HAW HAMBURG

BACHELOR THESIS
Vladislav Konovalov

Comparison of Enterprise Architecture Tools for Their Usability and Integration Capability in DevOps Scenarios

Faculty of Engineering and Computer Science
Department Computer Science

HOCHSCHULE FÜR ANGEWANDTE
WISSENSCHAFTEN HAMBURG
Hamburg University of Applied Sciences

Vladislav Konovalov

Comparison of Enterprise Architecture Tools for Their Usability and Integration Capability in DevOps Scenarios

Bachelor thesis submitted for examination in Bachelor´s degree
in the study course *Bachelor of Science Angewandte Informatik*
at the Department Computer Science
at the Faculty of Engineering and Computer Science
at University of Applied Science Hamburg

Supervisor: Prof. Dr. Ulrike Steffens
Supervisor: Prof. Dr. Stefan Sarstedt

Submitted on: 23rd of March 2023

Vladislav Konovalov

Thema der Arbeit

Comparison of Enterprise Architecture Tools for Their Usability and Integration Capability in DevOps Scenarios

Stichworte

Enterprise Architecture Management, LeanIX, ADOIT, LUY, Vergleich, DevOps, Integration

Kurzzusammenfassung

Enterprise Architecture Management (EAM) Tools spielen eine entscheidende Rolle bei der Unterstützung von Organisationen, komplexe IT-Umgebungen zu verwalten. Da DevOps-Praktiken immer mehr an Bedeutung gewinnen, wird es zunehmend wichtiger, die Benutzerfreundlichkeit und Integrationsfähigkeiten von EAM-Tools in DevOps-Szenarien zu bewerten. Diese Arbeit präsentiert einen Vergleich verschiedener EAM-Tools und bewertet deren Potenzial für die Integration in DevOps-Umgebungen. Die Studie verwendet einen multimethodischen Ansatz, bestehend aus Schnittstellenanalyse, Dokumentationsüberprüfung und Beispiel-API-Anfragen, um ein ganzheitliches Verständnis der Leistung der EAM-Tools im Kontext von DevOps zu bieten.
Die Methode der Schnittstellenanalyse konzentriert sich auf die Funktionalität der APIs der EAM-Tools und bewertet deren Merkmale, Benutzerfreundlichkeit und Unterstützung für DevOps-Praktiken. Die Methode der Dokumentationsüberprüfung bewertet die Qualität, Vollständigkeit und Klarheit der offiziellen Dokumentation der EAM-Tools und berücksichtigt Aspekte, die für deren Einsatz in DevOps-Kontexten am relevantesten sind. Schließlich werden durch die Beispiel-API-Anfragen praktische Anwendungsfälle der APIs der EAM-Tools demonstriert, die ein Benutzer oder Entwickler im Rahmen der Integration von EAM-Tools in DevOps-Szenarien durchführen könnte.

Vladislav Konovalov

Title of Thesis

Comparison of Enterprise Architecture Tools for Their Usability and Integration Capability in DevOps Scenarios

Keywords

Enterprise Architecture Management, LeanIX, ADOIT, LUY, Comparison, DevOps, Integration

Abstract

Enterprise Architecture Management (EAM) tools play a critical role in helping organizations maintain and manage complex IT environments. As DevOps practices gain prominence, it becomes increasingly important to evaluate the usability and integration capabilities of EAM tools in DevOps scenarios. This thesis presents a comparison of various EAM tools, assessing their potential for integration within DevOps environments. The study employs a multi-method approach, including Interface Analysis, Documentation Review, and Example API Requests, to provide a holistic understanding of the EAM tools' performance in the context of DevOps.

The Interface Analysis method focuses on the functionality of the EAM tools' APIs, evaluating their features, ease of use, and support for DevOps practices. The Documentation Review method assesses the quality, comprehensiveness, and clarity of the EAM tools' official documentation. The Example API Requests section demonstrates practical use cases of the EAM tools' APIs by providing sample requests for typical tasks that a user or developer might perform in the context of integrating EAM tools within DevOps scenarios.

