

The present work was submitted
to the Research Group
Software Construction

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

Master Thesis

Reporting Framework for Enterprise Architecture Debts

Reporting Framework for
Enterprise Architecture Debts

presented by

Sana Kanji

Aachen, August 23, 2022

Examiner

Prof. Dr. rer. nat. Horst Lichter

Prof. Dr. rer. nat. Bernhard Rumpe

Supervisor

Peter Alexander, M.Eng.

Eidesstattliche Versicherung Statutory Declaration in Lieu of an Oath

Kanji, Sana
Name, Vorname/Last Name, First Name

407536
Matrikelnummer (freiwillige Angabe)
Matriculation No. (optional)

Ich versichere hiermit an Eides Statt, dass ich die vorliegende Arbeit/Bachelorarbeit/
Masterarbeit* mit dem Titel

I hereby declare in lieu of an oath that I have completed the present paper/Bachelor thesis/Master thesis* entitled

Reporting Framework for Enterprise Architecture Debts

selbstständig und ohne unzulässige fremde Hilfe (insbes. akademisches Ghostwriting) erbracht habe. Ich habe keine anderen als die angegebenen Quellen und Hilfsmittel benutzt. Für den Fall, dass die Arbeit zusätzlich auf einem Datenträger eingereicht wird, erkläre ich, dass die schriftliche und die elektronische Form vollständig übereinstimmen. Die Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

independently and without illegitimate assistance from third parties (such as academic ghostwriters). I have used no other than the specified sources and aids. In case that the thesis is additionally submitted in an electronic format, I declare that the written and electronic versions are fully identical. The thesis has not been submitted to any examination body in this, or similar, form.

Aachen, 23/08/2022
Ort, Datum/City, Date

Unterschrift/Signature
*Nichtzutreffendes bitte streichen
*Please delete as appropriate

Belehrung:

Official Notification:

§ 156 StGB: Falsche Versicherung an Eides Statt

Wer vor einer zur Abnahme einer Versicherung an Eides Statt zuständigen Behörde eine solche Versicherung falsch abgibt oder unter Berufung auf eine solche Versicherung falsch aussagt, wird mit Freiheitsstrafe bis zu drei Jahren oder mit Geldstrafe bestraft.

Para. 156 StGB (German Criminal Code): False Statutory Declarations

Whoever before a public authority competent to administer statutory declarations falsely makes such a declaration or falsely testifies while referring to such a declaration shall be liable to imprisonment not exceeding three years or a fine.

§ 161 StGB: Fahrlässiger Falscheid; fahrlässige falsche Versicherung an Eides Statt

(1) Wenn eine der in den §§ 154 bis 156 bezeichneten Handlungen aus Fahrlässigkeit begangen worden ist, so tritt Freiheitsstrafe bis zu einem Jahr oder Geldstrafe ein.

(2) Strafflosigkeit tritt ein, wenn der Täter die falsche Angabe rechtzeitig berichtigt. Die Vorschriften des § 158 Abs. 2 und 3 gelten entsprechend.

Para. 161 StGB (German Criminal Code): False Statutory Declarations Due to Negligence

(1) If a person commits one of the offences listed in sections 154 through 156 negligently the penalty shall be imprisonment not exceeding one year or a fine.

(2) The offender shall be exempt from liability if he or she corrects their false testimony in time. The provisions of section 158 (2) and (3) shall apply accordingly.

Die vorstehende Belehrung habe ich zur Kenntnis genommen:

I have read and understood the above official notification:

Aachen, 23/08/2022
Ort, Datum/City, Date

Unterschrift/Signature

Acknowledgment

I would like to express my gratitude to Prof. Dr. rer. nat. Horst Lichter for providing me with the opportunity to write a thesis in his department. I would also like to thank Peter Alexander, who suggested this interesting thesis project and helped me throughout the thesis. Also, a special thanks to a very dear friend of mine, Leon Akkermans for always pushing me to work hard and stay motivated. Finally, I would like to thank my family for providing immense support throughout my whole studies, this would not have been possible without them.

Sana Kanji

Abstract

Enterprise architecture debt (EA debt) is a metric that depicts the deviation of the current present state of the enterprise from a hypothetical ideal state. EA debt expands the concept of *Technical Debt (TD)* which focuses on only the technical aspects, by covering all the layers of an *Enterprise Architecture (EA)*. As the EA debt concept is relatively young and the *Enterprise Architecture Debt Management Framework (EADM)* is also undergoing research, therefore the goal of this thesis is to work on the monitoring, documentation & communication activities of the EADM, to have a clear mechanism for reporting of EA debts. An EA debt reporting framework is implemented to create a reporting mechanism specifically for EA debts. Each component in the EA debt reporting framework is described and the requirement it fulfills in order to explain the purpose of each component. Finally, a front-end application is also implemented to interact with and showcase the EA debt reporting framework. This thesis serves as a basis for having a continuous reporting mechanism for the EA debts, and further research that has to be done in this area. In the future, this can be connected with the other activities of the EADM.

Contents

1. Introduction	1
1.1. Motivation	1
1.2. Problem Statement	2
1.3. Research Questions	2
1.4. Contribution	3
1.5. Outline	4
2. Background and Related Work	5
2.1. Enterprise Architecture	5
2.2. Enterprise Architecture Debt	6
2.3. Debt Management & Reporting Tools	7
3. Concept	11
3.1. Requirements	11
3.2. Design of Framework	13
4. Realization	17
4.1. Architecture	17
4.2. EA Debt Reporting Framework Implementation	21
5. Evaluation	49
5.1. Setup for Evaluation	49
5.2. Evaluation Results	53
6. Discussion	57
6.1. Steps After Evaluation	57
6.2. Implications	61
6.3. Threats to Validity	62
7. Conclusion	65
7.1. Summary	65
7.2. Future Work	66
A. Debt Item Attributes	67
Bibliography	73
Glossary	77

List of Tables

5.1. Evaluators Background and Expertise	53
5.2. Evaluation Result Range	53
5.3. Evaluation Results	54
5.4. Evaluation Feedback	56

List of Figures

3.1. Conceptual View	13
3.2. Data Management Use Cases	15
3.3. Template Management Use Cases	15
3.4. Report Management Use Cases	16
3.5. Change Management Use Cases	16
4.1. EA debt reporting framework implementation architecture	18
4.2. Entity Relationship Diagram for EA debt reporting framework	21
4.3. Separation of concerns overview	22
4.4. Adapter Implementation Class Diagram	23
4.5. Treemap Example	25
4.6. State Diagram for EA debt Statuses	25
4.7. State Diagram for EA debt States	26
4.8. EA Debt State Timeline	26
4.9. EA debt Reporting Framework Front-end Application	33
4.10 Data Entry View to Upload an Excel Sheet	33
4.11 Debt Registry View	34
4.12 Debt Registry More Options	34
4.13 Debt Registry Table Add Columns	35
4.14 Debt Registry Saving the Filter	36
4.15 Debt Registry Loading the Filter	36
4.16 Debt Registry Loading Filter Options	37
4.17 Debt Registry Global Filter	37
4.18 Debt Registry Global Filter with Additional Filter	37
4.19 Dashboard View	38
4.20 Debt Item Details	39
4.21 Template Management View	40
4.22 Template Creation First Step	40
4.23 Template Creation Second Step	41
4.24 Widgets Selection Dialog	42
4.25 Template Creation Third Step	42
4.26 Template Added in Report Templates Table	43
4.27 Report Management View	43
4.28 Report Generation First Step	44
4.29 Report Template Selection Drop-Down	44

4.30	Report Filter Selection	45
4.31	Report Generation Second Step	45
4.32	Generated Report Preview	46
4.33	Notifications View	46
4.34	Notify Dialog	47
4.35	Received Notification View	47

List of Source Codes

4.1. Configuration for Column Chart Widget	28
4.2. Configuration for Table Widget	28
4.3. Configuration containing Table and Column Chart Widget	28
4.4. Report Metadata	30

1. Introduction

Contents

1.1. Motivation	1
1.2. Problem Statement	2
1.3. Research Questions	2
1.4. Contribution	3
1.5. Outline	4

In the introduction, we will start by presenting the motivation and the problem statement of this thesis. This section will provide a brief introduction to EA and EA debt and then discuss the problems that we are going to address in this thesis. Furthermore, this chapter will also discuss the research questions and the contribution of this thesis, so that the scope of the thesis is well-defined.

1.1. Motivation

Enterprises today are large socio-economic systems, with complex business processes and application landscapes, thereby making business-IT alignment a challenging task [Jun+21]. The EA discipline was introduced to support business-IT alignment by enabling transparency [Lan+05]. Since EA typically covers multiple organizational units and involves a plethora of applications and projects running in parallel, making the coordination, communication, and documentation complex and are therefore amongst the identified EA issues as stated in [LKL10; Rot+13]. The consequences that may arise due to the above-mentioned EA issues may result in compromises made in EA projects, thus not fulfilling the organization's expectations [BH19].

EA Debt term was introduced to track these compromises and to provide a common language for business and IT representatives. EA debt is defined as 'a metric that depicts the deviation of the currently present state of an enterprise from a hypothetical ideal state.' [Hac+19]. As EA debt is a relatively young field, a framework for communicating and reporting these EA debts has not yet been proposed. Therefore, in this thesis, we are going to research into developing a framework for communicating and reporting EA debts; this framework will help in raising awareness of the organization's EA debts by enabling the creation of EA debt reports and the continuous check of their validity.

The framework we will be researching, the EA debt reporting framework will cover three of the activities from the EADM framework [Ale+20], namely: monitoring, documentation, and communication.

1.2. Problem Statement

Managing EA debt requires a common understanding among the business & IT stakeholders, and this could be achieved in the form of continuous reporting, which will be sent whenever there are changes in EA debts in the organization. Such a reporting mechanism for EA debt has not been proposed yet; therefore, we need to research and create a process or a mechanism for reporting EA debts to the relevant stakeholders. Furthermore, to effectively report on the EA debts within the organization, the process or mechanism will need access to EA debt registries in the organization; hence we need to research how to keep track of these EA debt registries.

1.3. Research Questions

The problem statement stated the need to have a clear process and mechanism for reporting EA debts. To develop such a process, we need to consider certain steps. The research questions will help in understanding what is required and the problems we need to answer to guide our actions towards attaining the goal.

RQ1: What concerns should the EA debt reporting framework support?

This research question will help us to identify and verify the stakeholder requirements that must be supported in the EA debt reporting framework. We need this to know the obstacles faced by the stakeholders and their wants that will help reduce these obstacles. This will enable us to make the blueprint for the framework and analyze the concerns that need to be answered from an implementation perspective.

RQ2: How to implement a continuous reporting of EA debts?

Since EA debt should be among the topics which must be communicated continuously to stay updated about the organizations current state. Hence, this research question will help develop a strategy to keep the EA debt reporting continuous so that there is a clear process for reporting EA debts, and the process accommodates the changes that happen frequently.

RQ2.1: What will be the lifecycle of EA debts?

As the EA debts might undergo changes and may become critical or mitigated with time. We need to have a mechanism to see the changes in the EA debts and the stages that the EA debt goes through to get mitigated or discarded. We need to know this so that the details regarding the EA debt can be reported to the concerned stakeholders at the right time.

RQ2.2: What will be the lifecycle of EA debt reports?

After identifying the lifecycle of EA debts, we need to provide updates regarding the changes in an existing EA debt or if a new EA debt gets introduced. These changes in EA debts may result in EA debt reports becoming obsolete, and to know this, we need to identify the different stages of an EA debt report.

RQ2.3: How to make the EA debt reporting process continuous?

The poor communication between business and IT may lead to misunderstandings, which could result in the EA debt Lack of collaboration between business and IT [Jun+21]. This happens due to missing strategies and processes, so in this research question, we need to understand how to keep the reporting of EA debts continuous, that is, a process that needs to be used to keep the concerned stakeholder updated throughout the lifecycle of EA debts. This will, as a result, help in attaining common grounds and minimizing the misunderstandings that lead to this lack of communication and collaboration regarding EA debts.

1.4. Contribution

This thesis contribution is to provide an extensible framework for the reporting of EA debts which includes the process of uploading EA debt registries, searching and filtering through the provided EA debt registries, the ability to make reports, and a way to keep the process continuous. This implementation will cover the monitoring, documentation, and communication activities as introduced in EADM [Ale+20]. The second contribution to this thesis will be a front-end application that will provide a visual presentation for functionalities supported by the EA debt reporting framework.

1.5. Outline

This report presents a mechanism for reporting EA debts. This report is structured as follows:

In the second chapter, we introduce the background of EA debts and discuss the other debt management and reporting tools. In the third chapter, we look into the requirements and discuss the components and use cases in the form of a conceptual and process view. The fourth chapter is the realization, where we discuss the implementation details used for the EA debt reporting framework. Next, in the fifth chapter, we examine the evaluation strategy and the results. In the sixth chapter, we discuss the feedback received in the evaluation, the implications, and the threats to validity. Lastly, in the seventh chapter, we provide the conclusion and the future work directions.

2. Background and Related Work

Contents

2.1. Enterprise Architecture	5
2.2. Enterprise Architecture Debt	6
2.3. Debt Management & Reporting Tools	7

In this chapter, we are first going to look further into the concepts of EA and EA debt and then discuss some debt management and reporting tools to know if we can further extend these tools to accommodate the solution for the problem that we are going to solve in this thesis.

2.1. Enterprise Architecture

Enterprise architecture (EA) is a description of an enterprise [KSS15], which provides a high-level view of an enterprise’s business processes and IT systems [Tam+11]. It provides tools and methods to align business with IT to attain innovation in the enterprise [Lap12]. This high-level view of an enterprise is provided using architecture artifacts such as model diagrams, also referred to as the architecture blueprint [BBL12] or EA models. These EA models are used as the decision-making tools providing stakeholders with the relevant information about the enterprise [SSS17]. This information entails the current state of the enterprise, the target state to be achieved by the enterprise, and the roadmap to transition from the current state to the target state, along with transient states in between [BBL12].

Enterprise Architecture Management (EAM) is a structured approach used by the enterprise to create, manage, and use the EA to align business and IT. EAM helps translate the enterprise vision into a journey and makes an enterprise go through the journey from its current state to the target state [BBL12]. Even though a lot of research has been done in the discipline of EA and its management, the attainment of the business and IT alignment is still challenging.

The significant findings regarding the challenges in EA and its management are communication, which exists since EA requires coordination across multiple enterprise units, managing the plethora of stakeholders from different hierarchical positions in the enterprise, and having different understandings of the requirements, which makes the communication very challenging. Another challenge is, as EA models are used to provide a holistic view of the enterprise; these models may lead to

the ‘ivory tower’ syndrome due to their complexity which may result in models being too abstract [Rot+13], making the reading and updating of models more complex [SSS17]. This may also result in poor EA documentation because it is too time-consuming and costly to maintain [Rot+13].

Several EA frameworks and tools have been established to overcome the above mentioned issues, but they describe the process of EA from a theoretical perspective. Lisa et al. [UM06] discuss the comparison of enterprise architecture frameworks in which they refer to several frameworks like Zachman framework, *The Open Group Architectural Framework (TOGAF)*, *Federal Enterprise Architecture Framework (FEAF)* and *Department of Defense Architecture Framework (DoDAF)*. Zachman framework for enterprise architecture comprises several views: Planner, Owner, Designer, Builder, Subcontractor, and User, and focuses on ensuring that all views are well established. But it does not provide any details regarding the process or implementation of these views. TOGAF consists of Architecture Development Method (ADM) that specifies the process for developing enterprise architecture [THC04], but it focuses on the good principles rather than providing the set of architecture principles [UM06]. FEAF is more focused on architecture planning, and the overall approach to architecture rather than the enterprise architecture [THC04]. DoDAF framework was designed for supporting the defense operations; therefore, few of its processes are domain dependent. DoDAF, even though it focuses on supporting the documentation of architecture models, still lacks the provision for recording architecture rationale and does not have architecture modeling capabilities [THC04]. Hence, the enterprises still struggle with the documentation of EA, because of the high level of abstraction in the EA models, due to the lack of architectural design details in EA frameworks, as mentioned above. Further communication of the EA to different levels of stakeholders, having diverse concerns, is equally difficult [Rot+13].

Therefore, there is still a need to document EA with varying levels of abstraction to communicate to the stakeholder with the information their hierarchical position requires [HBA17].

2.2. Enterprise Architecture Debt

As discussed in the above section, EA is gaining significant attention as the management instrument in business and IT [LMR16]; however, implementing EA may not always be successful due to uncertainty and unavailability of resources; therefore, a new metaphor named Enterprise Architecture Debt (EA debt) was introduced to relate to the consequences that occur when trade-off decisions are made in implementing an EA.[Hac+19]

EA debt is a metric that depicts the deviation of the currently present state of an enterprise from a hypothetical ideal state [Hac+19]. This term was inspired by

the term Technical Debt (TD), a metaphor that Ward Cunningham proposed in 1992 to describe past technical shortcuts that hamper IT developments [Cun92] [Lap12]. After gaining a lot of interest in the software development industry, the term is now extended to include software documentation, architecture, requirements, and testing [Bro+10]. TD still covers only the technical issues and not the business issues, which is why to broaden the definition of TD and to introduce such a concept in the EA domain to overcome the shortcomings in EA, the term EA debt was introduced [Ale+20]. EA debt may be caused due to expedient architectural decisions that make future changes costly or even impossible [Ale+20]. They may also negatively impact the enterprise's business if they are not mitigated on time [Ale+20].

An outline for the EA debt management (EADM) framework was introduced by Alexander et al., in which the research objectives were suggested for all the nine activities included in the EADM, namely identification, collection, assessment, prioritization, monitoring, repayment, prevention, documentation, and communication [Ale+20]. All these activities would contribute toward managing EA debts in an enterprise [HJ20].

As the concept of EA debt is relatively young [HJ20], a workshop was done to identify the EA debts in which some hidden aspects, which contribute to EA debt, were identified. The few identified EA debts were: communication, contradictory goals of stakeholders, and lack of documentation [Jun+21]. As stated in the previous section, these issues are already identified in EA. Therefore, there is a need to have a process that could contribute to resolving these issues. Since the monitoring, documentation & communication activities in EADM could contribute toward solving the above-mentioned issues, it would be beneficial to research these management activities. This way, we could create a process that would make the reporting of EA debt easier.

2.3. Debt Management & Reporting Tools

As we have to develop a process to report EA debts that would contribute to monitoring, documentation & communication activities in EADM, we did some analysis to see if we can use the already present reporting techniques to not reinvent the cycle. Our goal is to have an automated way of loading the EA debts and generating a report out of them; we need a reporting tool that we can programmatically integrate with to create and generate reports. First, we analyzed tools used to analyze and identify TD. We explored a few tools for efficiently managing TD, which are:

SonarQube [Son] is a code quality and code security tool that helps identify technical issues from which TD is one of them by doing automated checks for detecting code smells, bugs, vulnerabilities, and code coverage [PH19]. It achieves this by checking the code against a predefined list of rules which can be modified. This tool

2. Background and Related Work

is limited to applying the rules on code; therefore, we cannot upload the list of EA debts as EA debts are not limited to technical perspectives.

