed includes four different components: prescription management, rule management, project and assessment management and the analytics component. The first three components are explained in chapter 4, as they form the foundation of the compliance framework. The analytics component is explained in chapter 5 since it is derived from the data generated by the other three components. Additionally to the solution concept, a software prototype was developed which was presented in chapter 5. The chapter explained how the solution concept developed in chapter 4 was implemented in a software prototype.

Finally, an evaluation was conducted to evaluate the compliance framework. The evaluation revealed a few issues that might occur when applying the compliance framework in practice. The issues primarily concern the availability of the required data which is needed to perform the compliance assessment. These issues were discussed in chapter 7.

8.2 Future Work

The compliance framework that was developed in this thesis is a step toward a rule-based compliance assessment framework. The compliance framework can be used to assess the alignment of solution architectures with the enterprise architecture. However, since the compliance framework proposed in this thesis still has some issues, there are parts for future work.

The first part of future work is to improve the classification of the assessment rules. Currently, the compliance framework supports the classifications: *Contributing*, *Neutral*

and *Divergent*. However, as the classification *Neutral* covers a wide range of considerations, there might be a need for more fine-grained classification. The classification of the assessment rules can be improved by using a more fine-grained classification, especially focusing on splitting the *Neutral* classification into multiple classifications.

The second part of future work is to look into the analytical output of the compliance framework. The software prototype presented in chapter 5 provides an interactive sunburst chart that allows different views of the alignment of the solution architectures. However, even though the sunburst chart provides a mechanism to visualize the data in multiple ways, it might not meet the requirements of all stakeholders that are involved when discussing the assessment reports.

The third part of future work could be to exploit the compliance assessment process in a real-world scenario. Even though the software prototype was tested for its functionality, it was not tested at a large scale. It would be interesting to see how the features, which support the management of large amounts of prescriptions, rules and assessments, scale in a real-world enterprise architecture management environment.

Bibliography

[Ale+20] P. Alexander et al. "A Framework for Managing Enterprise Architecture Debts-Outline and Research Directions." In: *EMISA*. 2020, pp. 5–10 (cit. on pp. 2, 8, 9, 53).

[Bru+10] W. A. Bruls et al. "Domain architectures as an instrument to refine enterprise architecture." In: *Communications of the Association for Information Systems* 27.1 (2010), p. 27 (cit. on p. 1).

[BY06a] W. F. Boh and D. Yellin. "Using enterprise architecture standards in managing information technology." In: *Journal of Management Information Systems* 23.3 (2006), pp. 163–207 (cit. on pp. 1, 2).

[BY06b] W. F. Boh and D. Yellin. "Using enterprise architecture standards in managing information technology." In: *Journal of Management Information Systems* 23.3 (2006), pp. 163–207 (cit. on pp. 18, 19).

[ČR12] V. Čyras and R. Riedl. "Formulating the enterprise architecture compliance problem." In: *Databases and Information Systems BalticDB&IS* 2012 (2012), pp. 142–153 (cit. on p. 12).

[Dei+09] C. Deiters et al. "Rule-based architectural compliance checks for enterprise architecture management." In: *2009 IEEE International enterprise distributed object computing conference*. IEEE. 2009, pp. 183–192 (cit. on p. 11).

[FFB19] H. Foidl, M. Felderer, and S. Biffl. "Technical debt in data-intensive software systems." In: *2019 45th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*. IEEE. 2019, pp. 338–341 (cit. on p. 6).

[Foo+12] R. Foorthuis et al. "Compliance assessments of projects adhering to enterprise architecture." In: *Journal of Database Management (JDM)* 23.2 (2012), pp. 44–71 (cit. on pp. 1, 2, 9, 12–15, 19–22).

[FWJ20] P. Filet, R. van de Wetering, and S. Joosten. "Enterprise architecture alignment." In: *Enterprise Architecture and Service-Oriented Architecture*. Nova Southeastern University, 2020 (cit. on p. 11).

[Groa] T. O. Group. *Architecture Compliance*. URL: `https://pubs.opengroup.org/architecture/togaf8-doc/arch/chap24.html` (cit. on pp. 9, 54).

[Grob] T. O. Group. *Architecture Compliance Review*. URL: `https://www.opengroup.org/architecture/togaf7-doc/arch/p4/comp/comp.htm#Reviews` (cit. on p. 13).