Contents

List of Figures

1 Introduction

1.1 Motivation

Enterprise architecture (EA) is a discipline that focuses on the alignment of business and information technology (IT) in an organization. EA tools are used to support the creation, documentation, and analysis of EA models, which help to ensure that the organization's business and IT are aligned and optimized for achieving its goals.

There are many different EA tools available on the market, each with its own set of features and capabilities. In this bachelor thesis, we will compare a selection of EA tools in order to understand their strengths and weaknesses and to identify which tool may be the most suitable for a given organization.

To compare the EA tools, we will first define a set of criteria that are relevant to the selection and evaluation of EA tools. These criteria may include factors such as the tool's level of support for various EA frameworks (e.g. TOGAF, Zachman), the tool's ability to integrate with other systems and tools, the tool's ability to support collaboration and stakeholder communication, and the tool's overall usability and user-friendliness.

Next, we will identify a selection of EA tools to be included in the comparison. This selection should be based on the popularity and prevalence of the tools in the market, as well as any unique features or capabilities that may set them apart from the competition.

Once the tools have been selected and the criteria defined, we will conduct a detailed comparison of the tools based on the defined criteria. This comparison should include an analysis of the tools' capabilities, limitations, and suitability for different types of organizations.

1.2 Research Objectives

The objectives of this thesis are:

1. To provide an understanding of the current state of the EAM tools market, and how it is evolving.
2. Analyzing the user interface and user experience of the tools, and determining how well they support the needs of development and operations teams in a DevOps environment.
3. Analyzing how well the tools integrate with other tools and systems used in a DevOps environment, such as continuous integration and delivery (CI/CD) tools, monitoring and logging tools.

1.3 Scope and Limitations

1.3.1 Scope

Focus and Selection of EAM Tools

Our study is specifically focused on assessing the usability and integration capability of selected EA tools only within the context of DevOps practices and scenarios.

Our research will examine a set of EA tools, based on specific criteria such as functionality available under different types of licenses, and the accessibility of REST APIs.

Evaluation Methods

The study will employ particular usability evaluation methods and integration capability assessment techniques, as detailed in the Methodology chapter, to systematically analyze and compare the selected EA tools.

1.3.2 Limitations

- The selected EA tools for this study do not encompass all available options in the market. As a result, our findings may not be universally applicable to every organization or scenario, but they should provide valuable insights into the key factors to consider when selecting EA tools for DevOps integration.
- The choice of EA tools in our research is limited to those with available REST APIs, as these APIs allow for integration with DevOps tools and practices. This limitation may exclude certain EA tools that lack REST APIs or have restricted access due to licensing issues.
- Our study is also limited by time and resource constraints, which could impact the depth of our research. Nevertheless, we aim to provide a comprehensive analysis of the selected EA tools within the given constraints.

1.4 Thesis Structure

- **Chapter 1** (this chapter) describes the problem, sets the objectives of this research, defines the scope and limitations.
- **Chapter 2** provides definitions of key terms and concepts related to Enterprise Architecture tools, DevOps practices, and their integration. Discusses the state of current research on the topic and establishes a theoretical foundation for the study.
- **Chapter 3** describes the research design, analysis methods, and the process of selecting and evaluating enterprise architecture tools in DevOps scenarios.
- **Chapter 4** presents an overview of the chosen Enterprise Architecture tools, based on the criteria and methods described in the Methodology chapter.
- In **Chapter 5**, based on the study's findings, recommendations for organizations considering EA tool integration in DevOps environments are provided, along with suggestions for future research directions.
- **Chapter 6** provides a summary of the research.

2 Theoretical Background

This chapter lays the foundation for understanding the research topic by defining key terms and concepts related to Enterprise Architecture (EA). It provides a brief overview of the current research landscape in the areas of EA, its frameworks, management, and tools.

The primary source of information and foundation for the majority of the theoretical concepts discussed is the book "Enterprise Architecture at Work" by Marc Lankhorst[17]. This work has been recognized as a comprehensive and authoritative resource on the subject of Enterprise Architecture Management (EAM), EAM frameworks, tools, and related topics.

The decision to rely heavily on this book as the primary reference is due to its well-established reputation and its extensive coverage of the fundamental principles, methodologies, and best practices in the field of Enterprise Architecture. By building upon the solid theoretical foundation provided by "Enterprise Architecture at Work," this thesis ensures a thorough understanding of the core concepts and their application in the context of EAM tools and their integration in DevOps scenarios.