Squore [Squ] is a software analytics and static code analysis tool. This allows it to find bugs & vulnerabilities and measure TD. Furthermore, it supports dashboard functionality, enabling the user to view all the found issues. Like SonarQube, we cannot upload custom debts, and the tool is again just limited to code.

Teamscale [Tea] is a source code quality analysis tool and supports various visualizations of the issues found in code. Its primary focus is to determine the maintainability constraints and avoid unexpected maintenance costs in the future [PH19]. Unlike the tools mentioned above, this tool can accept external findings. Still, these are limited to static analysis tools, which again focus only on code, or coverage tools, which are also focused on written tests for the code, making it not usable for EA debts.

Kiuwan [Kiu] focuses on code security and provides vulnerability insights into the code. It has functionalities for code analysis to determine the code quality constraints. Furthermore, it helps teams plan out how to fix the defects found, but it does not support the upload of external findings and is again limited to code, making it not usable for EA debts.

As mentioned before, these tools cannot contribute to the monitoring, documentation & communication activities. This is because none of the tools support the uploading/storing of the EA debts, which we would require to generate reports. Therefore, we started looking into more general reporting tools, specifically open-source reporting tools. For this, we have limited the search to open-source reporting tools that support Java because this is the preferred language for the project. This left us with a relatively concise list of tools to review. The evaluation criteria for the reporting tools include affordability, ease of making reports, support for various visualizations, and access to programmatically make changes to automate the report-making process and the ability to maintain it. The tools we analyzed are as follows:

JasperReports [Jas] is a feature-rich and widely used open-source reporting tool. It allows its users to make reports for various target formats. However, due to its comprehensive feature list, it is a tool that is very complex and hard to use. Due to the variety of stakeholders, this is not a viable tool because it would first require knowing about the tool and having some expertise to design the reports, which could make the reporting process time-consuming. We can automate the design of a report

by code; still, the process is a lot harder and more tedious as the design requires details like size, length, and grid position, which are easier to tell by designing the report in the tool, but as mentioned above that requires knowledge and expertise. Hence this reporting tool does not fulfill our criteria; even though it is free and has various supported visualizations, but the tool's complexity and lack of flexibility in programmatically automating it makes it not fulfill our evaluation criteria.

BIRT [Bir] is a reporting tool much like JasperReports. It allows users to make reports and data visualizations that can be integrated into Java web applications or exported into various formats. However, when we look at the evaluation criteria, it has the same shortcomings as Jasper reports. It is a tool that is hard to use, with a very steep learning curve. While we can programmatically create reports, much like JasperReports, it is extremely complex to use and hard to maintain. This makes it not pass our evaluation criteria.

Pentaho [Pen] is another reporting tool that allows users to create reports and output these in various formats. Compared to JasperReport and BIRT, it is easier to learn and less complex, but at the expense of more advanced features. This tool lacks the ability to create the desired visualizations easily; for example, this tool does not support displaying data in the form of tables; furthermore, programmatically using the tool to generate a report requires a paid subscription. Due to this, this tool also does not fulfill our evaluation criteria.

From the above analysis, we can conclude that we will need a separate framework for reporting EA debts that can work with EA debts data and provide a straightforward interface to allow various stakeholders of diverse skill levels to make reports.

3. Concept

Contents

3.1. Requirements	11
3.2. Design of Framework	13
3.2.1. Conceptual View	13
3.2.2. Process View	14

This thesis aims to develop a straightforward process to handle the life cycle of EA debts, the life cycle of EA debt reports, and the continuous distribution of EA debts. To attain this, we need to make a framework that must fulfill the requirements that a stakeholder might need. So, in this chapter, we will discuss the requirements, their purpose, and how we will validate the shortlisted requirements. Further, we will look at the design of the framework with the help of the conceptual and process view of the EA debt reporting framework.

3.1. Requirements

These requirements are attained with a series of discussions with the product owner, who in this thesis case is the supervisor, Peter Alexander, who is in contact with the customers and knows the difficulties being faced by the practitioners regarding the reporting of EA debts and knows the prioritized requirements that must be included in the implementation of EA debt reporting framework. The requirements were finalized after a series of negotiations with the product owner (supervisor).

The EA debt reporting framework needs a mechanism to integrate the existing debt registry contained in organizations with the framework. The organization where we are researching the applicability of EA debt management already has a debt registry maintained in the form of an excel document. So first, we need to have a mechanism to extract all the essential information and display it to the stakeholder. An additional benefit to this is creating relationships in the data by converting them into entities and storing them in a database. Storing the data in a database makes data access and management easier and effortless by using existing data access libraries, for example, Spring JPA [Sprb], Hibernate [Hib], etc. This will also help us in having a filtration mechanism for the data. Now, as I have been provided with one organizations debt registry, in the end, we need to check if

3. Concept

the other organizations will be able to provide a similar debt registry. Will they be able to provide data in the form of an excel sheet? And how useful is the provided filtration process going to be?

After having the EA debt data, the EA debt reporting framework needs to provide a clear overview of the EA debt data and provide concise insights into the data. Now the purpose of this overview is to have a visual display of all the EA debt data present and provide information at a glance. For this, we will also require knowing the lifecycle of EA debts to show insights into the situation of each EA debt present in the organization. Lastly, we will need to verify whether the insights in visual form are helpful. Are they clear and concise enough? Does it increase the total visibility regarding the EA debts in the organization?

The EA debt reporting framework needs to enable the generation of EA debt reports using user-defined templates. As the EA debt reporting framework's goal is to provide a process and mechanism to report EA debts, therefore, the framework must support the process of creation and generation of these EA debt reports. The entire process must be intuitive and easy for stakeholders from all levels to use and understand. This requirement has two sub-parts. The first is the creation of templates, and the second is the generation of reports using those templates. Creating templates will enable the making of EA debt reports based on viewpoints, whereas the generation of reports will deal with the EA debt data to be shown in the EA debt report. The benefit of this separation is that created templates can be reused in multiple reports without repeatedly redesigning the same view. We must then verify whether this entire process of creating templates and generating reports is intuitive and easy. Will it save time and help in making prompt reports? Will it help overcome the communication gap between stakeholders?

Lastly, the EA debt reporting framework needs to ensure that the EA debt reporting process is continuous and has a mechanism to keep track of changes that may happen in EA debt data which will be reflected in the EA debt reports. We are required to keep track of changes to let concerned stakeholders know the progress of EA debts in the organization. It can also help in making future decisions for the activities being carried out in the organization. As receiving notifications is a good way to stay informed, a notification mechanism may be used to convey the changes in the EA debt report. Lastly, we need to check if the impact of continuous EA debt reporting is positive. Is it easy to keep track of changes that happen in the report? Will the stakeholder that needs to be informed review the conveyed changes? Will notifications help in taking action to mitigate the EA debt?

3.2. Design of Framework

To organize the information as stated in the requirements and come up with the visual structure, we will have a look at the design of the framework with the help of a conceptual view and process view.

3.2.1. Conceptual View

A conceptual view presents the framework from a conceptual level that has a higher level of abstraction. We have come up with the building blocks that are required for the EA debt reporting framework, as shown in Figure 3.1; we now need to understand the link between these components and the requirements as stated in Section 3.1

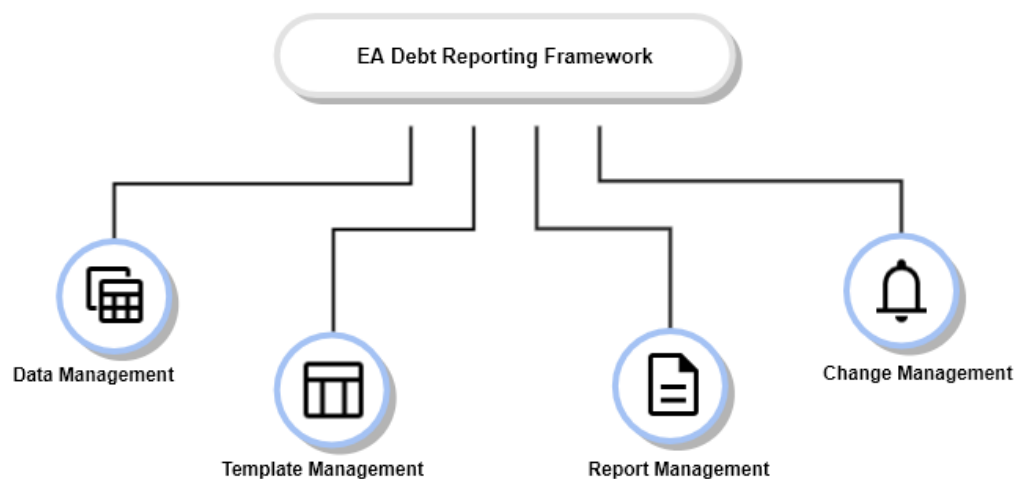


Figure 3.1.: Conceptual View

As stated in the requirements, we need a mechanism to integrate the existing debt registry with the EA debt reporting framework and also to have an overview of the imported EA debt data; for this, we need to have a data management component that, as the name states, manages the EA debt data, for example, reading data from the EA debt registry, securely storing, and retrieving it with the help of queries.

Next in the requirements, we stated the need to have the capability for creating templates and generating reports. As we already divided this requirement into two parts, template management deals with the actions related to making templates. Whereas, report management handles generating the reports using the templates made.

Finally, to keep the EA debt reporting process continuous, we need to have a change control process that identifies the changes that occur and communicate the changes in the form of reports. Therefore, the change management will deal with the change control process.

3.2.2. Process View

The process view explains the processes and how the communication happens. In the conceptual view, we saw the four building blocks required for the EA debt reporting framework. Now to understand the interactions between and to these components, which are referred to as building blocks, we need to see the activities a stakeholder can perform on the EA debt reporting framework.

For displaying the process view, we have used use case diagrams which help in identifying the interactions between the actors and the systems. It is the graphical depiction of the possible activities that an actor can perform with a system. This helps in designing the system from the end user's perspective.

The use case diagram is independent of the order in which the activities must be performed and shows the communication of an actor using the use case and the system. The actor is someone who plays a role in business and triggers use cases and expects output from the system. The use case is the functionality that the actor wants to perform. We have divided the use cases into four separate diagrams to understand the interaction with each component in the EA debt reporting framework.

The first use case diagram is for data management. The boundary is shown by the rectangular box, which tells the system, which in our case is the EA debt reporting framework. The sub boundaries denote the components present in the EA debt reporting framework. The 'includes'-relationship, depicted with a directed dotted line, shows the inclusion of the other use case into the base use case.

As shown in Figure 3.2, three primary use cases can be performed by the actor which in our case is the stakeholder, with the data management component of the EA debt reporting framework.

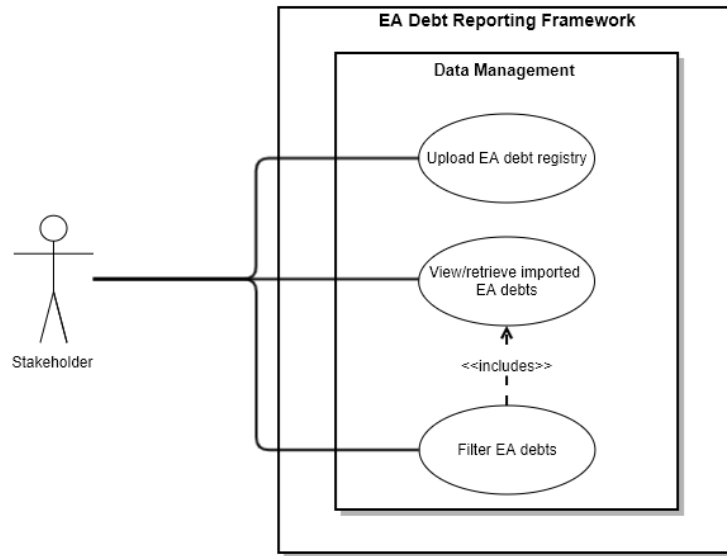


Figure 3.2.: Data Management Use Cases

The next use case diagram is for the Template Management component, where we can see the three main actions a stakeholder can perform on the Template Management component, shown in Figure 3.3, as part of the EA debt reporting framework.

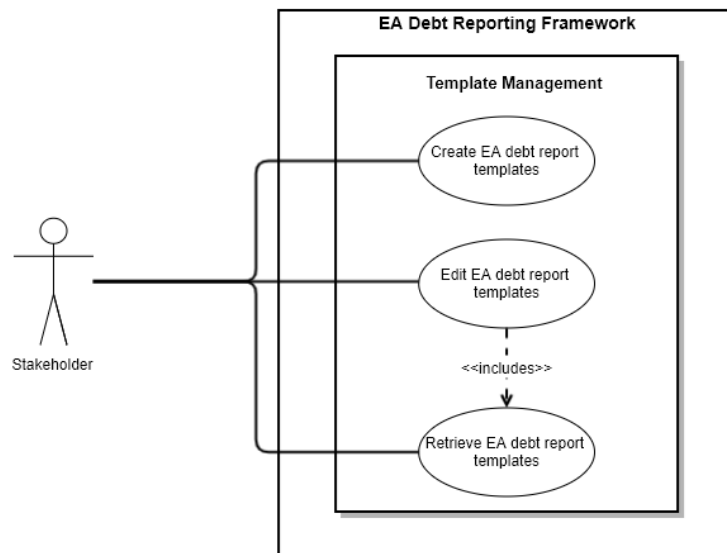


Figure 3.3.: Template Management Use Cases

3. Concept

In Figure 3.4, we can see the use cases for the Report Management component. This component interacts with the Template Management and Data Management components to fulfill the use cases.

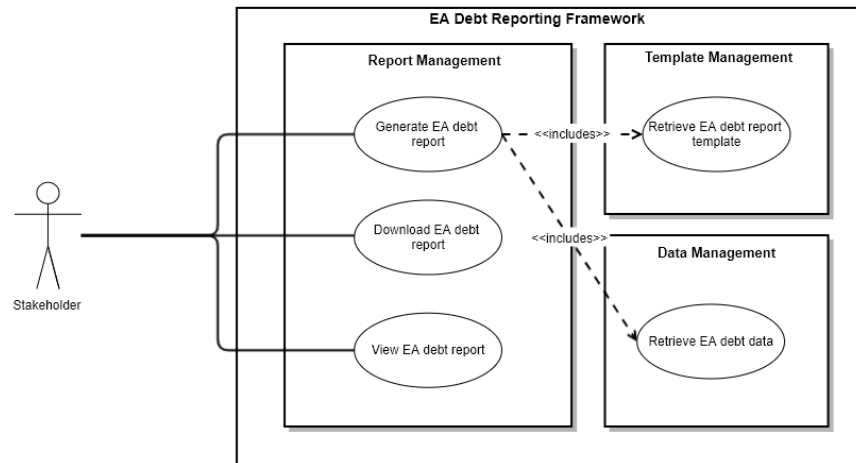


Figure 3.4.: Report Management Use Cases

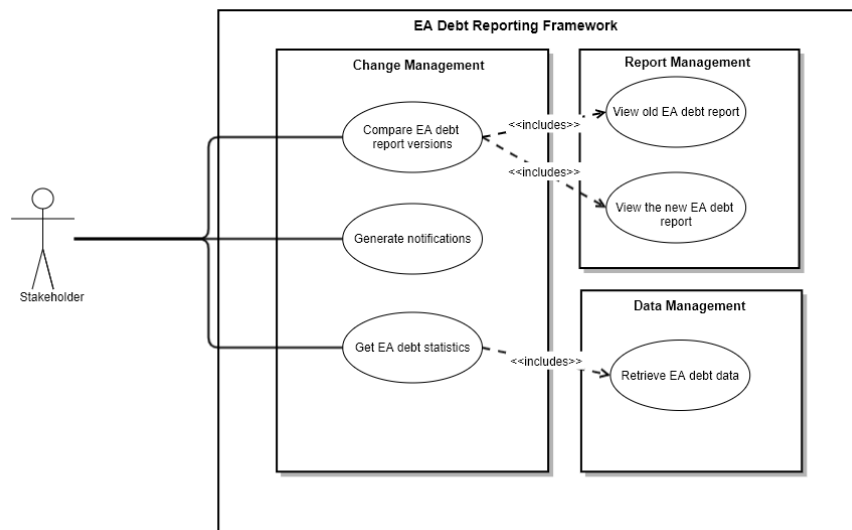


Figure 3.5.: Change Management Use Cases

Lastly, for handling changes in the EA debts, shown in Figure 3.5, the Change Management component is used. This component also interacts with other components to perform the use cases, which are Report Management and Data Management.

4. Realization

Contents

4.1. Architecture	17
4.1.1. EA Debt Reporting Framework Architecture	18
4.1.2. Technology Decisions	19
4.1.3. Data Model	20
4.2. EA Debt Reporting Framework Implementation	21
4.2.1. Data Management	22
4.2.2. Template Management	26
4.2.3. Report Management	29
4.2.4. Change Management	32
4.2.5. User Interface of the EA debt Reporting Framework Implemen- tation	33

After identifying the EA debt reporting framework requirements and design, we now need to work on realizing the EA debt reporting framework and its implementation. In this chapter, we will discuss the architecture and the technology decisions made, followed by the details regarding the implementation of the EA debt reporting framework. Lastly, we will also show the *User Interface (UI)* made for the EA debt reporting framework.

4.1. Architecture

Our objective is to make a prototype of a reporting framework for EA debts and an implementation using this framework; in the previous chapter, we discussed the requirements that we need to incorporate and discussed the high-level design, as shown in Figure 3.1, which will help in attaining and making the process for reporting EA debts. Next, these components need to be put into an EA debt reporting framework implementation. We need to see the interactions that will take place in the EA debt reporting framework implementation, and how will the EA debt reporting framework components work together to perform the functionality to cover all the technical and operational requirements; next, we will look at what the implementation architecture will look like.