[Groc] T. O. Group. *Architecture Domain*. URL: `https://pubs.opengroup.org/architecture/togaf9-doc/arch/chap03.html#tag_03_11` (cit. on p. 19).

[Grod] T. O. Group. *Architecture Domains*. URL: `https://pubs.opengroup.org/architecture/togaf9-doc/arch/chap02.html` (cit. on p. 19).

[Groe] T. O. Group. *Architecture Principles*. URL: `https://pubs.opengroup.org/architecture/togaf9-doc/arch/chap20.html` (cit. on pp. 12, 19).

[Gro11] T. O. Group. "The Open Group Architecture Framework." In: *The Open Group* (2011) (cit. on p. 6).

[Hac+19] S. Hacks et al. "Towards the definition of enterprise architecture debts." In: *2019 IEEE 23rd International Enterprise Distributed Object Computing Workshop (EDOCW)*. IEEE. 2019, pp. 9–16. URL: `https://arxiv.org/pdf/1907.00677.pdf` (cit. on p. 7).

[Joh+04] P. Johnson et al. "Using enterprise architecture for cio decision-making: On the importance of theory." In: *Second Annual Conference on Systems Engineering Research*. 2004 (cit. on p. 1).

[Jon+06] H. Jonkers et al. "Enterprise architecture: Management tool and blueprint for the organisation." In: *Information systems frontiers* 8.2 (2006), p. 63 (cit. on p. 5).

[Kot18] S. Kotusev. "TOGAF-based enterprise architecture practice: an exploratory case study." In: *Communications of the association for information systems* 43.1 (2018), p. 20 (cit. on p. 6).

[Kru+13] P. Kruchten et al. "Technical debt: towards a crisper definition report on the 4th international workshop on managing technical debt." In: *ACM SIGSOFT Software Engineering Notes* 38.5 (2013), pp. 51–54 (cit. on p. 6).

[Mar] Markdown. *Markdown*. Accessed: 2022-12-12. URL: `https://www.markdownguide.org/basic-syntax/` (cit. on p. 33).

[Mona] MongoDB. *MongoDB*. Accessed: 2022-12-12. URL: `https://www.mongodb.com/home` (cit. on p. 31).

[Monb] Mongoose. *Mongoose*. Accessed: 2022-12-12. URL: `https://mongoosejs.com/` (cit. on p. 32).

[Nex] Next.js. *Next.js*. Accessed: 2022-12-12. URL: `https://www.nextjs.org/` (cit. on p. 31).

[PG10] E. Proper and D. Greefhorst. "The roles of principles in enterprise architecture." In: *International Workshop on Trends in Enterprise Architecture Research*. Springer. 2010, pp. 57–70 (cit. on p. 6).

[Pru+14] L. J. Pruijt et al. "Husacct: Architecture compliance checking with rich sets of module and rule types." In: *Proceedings of the 29th ACM/IEEE international conference on Automated software engineering*. 2014, pp. 851–854 (cit. on p. 11).

[Rea] React. *React*. Accessed: 2022-12-12. URL: `https://reactjs.org/` (cit. on p. 31).

[Rio+20] N. Rios et al. “Hearing the voice of software practitioners on causes, effects, and practices to deal with documentation debt.” In: *International Working Conference on Requirements Engineering: Foundation for Software Quality*. Springer. 2020, pp. 55–70 (cit. on p. 6).

[ST06] S. H. Spewak and M. Tiemann. “Updating the enterprise architecture planning model.” In: *Journal of Enterprise Architecture* 2.2 (2006), pp. 11–19 (cit. on p. 6).

[Typ] TypeScript. *TypeScript*. Accessed: 2022-12-12. URL: `https://www.typescriptlang.org/` (cit. on p. 31).

[Uni14] P. University. “Enterprise Architecture Compliance Review Process.” In: *Plymouth University* (2014) (cit. on p. 13).

[YNH08] T. Ylimäki, E. Niemi, and N. Hämäläinen. “Enterprise architecture compliance: The viewpoint of evaluation.” In: *Evaluation of enterprise and software architectures: critical issues, metrics and practices/Eetu Niemi, Tanja Ylimäki & Niina Hämäläinen (eds.). Jyväskylä: University of Jyväskylä, Information Technology Research Institute, 2008.-(Tietotekniikan tutkimusinstituutin julkaisuja, ISSN 1236-1615; 18). ISBN 978-951-39-3108-7 (CD-ROM)*. (2008) (cit. on p. 9).