2.1 Definitions

2.1.1 Enterprise Architecture

Building an organisation is a complex and challenging task because of the multifarious dependencies within an organisation. Many (often unknown) dependencies exist between various domains, like strategy, products and services, business processes, organisational structure, applications, information management and technical infrastructure. Besides having a good overview of these different domains, one needs to be aware of their interrelationships. Together, these form the enterprise architecture of the organisation[17].

To better understand the concept of Enterprise Architecture, it is helpful to break it down into its two constituent terms, "Enterprise" and "Architecture".

According to The Open Group 2011, an 'Enterprise' in this can be defined as follows: **Enterprise**: any collection of organisations that has a common set of goals and/or a single bottom line[17].

The ISO 2022 standard definition of Architecture is defined as follows: **Architecture**: fundamental concepts or properties of an entity in its environment and governing principles for the realization and evolution of this entity and its related life cycle processes[11].

Enterprise Architecture (EA) can be defined in various ways, reflecting the multifaceted nature of the concept. Here are some definitions that provide a comprehensive understanding of EA:

Enterprise architecture: a coherent whole of principles, methods, and models that are used in the design and realisation of an enterprise's organisational structure, business processes, information systems, and infrastructure[17].

Enterprise Architecture is the blueprint that documents all the information systems within the enterprise, their relationships, and how they interact to fulfil the enterprises mission[21, as cited in[16]].

A more of an IT-specific definition of EA: The Enterprise Architecture (EA) describes the interaction between Information Technology (IT) and the business processes in a company[15].

With a clear understanding of Enterprise Architecture, we will dive into more specific concepts related to EA, such as frameworks, management, and tools.

2.1.2 Enterprise Architecture Management

Enterprise Architecture Management (EAM) refers to the management practice that establishes, maintains and uses a coherent set of guidelines, architecture principles and governance regimes that provide direction and practical help in the design and development of an enterprise's architecture to achieve its vision and strategy[1]. EA management is a continuous, iterative (and self maintaining) process seeking to improve the alignment of business and IT in an (virtual) enterprise[3].

By effectively managing their Enterprise Architecture, organizations can ensure that their architecture remains aligned with their strategic goals, supports efficient operations, and enables effective decision-making, collaboration, and adaptability in the face of change[17]. The foundation for the practices involved in managing an organization's architecture are the architecture methods. An architecture method is a structured collection of techniques and process steps for creating and maintaining an enterprise architecture. Methods typically specify the various phases of an architecture's life cycle, what deliverables should be produced at each stage, and how they are verified or tested.

The further implementation of the architecture methods are the architecture frameworks, which will be discussed in the following section.

2.1.3 Enterprise Architecture Framework

Over the past decade a plurality of enterprise architecture management approaches with fairly different characteristics has been proposed by academia and practitioners. Some of these approaches can be described as frameworks[22].

An Enterprise Architecture Framework (EAF) is a structured set of guidelines, best practices, and methodologies used to organize, describe, and manage various aspects of an organization's architecture. Frameworks structure architecture description techniques by identifying and relating different architectural viewpoints and the modelling techniques associated with them[17]. EA frameworks play a crucial role in implementing and managing EA. These frameworks provide the necessary structure, guidelines, and methodologies to effectively develop and maintain an organization's architecture, ensuring alignment with its strategic goals and objectives.

There are several widely recognized and utilized EAFs in the field of Enterprise Architecture. In this section, we will briefly introduce and discuss three popular examples: TOGAF (The Open Group Architecture Framework), Zachman Framework, and FEAF (Federal Enterprise Architecture Framework). Each of these frameworks has its unique features and methodologies, showcasing the variety of approaches to organizing and managing an organization's architecture.