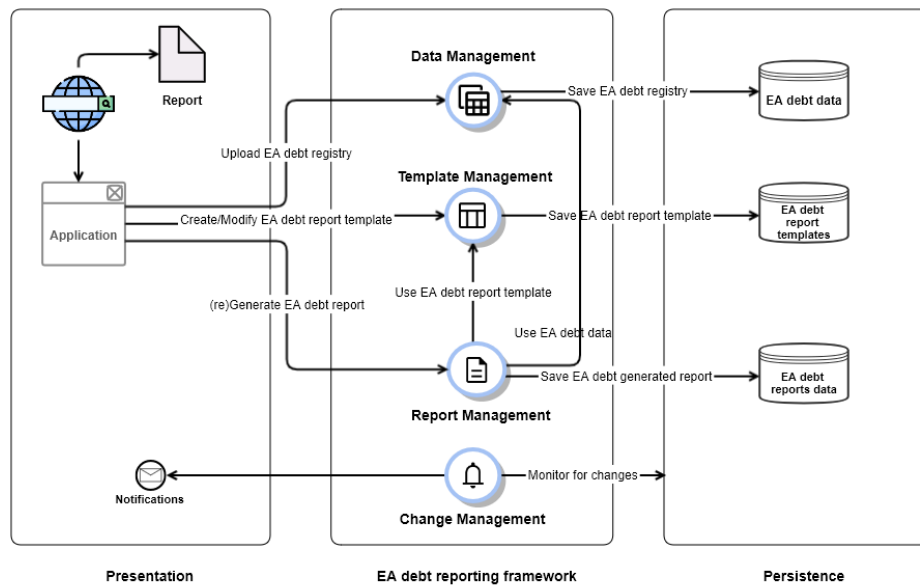


Figure 4.1.: EA debt reporting framework implementation architecture

4.1.1. EA Debt Reporting Framework Architecture

As shown in Figure 4.1, the EA debt reporting framework implementation can be structured in a three-layered architecture: a Presentation layer, an Application layer which is the EA debt reporting framework, and a Persistence layer.

Presentation Layer

The presentation layer handles the user interaction, for example, in the form of a desktop application or a website where a UI is displayed to the user and where they can perform actions. For the EA debt reporting framework implementation, we want a front-end application with which the stakeholders can interact, which will then interact with the application layer to execute the business logic.

Application Layer

The application layer is the main logic layer. This holds the business logic the application must support to perform the applications core functionalities. This can be in the form of a back-end application that can connect to the persistence layer.

Persistence Layer

The persistence layer, often called the data layer, is responsible for storing and retrieving the data. This could be in the form of a relational database, a non-relational

database, or a file store. We have decided on a relational database for the EA debt reporting framework implementation.

4.1.2. Technology Decisions

Figure 4.1 presents the three-layered architecture which displays the structure for the EA debt reporting framework; next, we need to decide on the technologies which we will use to realize the EA debt reporting framework.

The presentation layer will be implemented using an Angular 12 [Ang] application and template [Cre], which will communicate with the application layer by interacting with the back-end application. We chose Angular 12 [Ang] because it was the newest version of Angular, a highly used framework to develop front-ends. This will be the interaction point for the stakeholder.

The application layer will be implemented as a Java 17 [Jav] back-end application using the Spring Boot framework [Spr]. Spring Boot makes it easy to run a production-ready Spring-based application. Furthermore, it has many integrations to connect to external systems, like a data store.

Finally, for the persistence layer, we selected PostgreSQL(Postgres) because it is a free, stable, mature, and open-source relational database management system [Pos]. We picked a relational database because we will only be dealing with relational data.

Next, two more technology decisions must be made before starting with the EA debt reporting framework implementation. First, we must decide how the front-end and the back-end will communicate with each other. One of the main ways Spring Web, a sub-framework from the Spring framework, manages communication with other services is by using the *REST* architecture style. It accomplishes this by using *JSON* over *HTTP*.

The other decision we must take is how to generate reports without having to do extensive coding for making PDFs; we selected Browserless [Bro] as the solution. It is a web service that accesses web pages without showing them to the user. Its PDF *API* navigates to the site and captures it into a PDF, which helps us reuse the *HTML* and *CSS* used for the front-end application

To clarify, there will be a front-end application using the Angular 12 [Ang] framework with an Angular 12 template [Cre] and a back-end application built in Java 17 [Jav] using the Spring Boot framework [Spr], which will together be called the EA debt reporting framework implementation. Using the front-end, we can upload an EA debt registry, create/edit templates, (re)generate reports, and notify stakeholders of changes. Each task is carried out by a separate component and then persisted in the database. The technical working details are explained in the following sections.

4.1.3. Data Model

For this thesis, the product owner has provided us with an EA debt registry in the form of an excel document which contains two sheets, the Debt Registry and Debt Consequences. We first need to analyze the provided data and make a data model. The data model helps us visualize the physical database design in terms of entities, attributes, and relationships. As this provided excel sheet is of an organization, and they refer to their applications/system and business capabilities as gear applications and tita capabilities, we will also name it the same and use this terminology in the further sections and chapters.

We will use an *Entity Relationship Diagram (ERD)*, a visual representation used to explain the logical structure of the database, as shown in Figure 4.2. The rectangular boxes are the entities which are the tables that will be created in the database, and the attributes for the entity with its data type are written below. The attributes of the entities are further described in Appendix A.

As shown in the ERD, the debt item is the primary entity containing all the EA debt related information, which can be seen by the attributes listed in the debt item entity. As a project, gear application and tita capability can have multiple EA debts. Therefore, the debt item entity is linked to project, gear application, and tita capability entities. The consequence entity provides the consequences an EA debt will have on the gear applications and tita capabilities and describes the severity level of an EA debt consequence. Further, the report template, created report, notification, and filter store entities are made to deal with the EA debt reporting mechanism, which will be discussed in detail in the next section.

4.2. EA Debt Reporting Framework Implementation

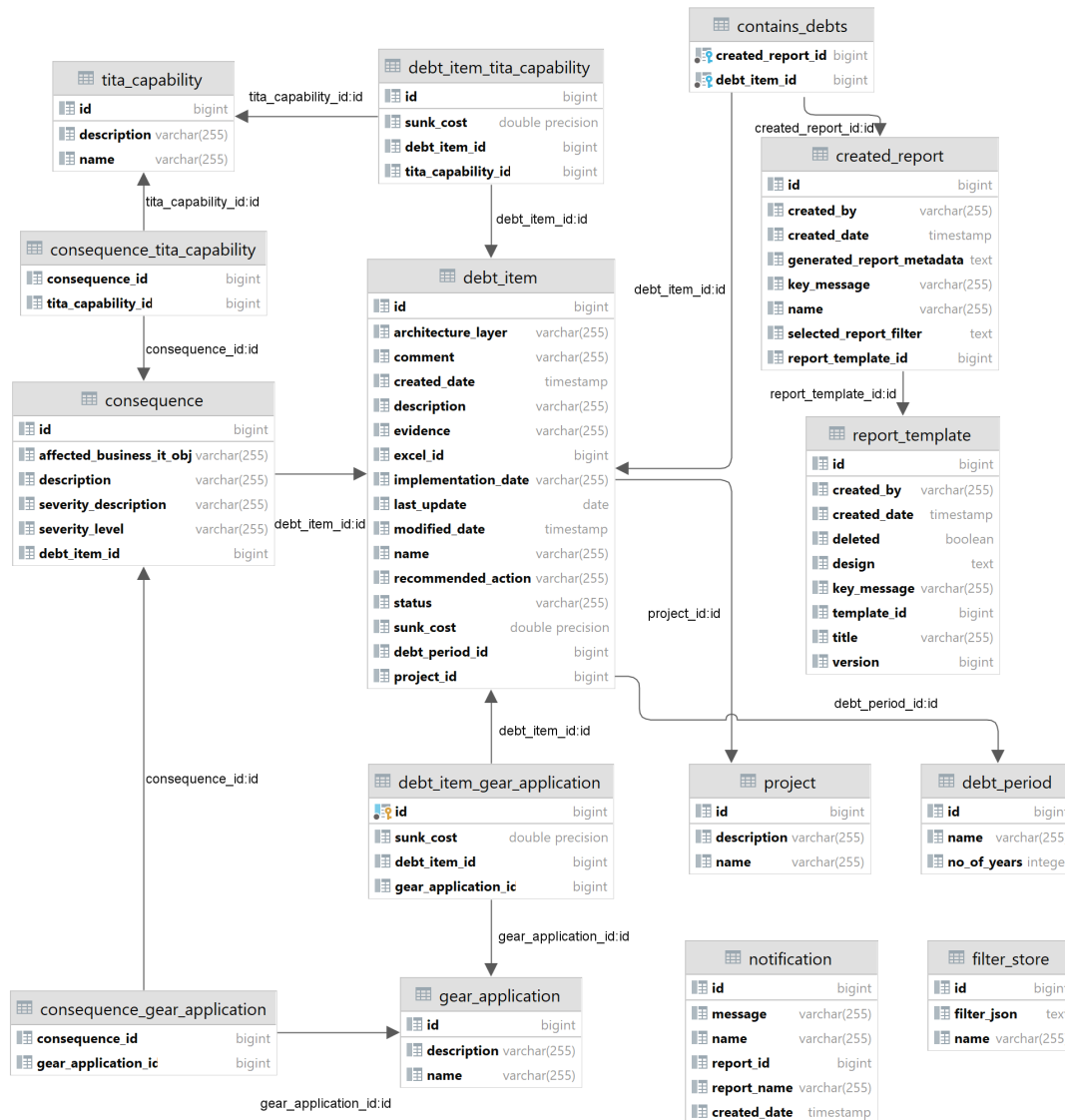


Figure 4.2.: Entity Relationship Diagram for EA debt reporting framework

4.2. EA Debt Reporting Framework Implementation

The EA Debt Reporting Framework Implementation is the collection of the front-end Angular 12 [Ang] application and the back-end Spring Boot [Spr] application. In this section, we will be explaining the implementation as a whole, and per topic explain what happens in the different parts.

The back-end application uses the Separation of Concerns principle to separate

the different layers so that every component has a single concern. As shown in Figure 4.3, the back-end application is divided into two Maven [Mav] modules, and further into Controllers, Services, and Repositories. The controllers expose *Data Transfer Objects (DTOs)* to the front-end and translate these objects using mappers, a class made for translating between DTOs and Entities, before accessing the services. The Repositories are in charge of storing and retrieving the data, in the form of Entities, in the data store. Finally, the Services bind the Controllers and Repositories together, providing the business logic needed to fulfill the requirements.

The Maven [Mav] modules allow the project to be built into two artifacts; the framework itself is one of the artifacts, which, if required, can be reused in another project. The second artifact is the entire back-end, which has been tailored to serve the information needs of the front-end. The second benefit of having the Maven modules is to ensure the visualization layer of the back-end does not get intertwined with the framework, further ensuring the reusability of the framework.



Figure 4.3.: Separation of concerns overview

Below we will discuss the four components as discussed in Figure 3.1. We'll go over each of these components to explain what it covers and how that has been implemented. After the four components, we will show and discuss the UI of the EA debt Reporting Framework Implementation.

4.2.1. Data Management

This component for the EA debt reporting framework supports several front-end functionalities, which are: Data Entry, Debt Registry, and Dashboard. Below we will discuss how it is implemented in the solution for the EA debt reporting framework implementation per front-end functionality.

Data Entry

The data entry component deals with uploading the provided EA debt registry into the database. In Section 4.1.3, we mentioned that the product owner has provided us with an EA debt registry in the form of an excel document. In excel it is challenging to make relations between the data and filter the data, making it harder to use as a datastore. To fix this, we want to convert the data from the excel document into entities that we can store. The front-end is in charge of providing a place for the user to upload the excel file and send this to the back-end. In the back-end,

we receive an excel document, which we need to convert. For this, we have implemented the Apache POI library, an open-source library that provides a Java API for reading and writing Microsoft Excel documents. Using this library, we read both sheets of the provided excel document and, per row, process the EA debts and EA debt consequences. Each cell on a row is a predefined input, that is, the title of an EA debt can always be found in the second cell of a row and the description in the third, and so on. We will use this to our advantage to map the excel input into a temporary model that looks similar to the input to more easily deal with changes in the provided excel debt registry, which then can be mapped to the entities which can be stored using the repositories.

In Figure 4.4, we can see the class diagram for the implementation of importing the excel document containing the EA debt registry. In this diagram, we can see two packages: service and adapter. The adapter package contains the logic required to translate the provided EA debt registry to the entities of the EA debt reporting framework implementation. The service package provides functionality to retrieve, save, and update the entities within the EA debt reporting framework implementation. The adapter will use this service package to fulfill its importing task. Other adapters can be added to the adapter package to support other formats of EA debt registries.

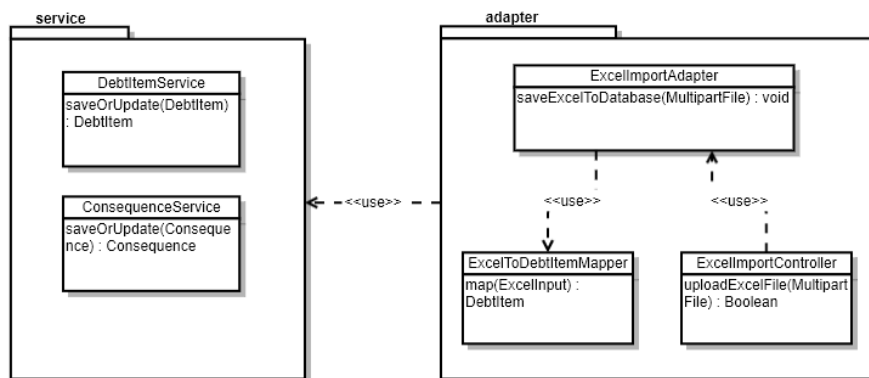


Figure 4.4.: Adapter Implementation Class Diagram

Debt Registry

The debt registry consists of two segments, the debt registry table, which is the view for all the EA debts, and the filter functionality. Both will be explained below.

- **Debt Registry Table:** The debt registry table displays the EA debts imported from the provided EA debt registry. A stakeholder can select the desired columns from a dialog, a window that opens over the content of the page and that needs user interaction to perform an action. When the stakeholder applies the selection, the debt registry table will be altered to show the selected columns. The table also offers pagination and the ability to sort on any of the columns, allowing the stakeholders to modify the view they get to their convenience.
- **Filter Functionality:** The EA debts shown in the debt registry table can be filtered using the filter functionality. The stakeholder can filter based on four columns. The product owner has selected these columns due to their frequent usage in the organization: Project, Gear Application, Tita Capability, and Status. Here, Gear Application corresponds to applications/system and Tita Capability corresponds to business capabilities. Each of these columns is a multi-select drop-down input in which the stakeholder can select the desired items. A stakeholder can then apply the filter that will reload the information in the debt registry table with the corresponding debts that match the filter. Alternatively, they can save the filter with a name. A saved filter can be loaded, deleted, or set as a global filter. Unlike a non-global filter, a global filter is saved in the stakeholder's browser under local session storage. This allows the page to be reloaded or, at a later moment, reopened with the global filter still applied.

Dashboard

The dashboard consists of three segments, the treemap views implemented using the Highcharts [Hig] library, the amount of EA debts in a certain status, and EA debt statistics which show the state of each EA debt. We have used two terminologies here, status and state. The status tells in what stage an EA debt is, for example, open, confirmed, etc. Whereas, the state defines how an EA debt is doing, for example, on track, overdue, etc.

- **Treemaps:** A treemap visually displays the hierarchy of data based on a numerical value. The treemaps show the summed sunk costs, the costs incurred due to the presence of an EA debt, per project, gear application, or tita capability. The visualization considers the total sunk cost and divides the space according to the individual sunk cost per project, gear application, or tita capability. As shown in Figure 4.5, we have a project A and a project B; the sunk cost for project A is twice as big as project B; therefore, the resulting treemap shows project A twice as big. This allows the stakeholder to see the highly impacted areas more quickly.



Figure 4.5.: Treemap Example

- EA debts per status:** As shown in the state diagram, which shows a sequence of events an object can go through, in figure 4.6, an EA debt can be in five different statuses: Open, Confirmed, Planned for mitigation, Mitigated, and Discarded. On the dashboard, we show how many EA debts we have in each status to provide a summarized view of the EA debts in the organization.

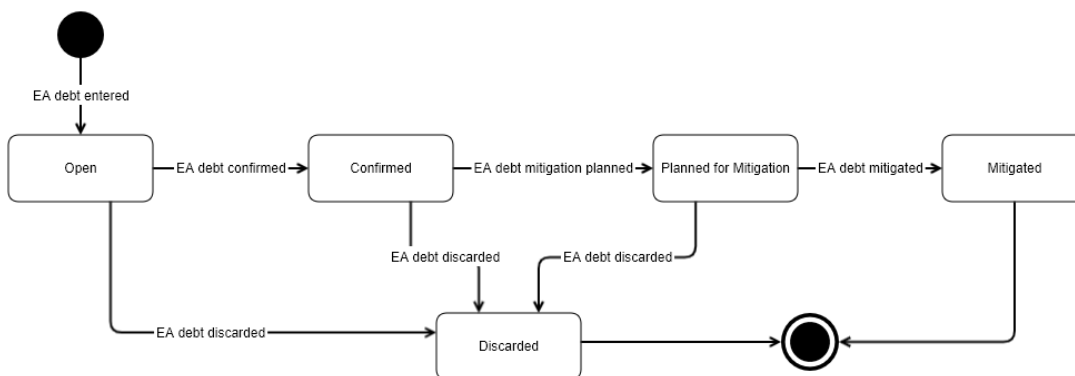


Figure 4.6.: State Diagram for EA debt Statuses

- EA debt statistics:** As shown in Figure 4.7, there are three states an EA debt can be in: on track, critical, and overdue. The state is defined by how far along the period and in which status an EA debt is. An EA debt may stay in the non-final statuses: Open, Confirmed, and Planned for mitigation, for a third of the total time each. In Figure 4.8, we can see the timeline for an EA debt, which starts with the creation of an EA debt and the expected end date, based on the debt period and which status an EA debt is in. For example, when an EA debt status is Open and the debt period is short-term (two years), then the time the EA debt is allowed to be in this status is one-third of the two years, approximately 35 weeks, and thus the expected end date for this EA debt is the

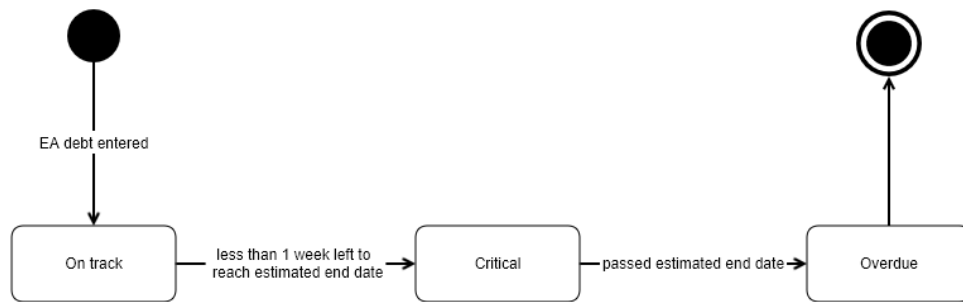


Figure 4.7.: State Diagram for EA debt States



Figure 4.8.: EA Debt State Timeline

created date plus the 35 weeks. When it has passed the expected end date, it is marked overdue; if it is in the week before the expected end date, it is marked critical, indicating the need for immediate action; otherwise, it is on track.

4.2.2. Template Management

As the name suggests, Template Management deals with the creation, editing, and deletion of templates.

Create Template

The creation of a template is a three-step process. The first step involves providing details regarding the template like the name for the template, key message for the template, created by and created date. The details provided in the first step are then reflected on the iframe as we proceed to the second step. In the second step, we select the widgets, which can be a column chart or table, which are the data visualizations supported by the front-end application for making the EA debt reports. We then have to provide data regarding the selected widget; for the column chart, we need to give the name of the chart that has to be displayed, the description for the chart that is to be made, additional explanation about the chart if required, the x-axis value, the y-axis value which provides us the possibility of using aggregation

functions for grouping the data based on the type of y-axis value selected. If it is a string type, then the COUNT aggregation function will be applied, and if the y-axis value is numerical-based, then we can apply SUM, AVG, or COUNT. The column chart is supported using the Highcharts [Hig] library. For the table view, we can select the data columns that are to be shown in the generated report. After providing widget details, we can move to the third step, and the iframe is updated with the information from the second step. In the third step, we can see a summary of the values entered in the first step and then can choose to save the template or go back to edit the data entered.

When we save the template, the configuration is stored in the database. The widgets are stored as a JSON string using the FasterXML Jackson [Jac] library. The reason for this is since there can be many widgets, and dealing with inheritance in a database either results in many small tables, which all have to be checked for entries, one big table with many empty fields, or one table for the base widget entity and another table for id-key-value combinations [Han]. The benefit of storing the collection of widgets as JSON means we can retrieve the configuration as one table read; this is much faster and easier.

We used several terms while telling about the template creation process. We will now clarify what these terms mean.

- **Widgets:** Widgets are the extensible graphical representations supported in the EA debt reporting framework implementation. We have implemented two widgets, namely the column chart and the table. For the column-chart, we have used the Highcharts library [Hig] on the front-end, which displays the data as vertical bars and represents data as columns according to the aggregation function selected. The vertical column view can then help in comparing values based on the data that is present on the x-axis [Col]. The Highcharts library supports many graphical visualizations, which can be added relatively easily in the future. The steps to adding more widgets can be found in the README.md [Fro], [Bac] files in the EA debt reporting framework implementation.

The table displays the information in grid format containing rows and columns. The columns represent the values of an attribute in all the entities. At the same time, rows describe the information regarding a single entity.

- **Iframe:** The iframe is an HTML element used to show another HTML page within the current document. We provide the iframe to help the user visualize the template design.
- **Widget Configuration:** Widget Configuration defines the structure in which the information regarding the template must be stored. We have defined the configuration for the widgets so that their data can be retrieved as a single object, as one template can contain one or more widgets.

4. Realization

```
1  [
2      {
3          "@type": "Column chart",
4          "chartDescription": "Demo Column Chart description",
5          "chartExplanation": "<font face=\\"Arial\\">Demo Column Chart
6              Message</font>",
7          "chartTitle": "Demo Column Chart",
8          "xaxisLabel": "Tita Capability",
9          "xaxisSelectedColumn": "Tita capability",
10         "yaxisLabel": "Sunk Cost",
11         "yaxisSelectedAggregation": "AVG",
12         "yaxisSelectedColumn": "Sunk cost"
13     }
14 ]
```

Source Code 4.1: Configuration for Column Chart Widget

```
1  [
2      {
3          "@type": "Table",
4          "selectedColumns": [
5              "Debt period",
6              "Sunk cost",
7              "Tita capability"
8          ]
9          "tableDescription": "Demo Table description",
10         "tableExplanation": "Demo Table Explanation",
11         "tableTitle": "Demo Table",
12     }
13 ]
```

Source Code 4.2: Configuration for Table Widget

```
1  [
2      {
3          "@type": "Table",
4          "selectedColumns": [
5              "Debt period",
6              "Sunk cost",
7              "Tita capability"
8          ]
9          "tableDescription": "Demo Table description",
10         "tableExplanation": "Demo Table Explanation",
11         "tableTitle": "Demo Table",
12     },
13     {
14         "@type": "Column chart",
15         "chartDescription": "Demo Column Chart description",
16         "chartExplanation": "<font face=\\"Arial\\">Demo Column Chart
17             Message</font>",
```

```
17     "chartTitle": "Demo Column Chart",
18     "xaxisLabel": "Tita Capability",
19     "xaxisSelectedColumn": "Tita capability",
20     "yaxisLabel": "Sunk Cost",
21     "yaxisSelectedAggregation": "AVG",
22     "yaxisSelectedColumn": "Sunk cost"
23   }
24 ]
```

Source Code 4.3: Configuration containing Table and Column Chart Widget

As shown in Source code 4.1, Source code 4.2, and Source code 4.3 the data entered for the widgets is stored as a JSON object. This object contains a "@type" field which identifies the widget type used.

Edit Template

After a stakeholder creates a template, it is shown in the template table. As we do not want to change existing templates because reports might still be linked to it, it is duplicated when the stakeholder clicks the edit button, and the version increases. This helps in the versioning of the templates.

Delete Template

If a template becomes obsolete, a stakeholder can delete it from the template table by clicking the delete icon. Since it can still be used in a report, we do not delete it but mark it as deleted, which means the front-end will not show it anymore.

4.2.3. Report Management

The Report Management is split up into several different front-end parts. We are going to explain each of these front-end parts below.

Generate Report

The first part of Report Management is the generation of the reports. To generate a new report, the user is expected to fill in an input form on the front-end to provide the required data. This consists of a report title, a report description, the creator's name, which report template to use, and the data that the user wants to view in the report by using a filter as made in the debt registry component. All this information is then sent to the back-end, where it is used to generate a metadata JSON which will be explained below, and then persisted into the database. The metadata JSON file is then sent to the front-end, which will be used to visualize the report.

- **Metadata:** The metadata is a JSON file containing the data required to visualize a report, as shown in Source code 4.4. This consists of a title, a description,

4. Realization

by whom it's created, when it is created, and the widget information (as explained in Section 4.2.2, Template Management). Each of the widgets in the metadata file contains all the information it needs to be loaded, including the user-provided description, the explanation, the labels, and the data that must be displayed. This means that when metadata is generated, it can be viewed and does not need to be regenerated for the same view.

```
1 |
2 | {
3 |   "title": "Demo Report Title",
4 |   "description": "Demo Report Message",
5 |   "createdBy": "Sana Kanji",
6 |   "createdOn": "2022-06-19T14:26:18.6126635",
7 |   "widgetDto": [
8 |     {
9 |       "@type": "Table",
10 |      "description": "Demo Table description",
11 |      "title": "Demo Table",
12 |      "explanation": "Demo Table Explanation",
13 |      "columnNames": [
14 |        "Debt period",
15 |        "Sunk cost",
16 |        "Tita capability"
17 |      ],
18 |      "columnValues": [
19 |        {
20 |          "period": {
21 |            "id": 511,
22 |            "name": "Medium-term",
23 |            "noOfYears": 5
24 |          },
25 |          "sunkCost": 0.0,
26 |          "titaCapabilities": [
27 |            {
28 |              "id": 516,
29 |              "name": "business capability B",
30 |              "description": null
31 |            },
32 |            {
33 |              "id": 515,
34 |              "name": "business capability A",
35 |              "description": null
36 |            }
37 |          ]
38 |        },
39 |        {
40 |          "period": {
41 |            "id": 511,
42 |            "name": "Medium-term",
43 |            "noOfYears": 5
```

```

44         },
45         "sunkCost": 0.5,
46         "titaCapabilities": [
47             {
48                 "id": 516,
49                 "name": "business capability B",
50                 "description": null
51             },
52             {
53                 "id": 515,
54                 "name": "business capability A",
55                 "description": null
56             }
57         ]
58     }
59 ]
60 },
61 {
62     "@type": "Column chart",
63     "description": "Demo Column Chart description",
64     "title": "Demo Column Chart title",
65     "explanation": "<font face='Arial'>Demo Column Chart
66         explanation</font>",
67     "xaxisLabel": "Tita Capability",
68     "xaxisValues": [
69         "[business capability B, business capability A]"
70     ],
71     "yaxisLabel": "Sunk Cost",
72     "yaxisValues": [
73         [
74             "business capability B",
75             "business capability A"
76         ],
77         0.25
78     ]
79 ]
80 }
81 ]
82 }

```

Source Code 4.4: Report Metadata

Preview

To get to a generated report, there is a table view, as shown in Figure 4.27, which lists all the generated reports. We can view and download the report and edit the key message in this table. For viewing the report, when the stakeholder clicks on the preview icon shown in Figure 4.27, the user is taken to the report that opens up

in a new tab on the browser.

The report is made by loading the metadata file into an HTML template; this is not to be confused with the report template, which holds the configuration for a report, but rather a template that has the layout of the visual aspect of the report. This design considers the different fields and options in the metadata file and will define how the actual report will appear. As such, the HTML template holds the logic to visualize the report. The resulting report can then be displayed to the stakeholder. Another benefit of this approach is that we can load the report into an iframe, as we do in the template creation, to show a rough image of how the report will look like when they are creating the template.

Edit

We have limited the editing feature for a generated report to just changing the key message. We have chosen to do this because once a report has been generated, it should not be changed by anyone and should stay the same. If we want to regenerate the same report with the same filter and template, then that can be done in the notifications tab, which we will discuss later in the Change Management section.

Download

The stakeholder should be able to download the generated EA debt reports, and this functionality is implemented by using the Browserless web service. We deployed the Browserless web service in a docker container and exposed the Browserless API to the back-end. The back-end then uses the API to generate PDFs containing the reports. It does this by sending Browserless the front-end URL along with the desired report's id. Then the front-end loads the report, and Browserless creates a PDF file. This file is then downloaded by the back-end, which serves it to the front-end, where the stakeholder can download it.

4.2.4. Change Management

This EA debt reporting framework component serves as the notification mechanism for the front-end application, which is discussed below.

Notifications

The notifications page in the front-end holds a table with all the generated reports. Per report, there are three buttons: Old, New, and Notify. When stakeholders click the Old button, they are taken to the report as stored in the database. This works like the preview functionality described in the Report Management Section 4.2.3. When the stakeholder clicks the New button, they are also taken to the report view. However, this is of a newly generated report. The stakeholder can compare the reports and decide if the changes are enough to notify other stakeholders. When

stakeholders click the Notify button, they are presented with a dialog, as shown in Figure 4.34. Here the stakeholder is expected to type the name of the recipients and the message they will see. When they click Notify in the dialog, a new report is generated and persisted in the database.

On the right top of the front-end, as seen in Figure 4.35, are two buttons: Dashboard and Notification. The Dashboard button will take the stakeholder to the dashboard view. The Notification button will have a list of notifications. The button also shows a badge, a small red circle, with the number of active notifications. When clicked, shows a list of notifications. Each of these can be clicked and will take the stakeholder to the notifications page.

4.2.5. User Interface of the EA debt Reporting Framework Implementation

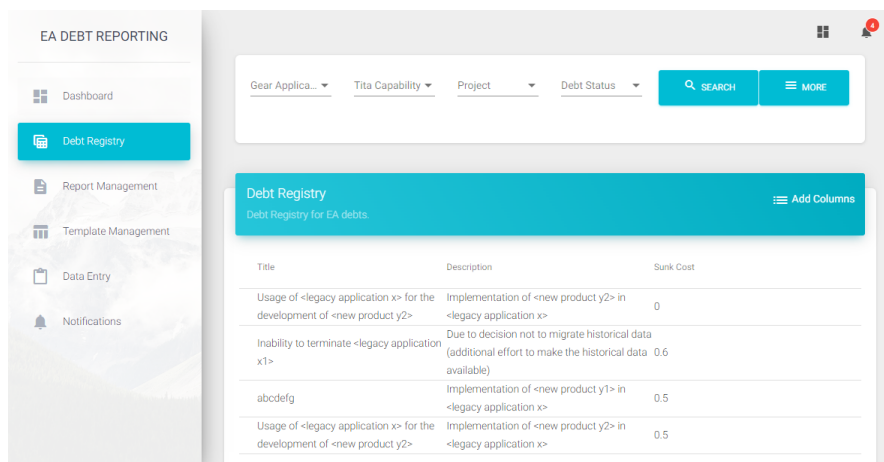


Figure 4.9.: EA debt Reporting Framework Front-end Application

As seen in Figure 4.9, the front-end is divided into two main sections, the sidebar and the content view. The sidebar enables the stakeholder to navigate within the application. The content view holds the views the stakeholder will go through and will be explained below.

Data Entry

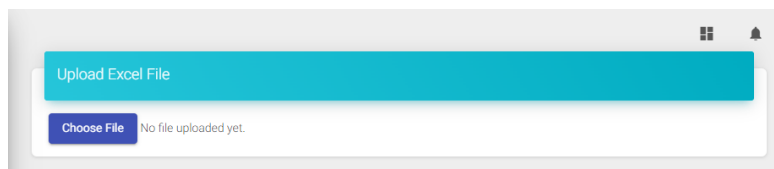


Figure 4.10.: Data Entry View to Upload an Excel Sheet

4. Realization

In Figure 4.10 we see the Data Entry view, here the stakeholder can upload the organization's debt registry in the form of an excel sheet.

Debt Registry

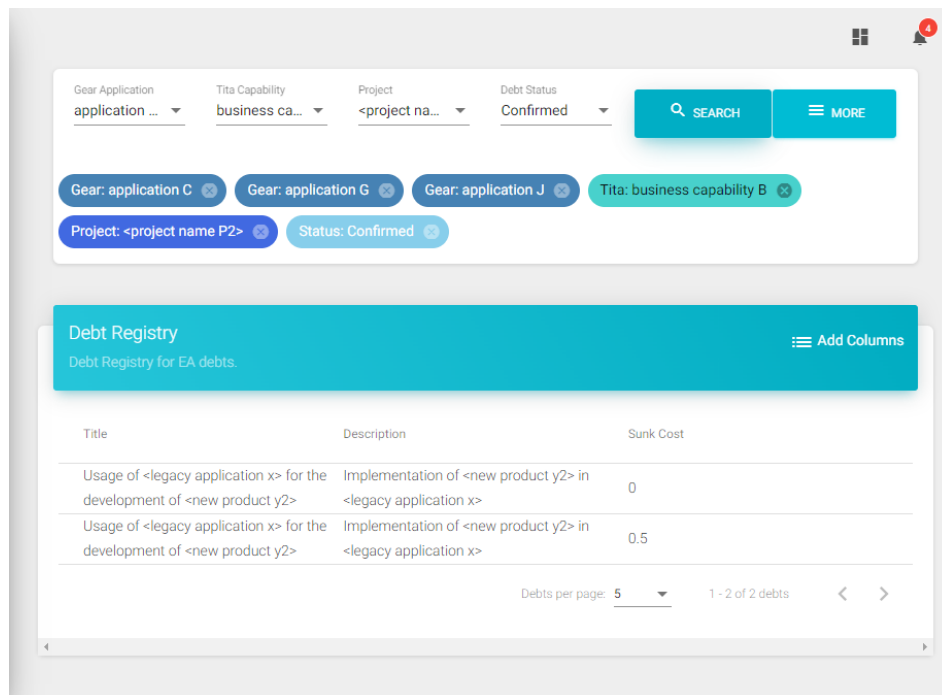


Figure 4.11.: Debt Registry View

Figure 4.11 Shows the Debt Registry view. Here a stakeholder can see all the EA debts that have been uploaded using the Data Entry View. In this view the stakeholder can add filters, through the use of the drop-downs. The active filters will be added as chips, the blue and green colored boxes with the rounded corners. The chips coming from the same column will have the same color.

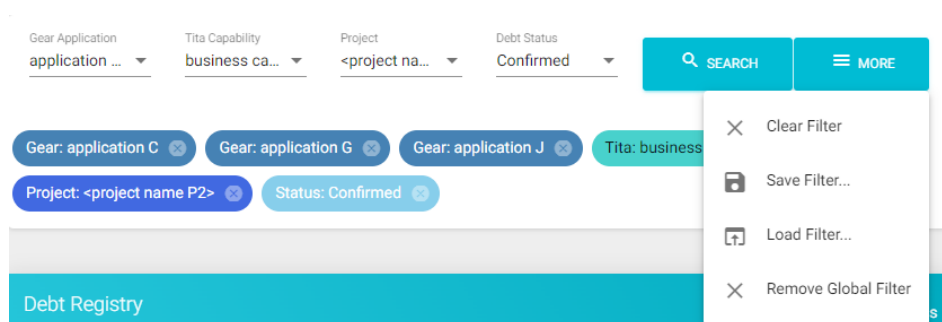


Figure 4.12.: Debt Registry More Options

Figure 4.12 shows the options a stakeholder can take on the applied filter, which are: Clear Filter, Save Filter, Load Filter, and Remove Global Filter. The clear filter removes the applied filter, the save filter will open the Save filter dialog as shown in Figure 4.14, load filter will display the load filter dialog as shown in Figure 4.15, and remove global filter will remove the global filter if applied.

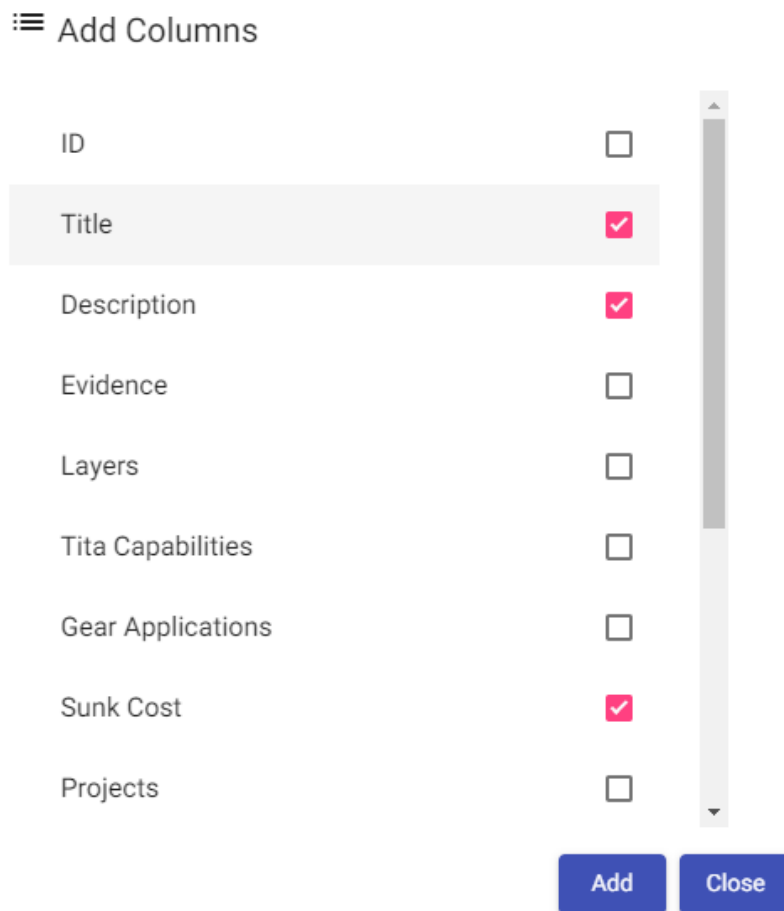


Figure 4.13.: Debt Registry Table Add Columns

The dialog as shown in Figure 4.13 enables users to add and remove columns in the debt registry table.

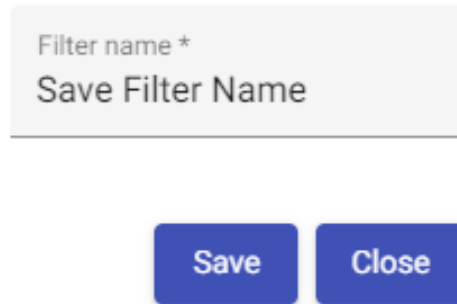


Figure 4.14.: Debt Registry Saving the Filter

The save dialog, shown in Figure 4.14, lets the stakeholder save the filter selections and assign the filter a name which can then be applied without repeatedly selecting the filters.

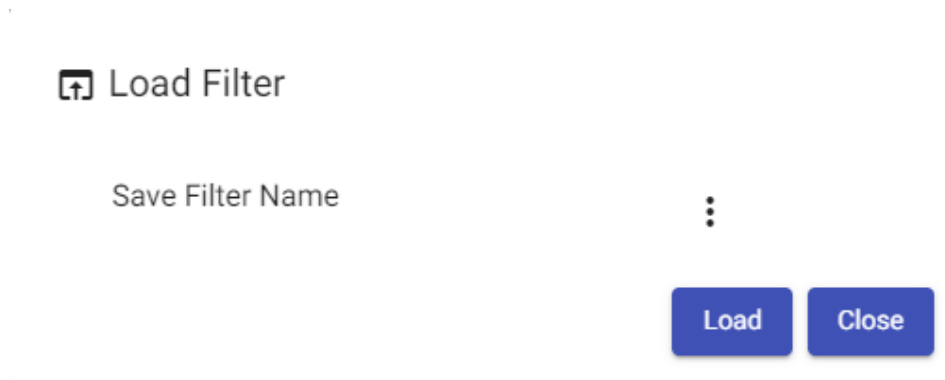


Figure 4.15.: Debt Registry Loading the Filter

Figure 4.15 shows a dialog where we can see the names of all the saved filters and have the ability to select and load them.

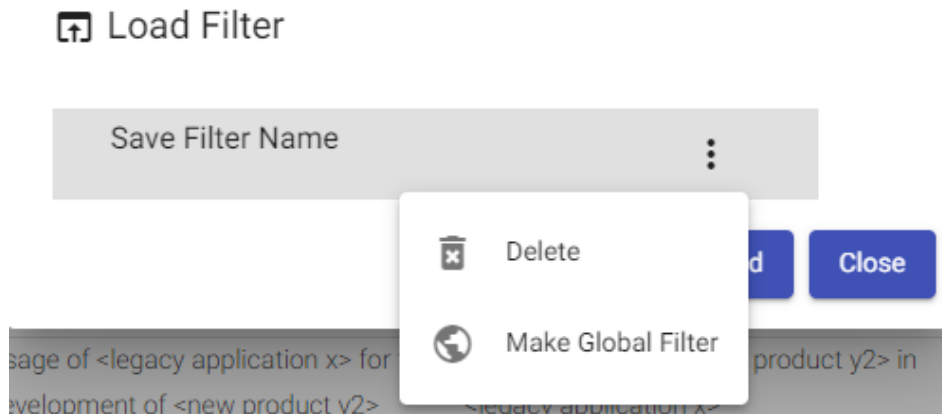


Figure 4.16.: Debt Registry Loading Filter Options

Each filter in the Load filter dialog also has a small menu, as shown in Figure 4.16, and allows the stakeholder to remove a filter, or apply it as a global filter.

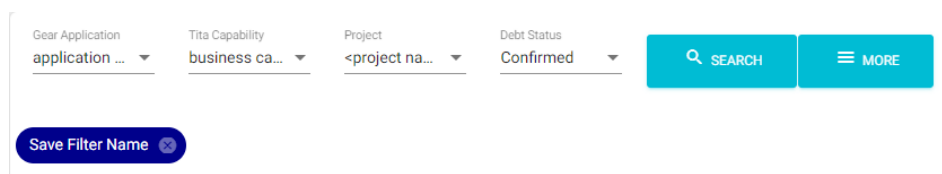


Figure 4.17.: Debt Registry Global Filter

After selecting the option 'make global filter' in the load dialog options, the global filter is applied, and to separate the view from other applied filters, the global filter is displayed by showing the name given to the filter, as shown in Figure 4.17

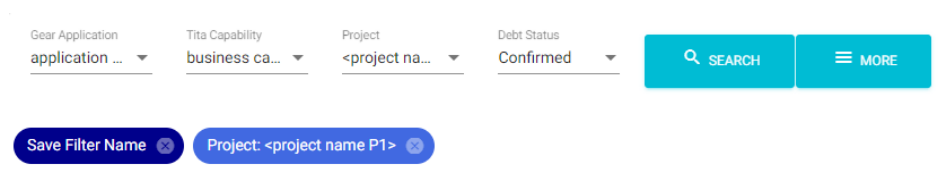


Figure 4.18.: Debt Registry Global Filter with Additional Filter

A stakeholder, with an applied global filter, can still add more filters. As Figure 4.18 shows, the global filter chip will remain and an additional chip will be added. If, however, the stakeholder removes a filter that is present in the global filter, then the global filter chip is removed and replaced with the actual chips of the applied filters.

Dashboard

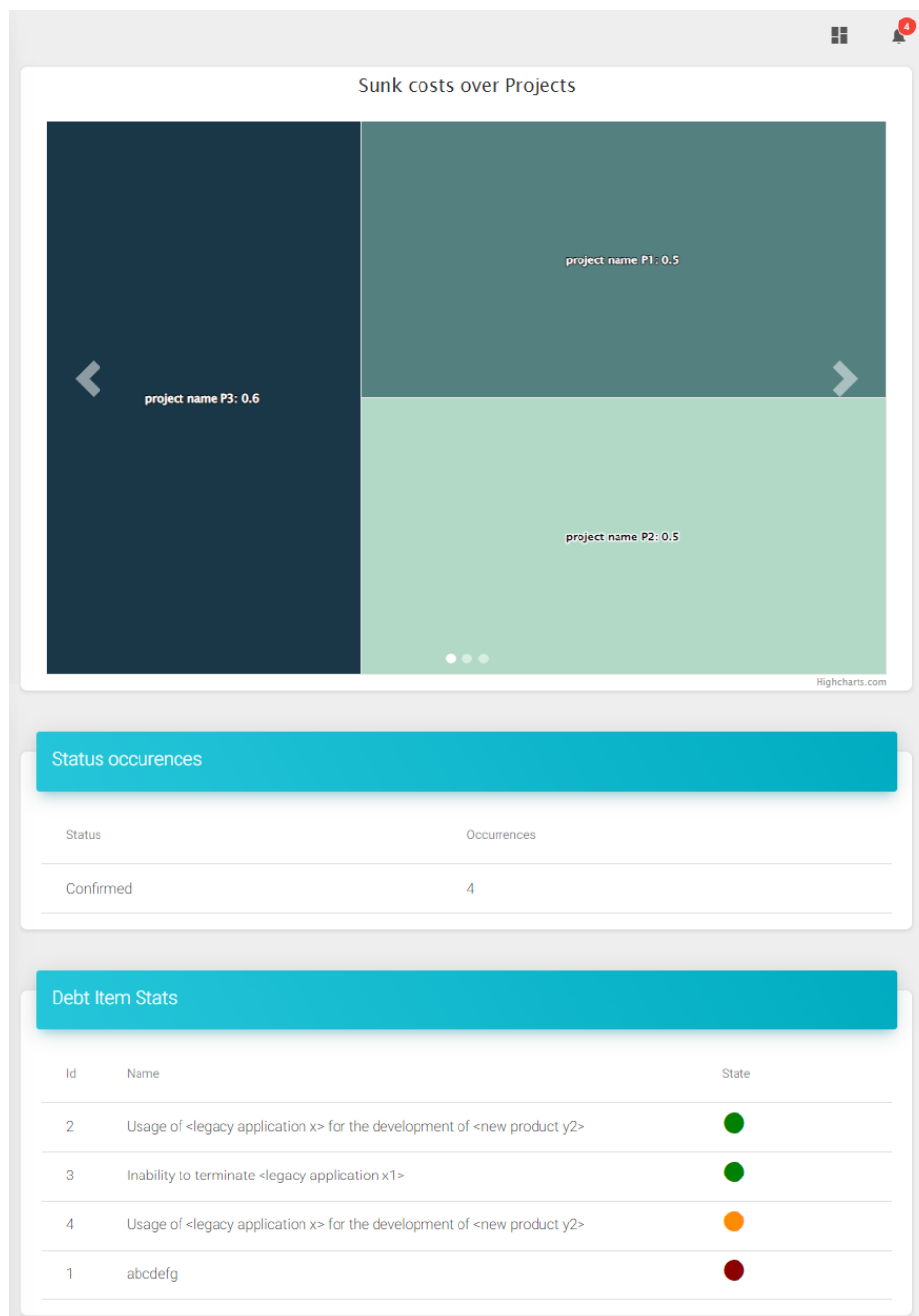


Figure 4.19.: Dashboard View

As described in Section 4.2.1, the dashboard is divided into three segments, as shown in Figure 4.19. The first segment is the treemaps which are in a carousel.

4.2. EA Debt Reporting Framework Implementation

A carousel is used for cycling through elements, in this case, three treemaps, and shows where the carousel is in its cycle, represented by three circles at the bottom.

Below is a table that shows the EA debt statuses and how often they occur. Followed by the Debt Item Stats, which shows the state of each EA debt as explained in Section 4.2.1. When an EA debt item in the Stats table is clicked, the debt item details are shown, as in Figure 4.20.

ID:	2
Title:	Usage of <legacy application x> for the development of <new product y2>
Description:	Implementation of <new product y2> in <legacy application x>
Evidence:	The <architecture standard z> <standard ID> is not respected here
Layers:	Application Architecture (AA)
Tita Capabilities:	business capability B, business capability A
Gear Applications:	application C
Sunk Cost:	0.5
Projects:	<project name P2>
Period:	Medium-term
Implementation Date:	4th quarter of 2020
Recommended Actions:	Define business requirements and rules for <strategy S>
Status:	Confirmed
Comments:	Reason for debt: Reducing the project risk of going overbudget and behind the schedule. [Assumption] investment I is planned to mitigate this debt item

Figure 4.20.: Debt Item Details

Figure 4.20 shows the details of a specific EA debt item. Here all the fields of an EA debt item are listed.

Template Management

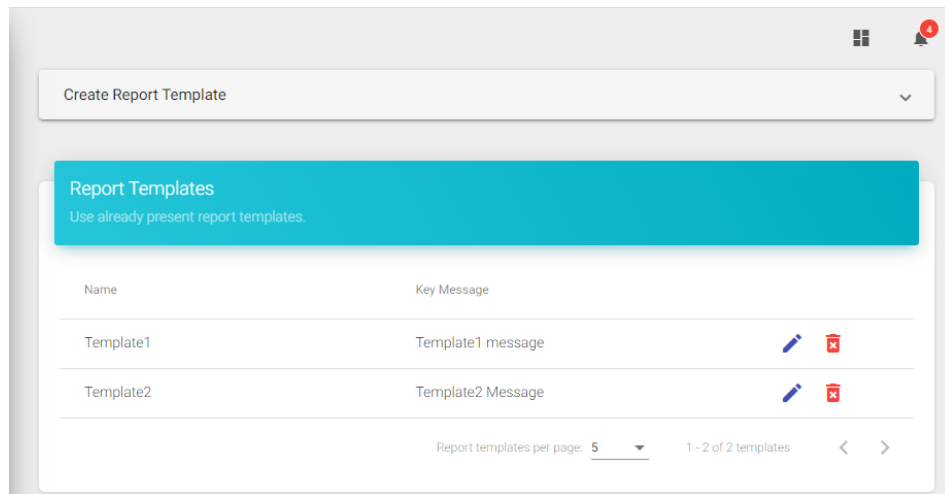


Figure 4.21.: Template Management View

Figure 4.21 shows the Template Management page, which is divided into two segments. The upper segment is for the creation of new report templates. The below segment is for displaying the list of created report templates.

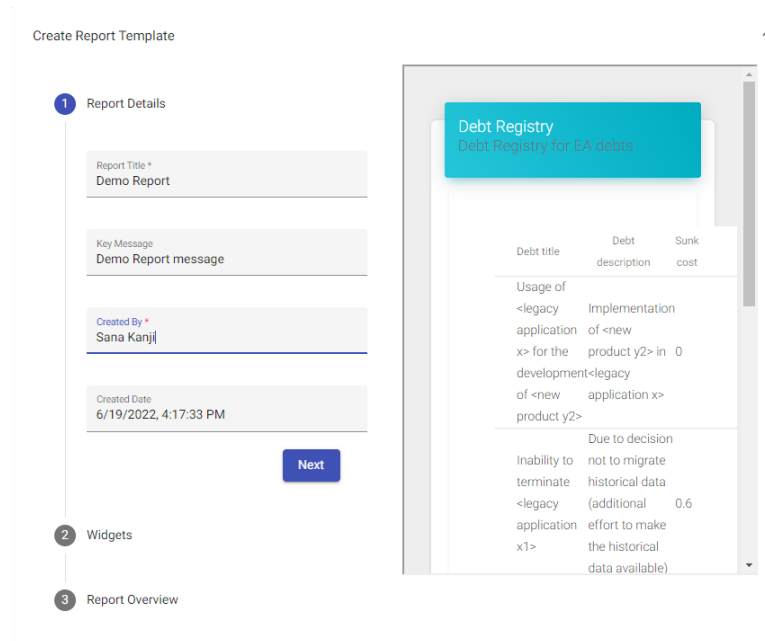


Figure 4.22.: Template Creation First Step

The creation of the report template is a three-step process. Figure 4.22 shows the first step in which the stakeholder can enter details in the fields which are explained in Section 4.2.2.

4.2. EA Debt Reporting Framework Implementation

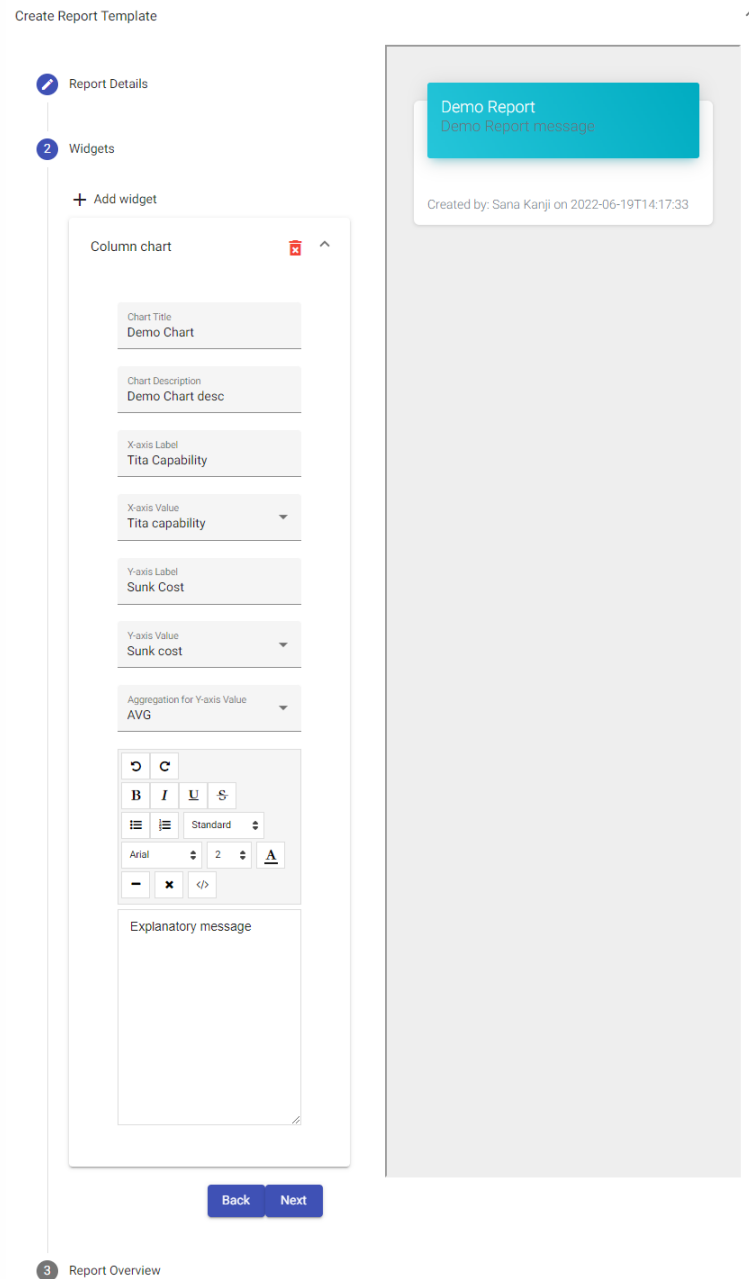


Figure 4.23.: Template Creation Second Step

The second step, as shown in Figure 4.23, will enable the stakeholder to add the widgets that should be shown in the generated report.

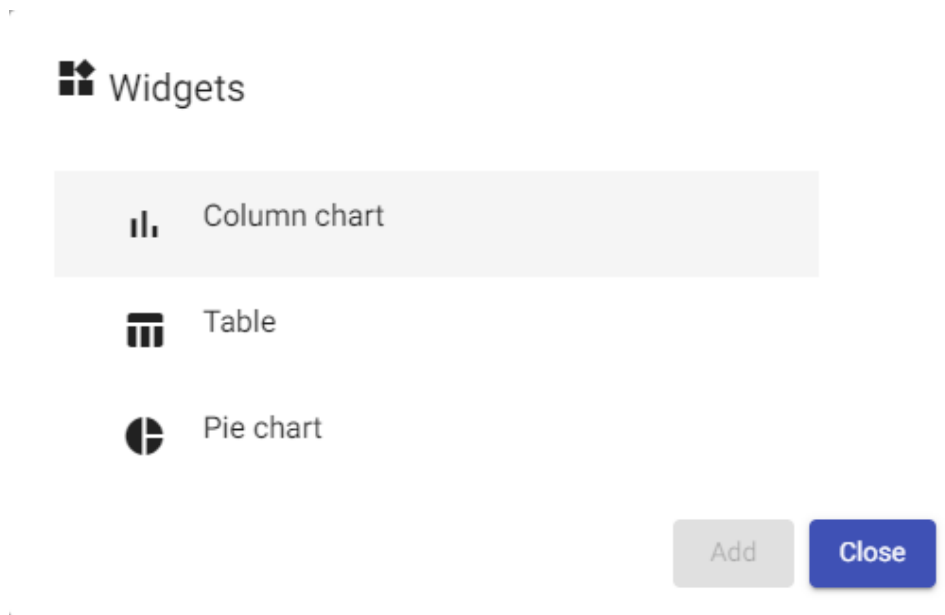


Figure 4.24.: Widgets Selection Dialog

As in the second step, the stakeholder has to select widgets, Figure 4.24 shows the dialog for the widgets that may be selected and added to the report. To clarify, the pie chart is added as an example.

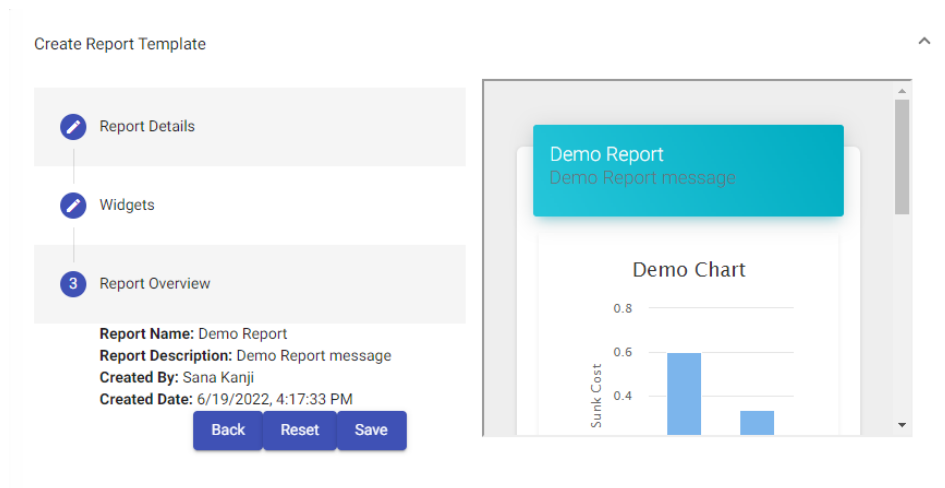


Figure 4.25.: Template Creation Third Step

The last step for the creation of the report template displays the summarized information entered in the first step as shown in Figure 4.25.

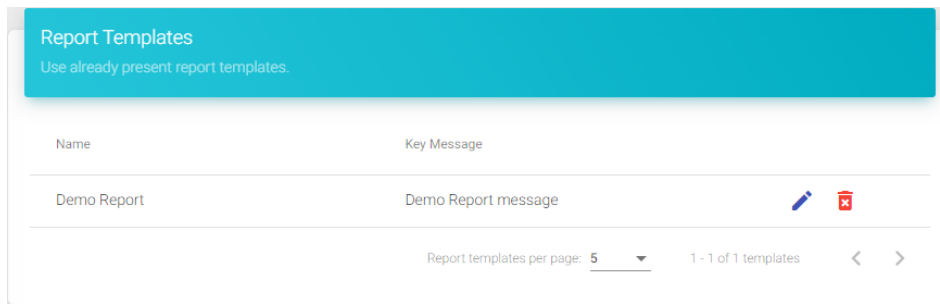


Figure 4.26.: Template Added in Report Templates Table

After the creation of the new template, the newly made template is added to the report templates table which can be seen in Figure 4.26. The template can be edited or deleted as explained in Sections 4.2.2, and 4.2.2.

Report Management

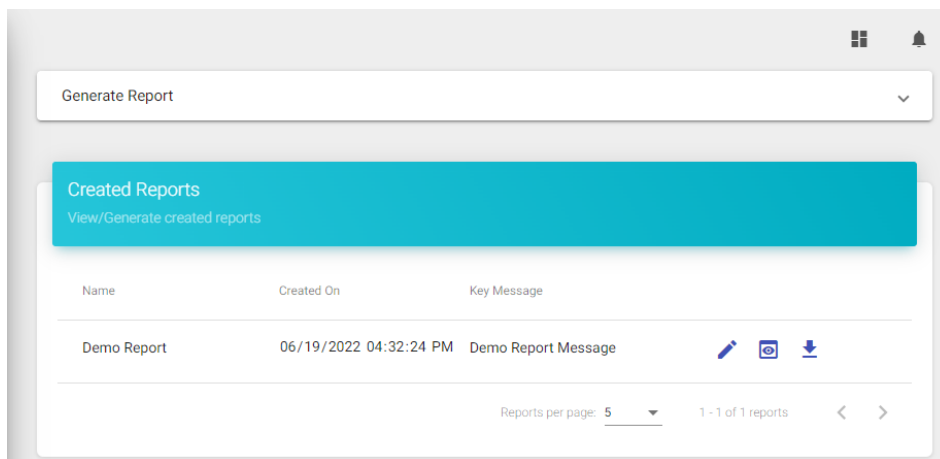


Figure 4.27.: Report Management View

Figure 4.27 shows the Report Management page, which is divided into two segments. The upper segment is for the generation of the new reports. The below segment is for displaying the list of generated reports.

4. Realization

The screenshot shows a web form titled "Generate Report" with a progress indicator showing "1 Report Details" and "2 Report Overview". The form contains the following fields and elements:

- Report Title ***: Demo Report
- Key Message**: Demo Report Message
- Created By ***: Sana Kanji
- Created Date**: 6/19/2022, 4:28:45 PM
- Select Template ***: Demo Report
- + Add Filter**: A button to add filters.
- Save Filter Name**: A button to save the filter name.
- Next**: A button to proceed to the next step.

Figure 4.28.: Report Generation First Step

The generation of the new report is a two-step process. The Figure 4.28 shows the first step in which the stakeholder can enter details in the fields which are explained in Section 4.2.3.

This is a close-up of the "Select Template" field from the previous figure. It shows a red asterisk next to the label "Select Template *". Below the label, it says "Pick one". A blue horizontal line separates the label from the selected option, "Demo Report".

Figure 4.29.: Report Template Selection Drop-Down

In the first step, the stakeholder has to select a report template that has to be used for the generation of the report. Figure 4.29 shows the drop-down option for selecting the possible report templates.

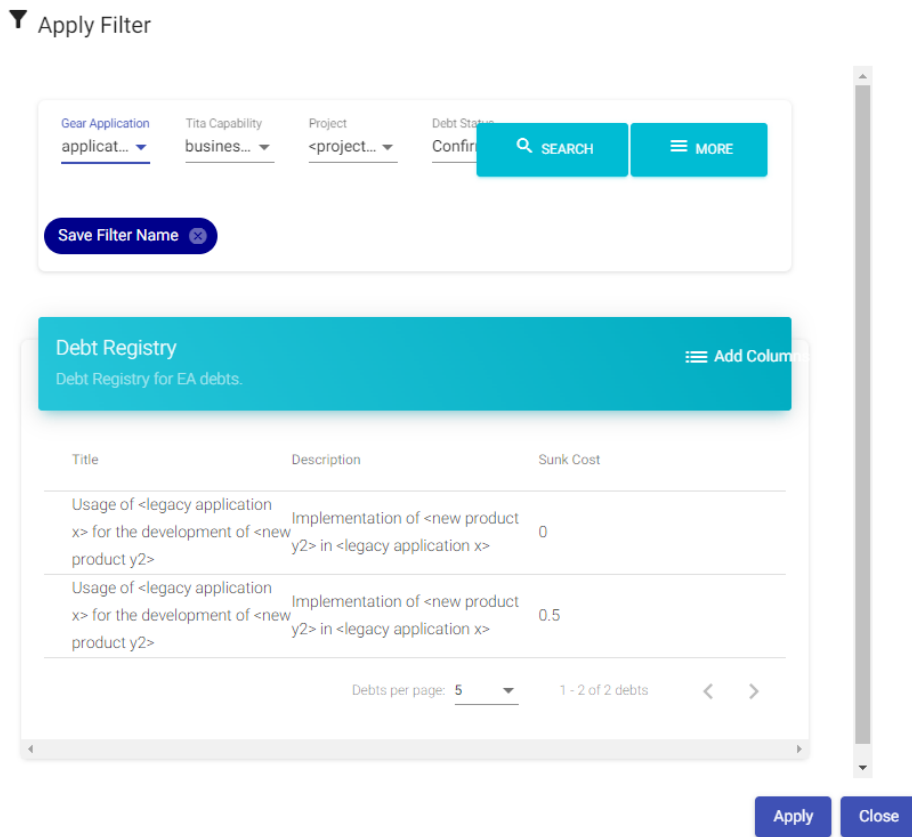


Figure 4.30.: Report Filter Selection

After selecting the template, there is a possibility of applying filters to the data that should be shown in the generated report. Figure 4.30 shows the dialog which opens the debt registry view in which we can use its filter functionality for filtering the data for the report.

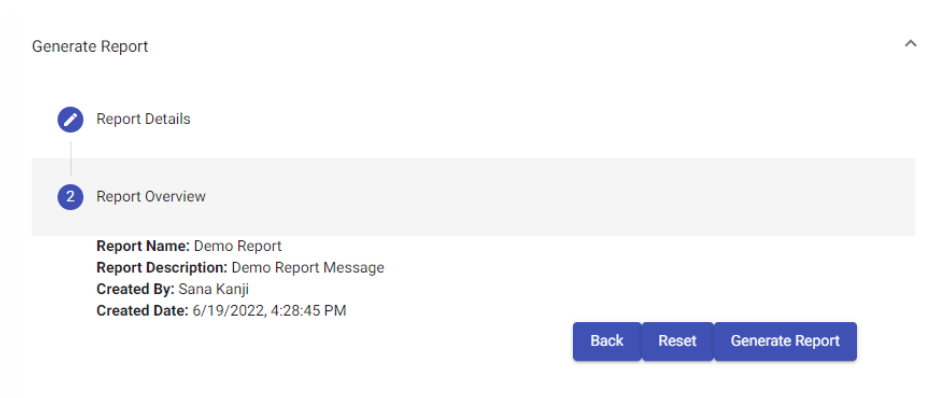


Figure 4.31.: Report Generation Second Step

4. Realization

The last step for the creation of the report displays the summarized information entered in the first step as shown in Figure 4.31.

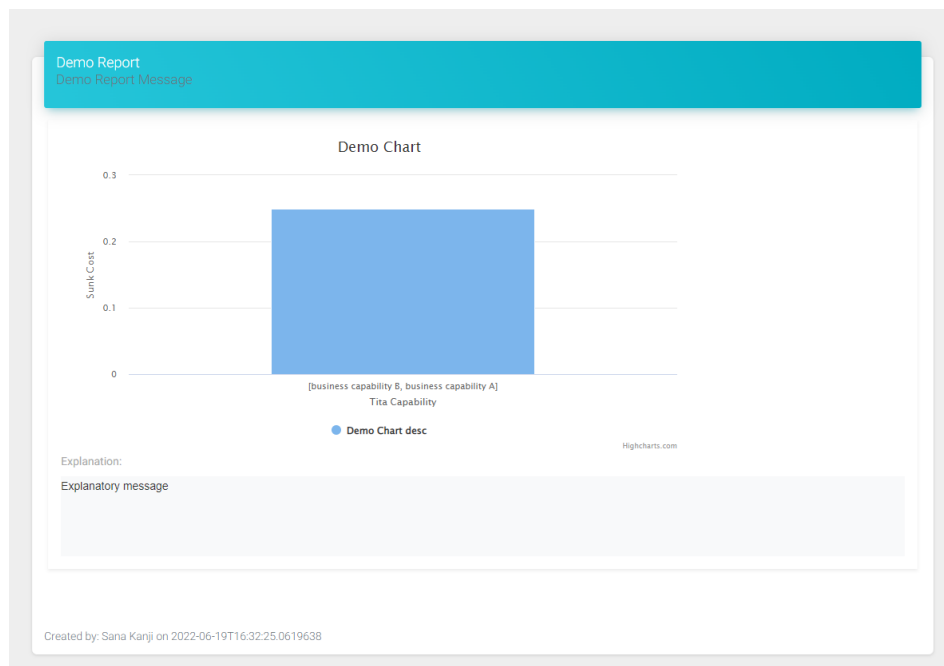


Figure 4.32.: Generated Report Preview

Figure 4.32 shows what a generated report looks like, this report consists of just a column chart.

Notifications

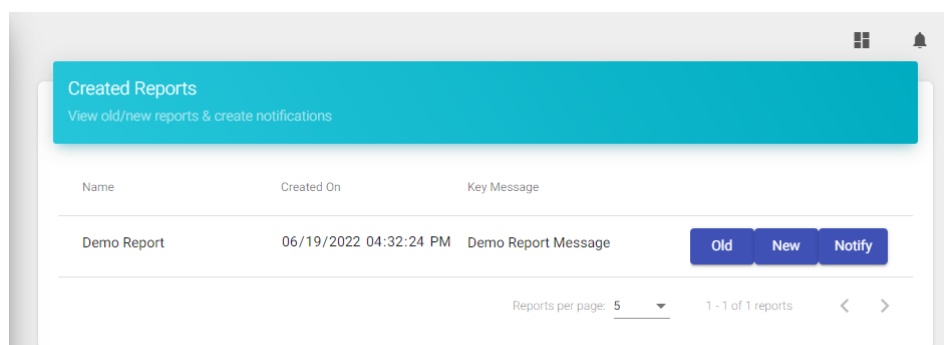
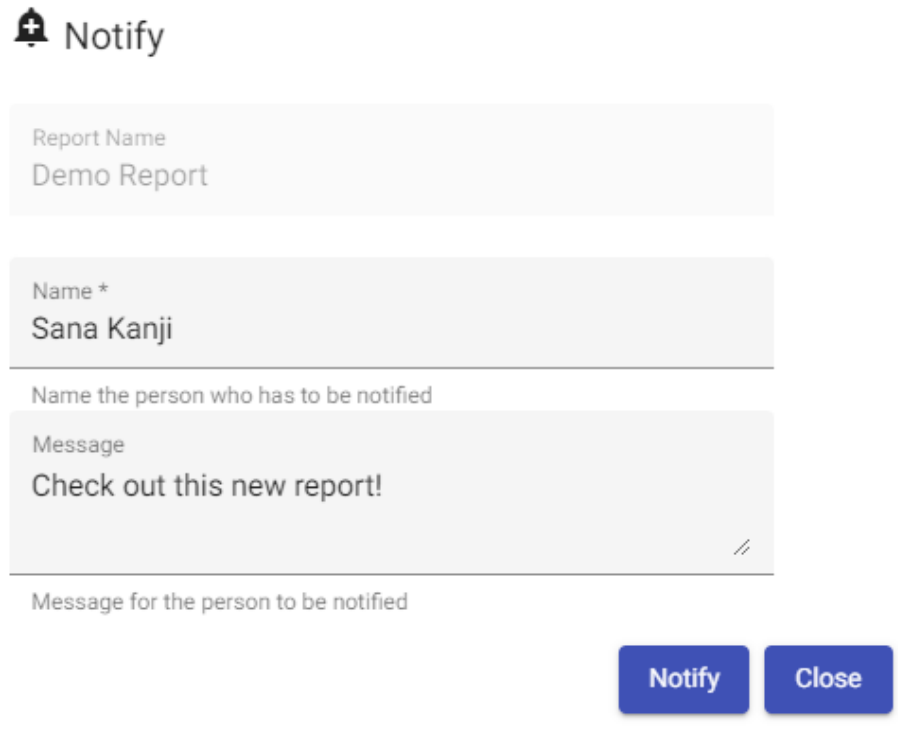


Figure 4.33.: Notifications View

In Figure 4.33, we see the Notification View; here, we have a list of generated reports. Each generated report gets three buttons, as described in Section 4.2.4. When a stakeholder clicks on Notify, the Notify dialog opens up, Figure 4.34.



The image shows a 'Notify' dialog box. At the top left is a bell icon followed by the word 'Notify'. Below this are three input fields: 'Report Name' with the value 'Demo Report', 'Name *' with the value 'Sana Kanji', and 'Message' with the value 'Check out this new report!'. Below the message field is a small double-slash icon. At the bottom right are two blue buttons labeled 'Notify' and 'Close'. Below the dialog box, the text 'Message for the person to be notified' is visible.

Figure 4.34.: Notify Dialog

When a user opens the Notify Dialog, as seen in Figure 4.34, they can enter the recipient's name and a message. This is explained in Section 4.2.4.

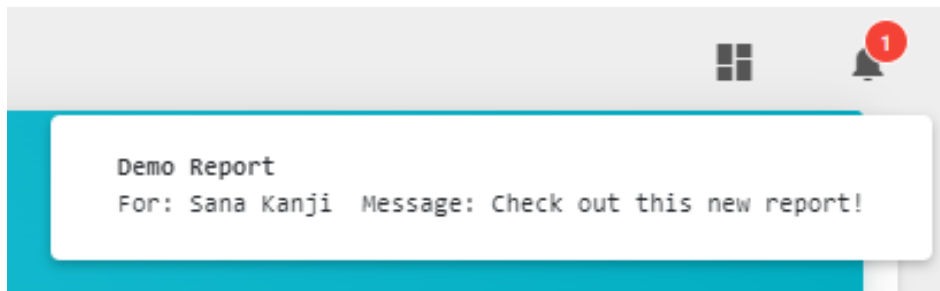


Figure 4.35.: Received Notification View

Figure 4.35 shows what a sent notification looks like; here, the notification's title contains the report name, and the body contains who it is meant for and what the message is.

5. Evaluation

Contents

5.1. Setup for Evaluation	49
5.2. Evaluation Results	53

In this chapter we will discuss the setup for the evaluation, the approach taken and the results we obtained from the evaluators.

5.1. Setup for Evaluation

For evaluating the EA debt reporting framework implementation, we have used an interactive video format to receive feedback and reviews from academic researchers and people working in the industrial domain. The interactive video embeds the questionnaire within the video so that when one section of the video is completed, the evaluators can answer the questionnaire regarding that section of the video. For making the interactive video, I have used OBS studio[Obs] for creating the video and voice recordings and Camtasia[Cam] for editing the video and adding the questionnaire.

The interactive video for the evaluation consists of eight sections; the video begins with learning the evaluator’s background and the level of familiarity with the term EA debt; after that, the sections are divided according to the front-end application of the EA debt reporting framework implementation which will be stated individually below. The video ends with some questions regarding the video and the concluding result on the EA debt reporting framework.

The questions for the sections mentioned above can be answered by selecting any one of the following values: strongly disagree, disagree, neutral, agree, and strongly agree, except for the first section, which is the background in which the evaluator must select one of the stated options which will be discussed in the background section. The sections of the video are as follows:

Background

In the background section, two questions are asked which are:

- In what role you are familiar with EA Debt?
- How knowledgeable are you with the term EA Debt?

The above questions can be answered by selecting the options given with the question. For the first question, the evaluator can choose a role: an EA debt researcher, an EA debt practitioner, a thesis student, or other. For the second question, the evaluator can select the familiarity with the term EA debt, which can be no knowledge, little knowledge, or very knowledgeable. These questions will help us know if the evaluator is working in this area and what level of expertise the evaluator possesses in EA debt.

Data Entry

The first component that is displayed is the data entry component, and after the explanation ends, the evaluator is shown with the set of questions, which are:

- The organization's debt registry can be provided like the one that we have been provided with for making this solution
- The upload functionality in the EA debt reporting framework is convenient

These questions will help us answer if other organizations can also provide a debt registry with similar attributes, if some changes must be done to accommodate other organizations, and whether they can upload it as an Excel document. The open question, in the end, gives them the opportunity to share any concerns or tips with us, as well as can help us learn about how they maintain their EA debt data and how the data can be provided to the EA debt reporting framework.

Debt Registry

The second component that is explained is the debt registry component. The functionalities offered by the debt registry are first shown, and then the questionnaire is displayed in which we would like to know the following:

- The filter parameters provided in the debt registry are sufficient.
- The filter functionality helps to view the data
- The application of filters process is clear
- The save and load filter functionality is useful
- The global filter functionality is useful

These questions will help us answer if the filter parameters are sufficient in the view of the evaluators filling the questionnaire or if there is a need for other filters to be added. Furthermore, together with the open question regarding the additional comments, they can give hints towards improvements that can be made.

Template Management

The third component displayed in the video is template management, in which the process for making templates is shown. In the end, our purpose is to know the following:

- The process of creating reusable templates supported in the EA debt reporting framework will save time.
- The widgets allow clear visualization of EA debt data.
- The template-making functionality in the EA debt reporting framework is straightforward.
- The iframe providing a preview of the template gives the stakeholder an impression of what the report will look like.

These questions will help understand if the functionality for making templates is straightforward and time-saving and if the provided widgets are sufficient. The open question will then allow us to know if some more additions need to be done, like adding more widgets or if a use case is missing.

Report Management

The video then displays the report management component, in which the process of generating and viewing reports using the created templates is shown. Afterward, we would like to know whether:

- The reports help understand the current situation of EA debts in the organization.
- The reports will help in better communication between the stakeholders.
- The reports help make better decisions after understanding the current situation of EA debts in the organization.
- Reports are an effective mechanism in making EA debts more visible.
- You would be able to generate a report using the EA debt reporting framework.

These questions will help understand whether the EA debt reports help with knowing the current situation of EA debts in the organization and if reports will help improve communication among various stakeholders. The further questions will help us know if these reports would contribute to the decision-making process after looking into the generated reports. The open question, in the end, will help us understand the additional features or improvements we must make in the report generation process.

Notification

Next, the notification component is displayed in which the stakeholder can view the old and new reports and generate a notification for the other stakeholders if the changes in the new report are worth notifying about, in which case, a new report is generated. In the end, we aim to know the following:

- Comparing the old and new reports helps in viewing the changes in the report
- Receiving a notification is a good indicator of newly regenerated reports
- Notifications about the report will help in taking action to mitigate the EA debts.
- The notification mechanism keeps the EA debt reporting process continuous

These questions will help in knowing if the changes in the report can be easily seen by comparing the old and new reports, if the notifications will contribute towards making the EA debt reporting process continuous and if the notifications are a good indicator for taking any possible actions for mitigating EA debt. In the end, the open question will help us investigate the other mechanisms which can help in making the EA debt reporting process continuous.

Dashboard

Lastly, the dashboard component is displayed in the video in which dashboard visualizations are shown. The following questions regarding the dashboard are then asked:

- The visualizations provide a summarized view of the EA debt situation
- The visualizations serve their purpose of providing a summary of the EA debt situation.
- The detail view (which opens when an EA debt is clicked in the table) of the EA debt allows for a quick insight into the EA debt details

These questions will help determine if the summarized view of the visualizations is helpful and if the detailed view of EA debt helps with a quick insight. The open question, in the end, will help in knowing if any additional visualizations must be added or additional EA debt information must be shown.

Video

The video ends with an additional questionnaire which will help us know if having a separate and specific framework for reporting EA debts will be helpful for the organizations. The questions asked are:

- Does the video portray the use cases clearly?
- The process of reporting on EA debts is now clear.
- Having a specific tool for the reporting of EA debts is useful.
- The EA debt reporting tool will be useful for organizations.

This open section, in the end, will enable in knowing any general remarks about the EA debt reporting framework and the evaluation.

5.2. Evaluation Results

The evaluation link was sent via email to the EA debt community within Europe and some IT professionals from which twelve responses were obtained in total from which one evaluator did not fill the evaluation completely and stopped after the first component 'Data entry'. The background information of the evaluators is shown in the below table.

EVALUATOR'S BACKGROUND	NOT KNOWLEDGEABLE	LITTLE KNOWLEDGE	VERY KNOWLEDGEABLE
AS A THESIS STUDENT		2	
AS AN EA DEBT RESEARCHER		1	
AS AN EA DEBT PRACTITIONER		1	1
OTHER	2	2	3

Table 5.1.: Evaluators Background and Expertise

We analyzed the data by using the average [Ana] as the statistical technique for the data we gathered. The data is based on strongly disagree, disagree, neutral, agree, and strongly agree. The average of the results that we received is shown in the Table 5.3

We have come to this average as follows: We gave all the possible options a value, from Strongly Disagree as 1 to Strongly Agree as 5, then we multiplied this value by the number of respondents that gave that answer, added them up together, and then divided them by the total number of responses. This results in a value that we then must match up with an answer. For this we have decided on the following ranges:

ANSWER TO SELECT	RANGE
STRONGLY DISAGREE	1 - 1.5
DISAGREE	1.5 - 2.5
NEUTRAL	2.5 - 3.5
AGREE	3.5 - 4.5
STRONGLY AGREE	4.5 - 5

Table 5.2.: Evaluation Result Range

5. Evaluation

For example, if the resulting value is 4.4, then the resulting answer will be Agree because it falls only in the range of Agree.

Question	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
The organization's debt registry can be provided like the one that we have been provided with for making this solution				✓	
The upload functionality in the EA debt reporting framework is convenient.				✓	
The filter parameters provided in the debt registry are sufficient.			✓		
The filter functionality helps to view the data					✓
The application of filters process is clear					✓
The save and load filter functionality is useful					✓
The global filter functionality is useful				✓	
The process of creating reusable templates supported in the EA debt reporting framework will save time					✓
The widgets allow clear visualization of EA debt data					✓
The template-making functionality in the EA debt reporting framework is straightforward				✓	
The iframe providing a preview of the template gives the stakeholder an impression of what the report will look like					✓
The reports help understand the current situation of EA debts in the organization				✓	
The reports will help in better communication between the stakeholders					✓
The reports help make better decisions after understanding the current situation of EA debts in the organization				✓	
Reports are an effective mechanism in making EA debts more visible				✓	
You would be able to generate a report using the EA debt reporting framework					✓
Comparing the old and new reports help in viewing the changes in the report				✓	
Receiving a notification is a good indicator of newly regenerated reports					✓
Notifications about the report will help in taking action to mitigate the EA debts				✓	
Notifications mechanism keeps the EA debt reporting process continuous				✓	
The visualizations provide a summarized view of the EA debt situation					✓
The visualizations serve their purpose of providing a summary of the EA debt situation				✓	
The detail view (which opens when an EA debt is clicked in the table) of the EA debt allows for a quick insight into the EA debt details				✓	
The process of reporting on EA debts is now clear.				✓	
Having a specific tool for the reporting of EA debts is useful.					✓
The EA debt reporting tool will be useful for organizations.				✓	

Table 5.3.: Evaluation Results

The questions in the evaluation have been formulated such that positive answers, like agree and strongly agree, indicate that the evaluators confirm that the requirements have been implemented and serve their purpose. From the results Table 5.3, we can see that the overall responses have been positive, and thus we can conclude that the implementation supports the requirements which can be seen in Section 3.1 Requirements. There is, however, one outlier: ‘The filter parameters provided in the debt registry are sufficient.’ The average response to this question is neutral, which will be discussed in the following Chapter 6 Discussion.

The feedback received in the open question in each section is shown in the Table 5.4. These comments will also be further discussed in the following Chapter 6 Discussion.

5. Evaluation

Video Section	Feedback
Data Entry	<p>Providing Drag and Drop functionality for uploading would be convenient</p> <p>Supporting different EA debt Registry formats</p> <p>Providing an integration for the source tools to automate the uploading</p> <p>There are so many terms (e.g., TITA, Gear Applications) that I am not familiar with. So it is hard to evaluate the approach presented</p>
Debt Registry	<p>Providing more filter attributes would support more ways of searching through the EA debts. Examples given:</p> <ul style="list-style-type: none"> - Architecture Layers - Creation Date - Sunk cost - Sunk cost over X amount <p>Instead of having global and normal filters, make the normal filters act like the global filters and add a clear filter button</p>
Report Template	<p>A full-screen preview rather than the one currently provided would help to better visualize the template</p> <p>Confirmation Dialogs before deletions to make the process a bit safer</p> <p>Template immutability will save you a lot of headaches!</p> <p>Instead of just adding a new template whenever it is edited, you could show a "latest version" column instead to prevent pollution in the table</p> <p>Additional widgets, like timeline charts to map the expected end dates of the EA debts into a view to visually show deadlines.</p> <p>It's really nice that you can add widgets in any order and quantity you like!</p>
Report Generation	<p>Isn't the "created by" a derivative of your account details?</p> <p>Can the generated reports be emailed to recipients? Would be great to keep users up to date!</p>
Notifications	<p>There is various feedback about the manual comparison of the old and new reports; it is referred to as cumbersome and time-consuming. The comparison of the changes in the old and new reports should be done automatically to make it easier for the users</p> <p>The notifications do not have a call to action by themselves and might get lost in the modern sea of notifications or simply ignored</p> <p>Receiving an email with a new report would be much appreciated!</p>
Dashboard	<p>Being able to order the EA debts based on the status (the red, yellow, and green balls) so that we can see the red ones on top!</p> <p>The details' view is cluttered and does not have the same status as is visible on the dashboard</p> <p>If the visualization is going to be used to show the outliers', then it can be a good fit (the tree map graphs). However, since (I think) the goal is to compare relative contributions, something like a pie chart would be better suited.</p> <p>The status view on the bottom of the dashboard page is a nice touch, but more information can be shown here to provide a quick overview.</p> <p>For example:</p> <ul style="list-style-type: none"> - EA debt owner - Last action taken at - Importance level (might be critical on the timeline but unimportant)
Video	<p>I really like this type of evaluation</p> <p>As a technical application demo, this is very nice (short and sweet). Nice work!</p> <p>Overall the tool looks really good. When can I use it?</p> <p>Very clear and understandable, the tool seems to be helpful, and I would love to use it.</p> <p>I miss the link to the actual process of solving the debt (scrum board, work items), but that might be beyond the scope.</p>

Table 5.4.: Evaluation Feedback

6. Discussion

Contents

6.1. Steps After Evaluation	57
6.2. Implications	61
6.3. Threats to Validity	62
6.3.1. Internal Validity	62
6.3.2. External Validity	63

In the previous chapter, we showed the results obtained from the evaluators. In this chapter, we are going to further look into those results and discuss the steps we took after the evaluation and discuss the implications. Lastly, we will identify the threats to both internal and external validity.

6.1. Steps After Evaluation

In the evaluation, we received some additional feedback on the open questions in each section of the video. The reason we had these questions was to gather suggestions for improvements that can be made. Below we will discuss the feedback per the video section:

Data Entry

For the Data Entry, as seen in Table 5.4, there are four feedback points, three of which were suggestions for improvement, and one was a comment about the terms used in the debt registry. We will go over these four points in order of the Table 5.4:

- The first feedback was a suggestion to implement a drag and drop functionality for uploading the EA debt registry. I agree this should be there because it would increase the user’s convenience. Since this is only a user interface change, it can be incorporated in the future as the user interface was the secondary goal of the thesis; the primary goal was the framework on which this has no effect.
- The next feedback in the table is to support multiple EA debt registries. I again agree with this, and the EA debt framework implementation has this considered, and it is extendable by allowing new adapters to be added that receive the other formats of EA debt registries and converting them to the

internal data models, as discussed in the Chapter 4 Realization. As this thesis scope was limited to the EA debt registry of only one organization, we will not implement other options at this point.

- The third provided feedback is a suggestion for proving a way to integrate with source tools, where the EA debt registry can be uploaded automatically. I agree that this would make it more convenient, as this is very similar to the previous feedback and can be supported by the framework in the future, but it is outside this thesis's scope.
- The last point of feedback was regarding the terms 'TITA capability' and 'GEAR application' being not understandable. The 'TITA capability' corresponds to 'business capability' and 'GEAR application' corresponds to 'applications', but the terminologies we have used in the implementation are limited to the organization for which we implemented this reporting framework. This generalized terminology can be accommodated in the future when a generalized tool is made and the organizations have more knowledge of the generalized terminologies. I disagree with this feedback as it was briefly explained in the excel file attached to the video details and further explained in the next section of the video.

Debt Registry

In the Debt Registry section of Table 5.4, there are two points of feedback, both of which are suggestions for improvements:

- The first suggestion relates to the filter attributes, where the evaluators would like to have more filter attributes, I agree, and while this was originally the goal of the implementation, it has been limited to the four attributes after some discussions with the product owner because these were the parameters that are frequently used in the organization whose EA debt registry we used. In the future, we can add more parameters for filtration as the need and use increase.
- The other suggestion was to make all filters global filters, as some might find this more intuitive, and to add a clear filter button. I have a neutral view on the first part of the feedback as this will depend more on the stakeholder needs that will vary with the organization. I disagree with the second point of the feedback as this functionality is already present and is also shown in the video. However, the button can be added in a more accessible position on the front-end. Since both points are just changes to the user interface, which was the secondary goal of this thesis, I would recommend looking more into this in the future.

Report Template

Next on Table 5.4 is the Report Template section, where we have received six points of feedback, of which four are suggestions, one definite compliment, and one which can be either, and are:

- The first suggestion is a full-screen preview that would help to visualize the template better. I agree this would be a good feature and could be implemented in a way where you can open it in another tab or overlay. However, as with the other front-end suggestions, this was part of the secondary goal, and I would highly suggest implementing this in the future.
- Next on the list is to add Confirmation dialogs before deletions to make the process safer. I agree this would be a great feature to implement, as it would save stakeholders from accidentally removing the wrong things. Due to time limitations, we recommend it to be added in the future.
- The third point can be interpreted in two ways; if it is a suggestion, then I would agree because that is what we have implemented in the Template Management component of the framework. Because we never edit an existing template but rather duplicate it and then change that version, so it does not impact existing reports. The other interpretation can be as a compliment, in which case we do not have to accommodate any changes, and our implementation is appreciated.
- The fourth point touches on the immutability as well, where for every edit, a new template is created; they suggest adding a column with the latest version and only showing this to prevent clutter in the template table. I agree this might be a good addition, but this has some limitations that would have to be also resolved, like accessing older versions of a template. Since this is quite a change in the table, I recommend doing this in the future.
- The fifth point is regarding additional widgets, of which one example is given. I agree that additional widgets should be added, but this should be done in the future according to the needs of the organizations.
- The last point for the Report Template section is a compliment on the widget functionality and the configurability of the reports.

Report Generation

In the Report Generation section of Table 5.4, there are two points of feedback, one of which has a suggestion, and the other is a question, as follows:

- The first point is a question regarding the 'created by' field that has to be filled in by the person making the report. And they are under the impression that

a user is logged into an account. I would agree with this point if that latter were the case; however, since we have kept user accounts out of the scope of this thesis, this is not the case and has to be entered manually. We can take a suggestion for this question, which would be to implement a user account mechanism, which I think should be accommodated in the future.

- The second point starts with a question about whether the reports can be emailed to the user; they feel this would be a good way to keep stakeholders up to date. I agree with this suggestion; it was part of the scope but has been removed due to time limitations and the unavailability of technical resources. Hence, this should be done in the future.

Notifications

In the Notifications section of Table 5.4, there are three points of feedback, all of which are suggestions for improvement, and are:

- The first point is about the reports' manual comparison, which has to be done by the stakeholders. I agree this should be implemented, but due to time limitations has been removed in discussion with the product owner as other requirements were more important. This should be implemented in the future to make the whole tool more useable.
- The second point warns of the notification not having a call to action, meaning they do not immediately attract the user's attention, which could result in the notifications getting lost. I agree this is the case, but we cannot determine which user to notify because we do not have a user account mechanism. Therefore, in collaboration with the product owner, we have decided to display the notification on the navigation bar.
- The last point is again about receiving emails with the new reports, which has been addressed in the Generate Report Section 4.2.3. This has been excluded due to the unavailability of technical resources. I agree this should be implemented in the future when the reports can also be emailed to the stakeholders.

Dashboard

In the Dashboard section of Table 5.4, there are four points of feedback, all of which are suggestions for improvements, which are:

- The first point is about ordering the tables to more easily see the overdue and critical EA debts. I agree this is a good addition, as it will make the dashboard more usable; however, due to time limitations and the front-end not being the main focus of this thesis, I would suggest implementing this in the future.

- The next point is about the details view. They mention it to be cluttered and that it is missing the status which is visible on the dashboard. I agree this should be improved, but as this was the secondary goal of the thesis, it should be implemented in the future to make the view more useable.
- The third point on the list suggests the use of pie charts, rather than the treemaps that have been implemented, might better suit the goal of comparing relative contributions. I agree with the statement; however, as stated at the beginning of this suggestion, our goal was to show highly impacted areas (the outliers) more quickly.
- The last point of feedback was a suggestion to add more fields to the Debt Item stats table. I disagree that adding more fields could be beneficial; we do not want to create too much clutter on the dashboard. The additional fields, for that reason, have been added to the detailed view of the EA debt item.

Video

In the Video section of Table 5.4, there are five points of feedback, one of which has a suggestion, and the others are compliments:

- The first four points are compliments about the video, the evaluation, and the EA debt reporting framework implementation.
- The last point mentions the evaluator missing the link to actually solving the debt, but as the evaluator already mentions, this is beyond the scope of this thesis.

6.2. Implications

For the implications, we want to differentiate between implications for researchers and implications for practitioners. First, we will address the implications for the researcher and how they can proceed further in the research, and then we will look into the implications for the practitioners.

Implications for Researchers

This thesis addresses the monitoring, communication & documentation activities of the EADM framework and has created the prototype for the EA debt reporting framework. As mentioned in Chapter 4 Realization, the current implementation is based on only one EA debt registry, and to further extend the current solution, researchers could research further into including EA debt registries of different types and from other organizations. Moreover, they can explore alternative views that might suit the needs of the practitioners better, as well as extend the change

management component of the framework to support alternative ways of making the reporting process continuous. Lastly, since this thesis only touched three activities of the EADM [Ale+20], a way to integrate this framework with the other layers of the EADM framework has to be researched.

Implications for Practitioners

The EA debt reporting framework implementation allows practitioners to ease their work. It provides methods to document and keep track of EA debts and to report on them by providing an extensive way of creating reusable report templates in a user-friendly interface and using these templates to create reports over different data filters. The practitioners get a wide selection of attributes to show and filter their reports on, as discussed in Chapter 4 Realization. Furthermore, it allows them to search through the EA debt registry, using a wide variety of filter attributes, and to see the parts of the EA debts that they regard as important by selecting the columns they want to see. Lastly, they can use the provided notification function to alert the stakeholders of any new reports that contain changes that need to be communicated.

The EA debt reporting framework will contribute to a process for reporting EA debts by providing a mechanism in which reports can be used as means of communication & documentation and a dashboard for monitoring EA debts. As stated in Chapter 2 background and related work, communication and documentation are already among the identified EA debts, and we needed a process to resolve these issues. Therefore, this thesis aims to implement the EA debt reporting framework, and by looking at the evaluation results, it can be concluded that having a framework for reporting EA debts will be helpful in the organization and will help the EA practitioners in making better decisions and staying updated with the current situation of EA debts in the organization.

6.3. Threats to Validity

In this section, we will discuss the threats to the validity of the evaluation. We need to do this because we want to know how reliable our evaluation is or whether it should require more consideration.

6.3.1. Internal Validity

There is a possible selection bias [Int] due to the limited responses in the evaluation, even though the evaluation had been sent to a relatively large group from which only twelve people participated. Of these twelve evaluators, as can be seen in Table 5.1, only five come from a background whose role is linked with the EA debts. And only four out of all evaluators are very knowledgeable. Another threat

to the internal validity can be the attrition [Int] threat. Due to some technical issues at the beginning of the evaluation, at least one of the evaluators dropped out; we know this because they informed us of the technical problem.

6.3.2. External Validity

Reflecting on the implementation done for the making of the reporting framework of EA debts, we have based it on an EA debt registry of one organization which was in the form of an excel document; this means that the current implementation is limited to receiving the EA debt registry in the form of an excel document. While this was in the scope of our research for this thesis, as discussed in Chapter 4 Realization, this can be extended with ease in the future when there is access to the EA debt registries of other organizations too.

7. Conclusion

Contents

7.1. Summary	65
7.2. Future Work	66

To conclude this thesis, we will first summarize the steps taken in the thesis, then we will discuss the future work that can be done to further research this topic.

7.1. Summary

In this thesis, we came up with a process and mechanism to report EA debts. We identified the stakeholder's requirements and then decided on the components that should be included in the EA debt reporting framework: data management, template management, report management, and change management. Each of the component's use cases are explained in Chapter 3 Concept and how it contributes to achieving the identified requirements. Then we look into the implementation in Chapter 4 Realization. We also implemented a front-end application so that the stakeholders can interact with the EA debt reporting framework, use it to generate reports, and keep stakeholders informed with those reports. Furthermore, the implementation also supports practitioners in searching through the EA debt registry, by providing them with an extensive set of filter attributes, to see the attributes they want to see from the EA debts.

Finally, we sent out an evaluation to get feedback from the academic researchers and the people from the industrial domain, and discussed the suggestions we received on the evaluation, after which we identified the implications and threats to internal and external validity. The feedback on the evaluation was overall positive; however, we received some feedback which can be seen in Table 5.4 Evaluation Feedback. The changes mostly suggested are the UI component extensions like adding more filters, and visualizations that can be done in the future once the EA debt reporting framework starts being used in the organization.

This EA debt reporting framework presents a prototype that contributes toward having a mechanism specifically dedicated to EA debts, and further research can then be guided toward how to accommodate more organizations.

The EA debt reporting framework also contributes to the monitoring, documentation & communication activities of EADM. However, further research on how to

have a more sophisticated reporting framework, and deal with various debt registries still needs to be done by conducting interviews with EA practitioners and a diverse set of organizations.

7.2. Future Work

This EA debt reporting framework is the first step toward the monitoring, documentation & communication activities in EADM; further work is needed.

As mentioned in Section 6.2 Implications, to further verify the applicability and usability of the EA debt reporting framework, we need to get EA debt registries from various organizations to know how they are maintained and then add adapters to facilitate those debt registries which has been explained in Chapter 4 Realization.

Furthermore, more widgets can be added, as per the requirements of the stakeholders in the organization, to have more variety of visualizations; for example, pie charts could be added to compare relative contributions, and timeline charts could be added to keep better track of EA debt statuses. Some additional user interface changes have been recommended by the evaluators and have been discussed in Section 6.1 Steps after Evaluation.

Moreover, this solution has a notification mechanism to keep the EA debt reporting process continuous, which keeps the stakeholders updated with the EA debt changes that occur in the organization, as explained in the Subsection 4.2.4 Notifications. This mechanism still needs manual work to look at the changes in the report. Therefore, we can make this process automatic, by detecting the EA debt changes and the reports affected by those changes. This can be done through a scheduler that checks EA debt reports for changes at specific intervals (for example, every 2 hours), and then emails updated reports to the concerned stakeholders.

The EA debt reporting framework implementation needs an authentication and authorization mechanism for auditing purposes and enabling the notification to be sent to a specific user, rather than everyone, as is the case now.

Lastly, this framework only implements three activities from the complete EADM framework; as suggested by Alexander et al. [Ale+20], it needs to be integrated with the other framework activities. This could be achieved by loading the newly identified EA debts from the identification activity from the EADM framework, or even by letting the assessment & prioritization activity update the already stored EA debts. This integration could be done by creating the relevant adapters in the EA debt reporting framework to receive EA debts over HTTP using REST and JSON.

To further evaluate this solution, a workshop can be held with the practitioners so that they can use the EA debt reporting framework implementation, potentially in a real work setting with the EA debt registries they provide. While the front-end of the solution has been tested vigorously, it has never been evaluated for being user-friendly, by letting users use the tool.

A. Debt Item Attributes

Debt Item Table

Attribute name	Description	Type
Id	Unique number for the database	Number
Architecture Layer	Architecture layers involved <ul style="list-style-type: none"> • BA: Business Architecture • AA: Application Architecture • IA: Information Architecture • TA: Technology Architecture 	String
Comment	Further comments/remarks about this EA debt (e.g., reason, opinion)	String
Created Date	The Date the EA debt has been uploaded	Date Time
Description	Explanation of the object	String
Evidence	Why is this an EA debt? Which architecture principles, guidelines, or standards are violated?	String
Excel Id	Unique number in the excel EA debt registry	Number
Implementation Date	Implementation date of the EA debt	Date Time
Last Update	When the EA debt has been last updated in the excel EA debt registry	Date Time
Modified Date	When the EA debt has been last updated in the database	Date Time
Name	Title of the object	String
Recommended Action	Recommended action to deal with the EA debt, to achieve compliance	String
Status	The status of the EA debt Item <ul style="list-style-type: none"> • Open: Waiting for confirmation as an EA debt • Confirmed: Fully confirmed as an EA debt • Discarded: Planned debt solution not implemented • Planned for mitigation: Time and budget for its mitigation have been planned • Mitigated: Mitigation completed 	String
Sunk Cost	Sunk costs incurred in Mio €.	Number
Debt Period	How long this EA debt is estimated to remain in the system <ul style="list-style-type: none"> • Short-term: Up until 1-2 years • Medium-term: 2-5 years • Long-term: More than 5 years 	String

Project Table

Attribute name	Description	Type
Id	Unique number for the database	Number
Description	Brief explanation of the project	String
Name	Related project name	String

Tita Capability Table

Attribute name	Description	Type
Id	Unique number for the database	Number
Description	Brief explanation of the affected business capability	String
Name	Affected business capability name	String

Gear Application Table

Attribute name	Description	Type
Id	Unique number for the database	Number
Description	Brief explanation of the affected application	String
Name	Affected application name	String

A. Debt Item Attributes

Consequence Table

Attribute name	Description	Type
Id	Unique number for the database	Number
Affected Business/IT Obj	Affected business/IT capability	String
Description	Brief explanation of the consequence	String
Severity Description	Description of the severity of the consequence	String
Severity Level	Severity of this consequence for the fulfillment of the target IT architecture <ul style="list-style-type: none"> • High: It makes things unreasonably difficult/expensive • Medium: It makes things more difficult or more expensive • Low: It is undesirable but has no functional impact" 	String

Report Template Table

Attribute name	Description	Type
Id	Unique number for the database	Number
Created By	Name of the creator of the template	String
Created Date	The Date the template has been created	Date Time
Deleted	Has the template been deleted	Boolean
Design	Template design	String
Key Message	Description to be included in the template	String
Template Id	Template used to create this new template	Number
Title	Name of the template	String
Version	Version of the template	Number

Created Report Table

Attribute name	Description	Type
Id	Unique number for the database	Number
Created By	Name of the creator of the template	String
Created Date	The Date the template has been created	Date Time
Generated Report Metadata	The design of the report in JSON	String
Key Message	Description to be included in the report	String
Name	Title of the report	String
Selected Report Filter	Filter parameters selected for the report in JSON	String

Filter Store Table

Attribute name	Description	Type
Id	Unique number for the database	Number
Filter JSON	Selected filters for the saved filter in JSON	String
Name	Title given for saving the filters	String

Notification Table

Attribute name	Description	Type
Id	Unique number for the database	Number
Message	Message to be shown in the notification	String
Name	Name of the user for which the notification is being sent	String
Report Id	Report Id for which the notification is being sent	Number
Report Name	Report name for which the notification is being sent	String
Created Date	Date the notification has been created	String

Bibliography

- [Ale+20] P. Alexander et al. "A Framework for Managing Enterprise Architecture Debts - Outline and Research Directions." In: *EMISA*. 2020 (cit. on pp. 2, 3, 7, 62, 66).
- [Ana] *Analyse Quantitative Data*. <https://www.ncvo.org.uk/help-and-guidance/strategy-and-impact/impact-evaluation/evaluation-and-impact-reporting/how-to-analyse-quantitative-data-for-evaluation/>. [Online; accessed 11-August-2022] (cit. on p. 53).
- [Ang] *Angular*. <https://angular.io/>. [Online; accessed 11-August-2022] (cit. on pp. 19, 21).
- [Bac] *EADebtReportingBackEnd README*. <https://git.rwth-aachen.de/swc-public/eadm/ea-debt-reporting/EADebtReporting/-/blob/main/README.md>. [Online; accessed 11-August-2022] (cit. on p. 27).
- [BBL12] S. Bente, U. Bombosch, and S. Langade. "Chapter 2 - What Is Enterprise Architecture?" In: *Collaborative Enterprise Architecture*. Ed. by S. Bente, U. Bombosch, and S. Langade. Boston: Morgan Kaufmann, 2012, pp. 31–38. isbn: 978-0-12-415934-1. doi: <https://doi.org/10.1016/B978-0-12-415934-1.00002-9>. url: <https://www.sciencedirect.com/science/article/pii/B9780124159341000029> (cit. on p. 5).
- [BH19] N. Banaeianjahromi and R. Hekkala. "Factors Influencing Communication and Collaboration in Enterprise Architecture Development." In: *HICSS*. 2019 (cit. on p. 1).
- [Bir] *BIRT*. <https://eclipse.github.io/birt-website/>. [Online; accessed 11-August-2022] (cit. on p. 9).
- [Bro] *Browserless*. <https://www.browserless.io/>. [Online; accessed 11-August-2022] (cit. on p. 19).
- [Bro+10] N. Brown et al. "Managing technical debt in software-reliant systems." In: *FoSER '10*. 2010 (cit. on p. 7).
- [Cam] *Camtasia*. <https://www.techsmith.com/video-editor.html>. [Online; accessed 11-August-2022] (cit. on p. 49).
- [Col] *Highchart Column Chart*. <https://www.highcharts.com/docs/chart-and-series-types/column-chart>. [Online; accessed 11-August-2022] (cit. on p. 27).

- [Cre] *Material Dashboard Angular*. <https://www.creative-tim.com/product/material-dashboard-angular2>. [Online; accessed 11-August-2022] (cit. on p. 19).
- [Cun92] W. Cunningham. "The WyCash portfolio management system." In: *OOP-SLA '92*. 1992 (cit. on p. 7).
- [Fro] *EADebtReportingFrontEnd README*. <https://git.rwth-aachen.de/swc-public/eadm/ea-debt-reporting/eadebtreportingfrontend/-/blob/master/README.md>. [Online; accessed 11-August-2022] (cit. on p. 27).
- [Hac+19] S. Hacks et al. "Towards the Definition of Enterprise Architecture Debts." In: *2019 IEEE 23rd International Enterprise Distributed Object Computing Workshop (EDOCW)* (2019), pp. 9–16 (cit. on pp. 1, 6).
- [Han] *Inheritance with Json*. <https://techcommunity.microsoft.com/t5/sql-server-blog/handling-inheritance-with-json/ba-p/384744>. [Online; accessed 11-August-2022] (cit. on p. 27).
- [HBA17] S. Hacks, M. Brosius, and S. Aier. "A Case Study of Stakeholder Concerns on EAM." In: *2017 IEEE 21st International Enterprise Distributed Object Computing Workshop (EDOCW)*. 2017, pp. 50–56. doi: 10.1109/EDOCW.2017.17 (cit. on p. 6).
- [Hib] *Hibernate*. <https://hibernate.org/>. [Online; accessed 11-August-2022] (cit. on p. 11).
- [Hig] *Highcharts*. <https://www.highcharts.com/>. [Online; accessed 11-August-2022] (cit. on pp. 24, 27).
- [HJ20] S. Hacks and J. Jung. "Enterprise Architecture Debts—A Concept to Manage EA Evolution?" In: *ICIS 2020 TREOs*. 2020 (cit. on p. 7).
- [Int] *Internal Validity*. <https://www.scribbr.com/methodology/internal-validity/>. [Online; accessed 11-August-2022] (cit. on pp. 62, 63).
- [Jac] *Jackson*. <https://github.com/FasterXML/jackson/>. [Online; accessed 11-August-2022] (cit. on p. 27).
- [Jas] *Jasper*. <https://www.jaspersoft.com/>. [Online; accessed 11-August-2022] (cit. on p. 8).
- [Jav] *Java*. <https://www.java.com/>. [Online; accessed 11-August-2022] (cit. on p. 19).
- [Jun+21] J. Jung et al. "Revealing Common Enterprise Architecture Debts: Conceptualization and Critical Reflection on a Workshop Format Industry Experience Report." In: *2021 IEEE 25th International Enterprise Distributed Object Computing Workshop (EDOCW)*. 2021, pp. 271–278. doi: 10.1109/EDOCW52865.2021.00058 (cit. on pp. 1, 3, 7).

- [Kiu] Kiuwan. <https://www.kiuwan.com/>. [Online; accessed 11-August-2022] (cit. on p. 8).
- [KSS15] S. Kotusev, M. Singh, and I. Storey. "Investigating the Usage of Enterprise Architecture Artifacts." In: *ECIS*. 2015 (cit. on p. 5).
- [Lan+05] M. Lankhorst et al. *Enterprise architecture at work: Modelling, communication, and analysis*. Jan. 2005, pp. 1–334. doi: 10.1007/3-540-27505-3 (cit. on p. 1).
- [Lap12] J. Lapalme. "Three Schools of Thought on Enterprise Architecture." In: *IT Professional* 14.6 (2012), pp. 37–43. doi: 10.1109/MITP.2011.109 (cit. on pp. 5, 7).
- [LKL10] C. Lucke, S. Krell, and U. Lechner. "Critical Issues in Enterprise Architecting - A Literature Review." In: *AMCIS*. 2010 (cit. on p. 1).
- [LMR16] M. Lange, J. Mendling, and J. Recker. "An empirical analysis of the factors and measures of Enterprise Architecture Management success." In: *European Journal of Information Systems* 25 (2016), pp. 411–431 (cit. on p. 6).
- [Mav] Maven. <https://maven.apache.org/>. [Online; accessed 11-August-2022] (cit. on p. 22).
- [Obs] OBS Studio. <https://obsproject.com/>. [Online; accessed 11-August-2022] (cit. on p. 49).
- [Pen] Pentaho. <https://www.hitachivantara.com/en-us/products/lumada-dataops/data-integration-analytics.html>. [Online; accessed 11-August-2022] (cit. on p. 9).
- [PH19] L. Pavli and T. Hlis. "The Technical Debt Management Tools Comparison." In: *SQAMIA*. 2019 (cit. on pp. 7, 8).
- [Pos] PostgreSQL. <https://www.postgresql.org/>. [Online; accessed 11-August-2022] (cit. on p. 19).
- [Rot+13] S. Roth et al. "Enterprise Architecture Documentation: Current Practices and Future Directions." In: *Wirtschaftsinformatik*. 2013 (cit. on pp. 1, 6).
- [Son] SonarQube. <https://www.sonarqube.org/>. [Online; accessed 11-August-2022] (cit. on p. 7).
- [Spr] Spring. <https://spring.io/>. [Online; accessed 11-August-2022] (cit. on pp. 19, 21).
- [Sprb] Spring Data JPA. <https://spring.io/projects/spring-data-jpa>. [Online; accessed 11-August-2022] (cit. on p. 11).

- [Squ] *Square*. <https://www.vector.com/int/en/products/products-a-z/software/square/>. [Online; accessed 11-August-2022] (cit. on p. 8).
- [SSS17] N. Silva, M. Mira da Silva, and P. Sousa. "A c for Analyzing Enterprise Architecture Evolution." In: *2017 IEEE 21st International Enterprise Distributed Object Computing Conference (EDOC)*. 2017, pp. 20–29. doi: 10.1109/EDOC.2017.13 (cit. on pp. 5, 6).
- [Tam+11] T. Tamm et al. "How Does Enterprise Architecture Add Value to Organisations?" In: *Commun. Assoc. Inf. Syst.* 28 (2011), p. 10 (cit. on p. 5).
- [Tea] *Teamscale*. <https://www.cqse.eu/en/teamscale/overview/>. [Online; accessed 11-August-2022] (cit. on p. 8).
- [THC04] A. Tang, J. Han, and P. Chen. "A comparative analysis of architecture frameworks." In: *11th Asia-Pacific Software Engineering Conference* (2004), pp. 640–647 (cit. on p. 6).
- [UM06] L. Urbaczewski and S. Mrdalj. "A comparison of enterprise architecture frameworks." In: 2006 (cit. on p. 6).

Glossary

API Application Programming Interface

CSS Cascading Style Sheets

DoDAF Department of Defense Architecture Framework

DTOs Data Transfer Objects

EA Enterprise Architecture

EADM Enterprise Architecture Debt Management Framework

ERD Entity Relationship Diagram

FEAF Federal Enterprise Architecture Framework

HTML Hypertext Markup Language

HTTP HyperText Transfer Protocol (HTTP) standard application-level protocol used for exchanging files on the World Wide Web (WWW).

JSON JavaScript Object Notation (JSON) is an open standard format for data interchange.

REST Representational state transfer (REST) is an architectural style for hypermedia systems. It guides the design and development of the architecture of World Wide Web (WWW).

TD Technical Debt

TOGAF The Open Group Architectural Framework

UI User Interface