- **The Open Group Architecture Framework (TOGAF)**, the latest, 10th, version of which was launched on 25 April 2022[8] is a widely adopted EAF that

provides a comprehensive and iterative approach to the development and management of an organization's architecture. TOGAF is designed to support the entire architecture development lifecycle, with a strong emphasis on the alignment of business and IT strategies. It consists of two main components: the Architecture Development Method (ADM) and the TOGAF Content Framework. The ADM is a structured, step-by-step process for developing, implementing, and managing an organization's architecture. The TOGAF Content Framework defines a standardized set of artifacts, models, and viewpoints used to document and communicate the organization's architecture.

- **The Zachman Framework**, introduced by John A. Zachman in 1987[24], is considered one of the earliest and most influential Enterprise Architecture Frameworks. It presents a matrix-based approach to organizing and classifying architectural artifacts based on six perspectives (or rows) representing different stakeholders' viewpoints and six aspects (or columns) addressing different architectural dimensions (What, How, Where, Who, When, and Why). The Zachman Framework provides a comprehensive and systematic way to visualize, understand, and manage the complexity of an organization's architecture by breaking it down into manageable components and relationships.
- **The Federal Enterprise Architecture Framework (FEAF)** is a U.S. government-specific framework developed to guide the development and management of Enterprise Architecture across federal agencies[19]. FEAF provides a common approach, structure, and vocabulary for federal agencies to align their individual architectures with the broader federal government architecture. It comprises several components, including the Consolidated Reference Model (CRM) and the Federal Segment Architecture Methodology (FSAM). The CRM is a set of reference models that standardize the description and categorization of architecture components, while the FSAM is a step-by-step process for developing and managing segment architectures within the federal government context.

According to [23], most frameworks differentiate the following five EA layers:

- **Business architecture**: defines an organization's strategic objectives, organizational structure, and key business functions, ensuring alignment between business goals and other architectural layers.

- **Process architecture**: focuses on the organization's business processes, workflows, and activities, modeling how these processes interact and contribute to achieving the organization's objectives.
- **Integration architecture**: deals with the connections and interactions between various systems, components, and processes, ensuring seamless communication and data exchange across the organization.
- **Software architecture**: Software architecture encompasses the design and organization of software systems, applications, and components, addressing aspects such as modularity, scalability, and maintainability.
- **Technology (or infrastructure) architecture**: covers the underlying hardware, networks, and infrastructure components that support the organization's software systems, applications, and processes, addressing aspects such as performance, reliability, and security.

After discussing popular Enterprise Architecture Frameworks and their unique approaches to organizing and managing an organization's architecture, it is essential to examine another critical aspect of the Enterprise Architecture domain: Architecture Description Languages (ADLs). These languages provide a standardized notation for representing and communicating the various components, relationships, and structures of an organization's architecture. ADLs play a vital role in facilitating the understanding, documentation, and analysis of Enterprise Architecture, ultimately enabling more effective collaboration and decision-making among different stakeholders.

2.1.4 Architecture Description Languages

In this section, we will explore the concept of Architecture Description Languages, their purpose, and some examples of widely used ADLs in the field of Enterprise Architecture. By understanding the role and application of ADLs, we can gain further insights into the broader landscape of tools and techniques employed to develop, maintain, and manage an organization's architecture.

Architecture Description Languages (ADLs) are formal notations used to represent, document, and communicate the components, relationships, and structures of an organization's architecture. These languages facilitate the understanding, documentation, and analysis of Enterprise Architecture, ultimately enabling more effective collaboration and

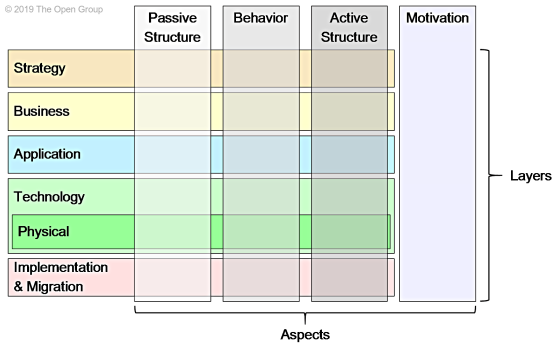

Figure 2.1: ArchiMate Full Framework

decision-making among different stakeholders. They are used to describe a software architecture in rather general terms. ADLs are essential for modeling the various architectural layers (business, data, application, and technology) and capturing their interdependencies in a standardized, consistent manner[4].

There are several widely used ADLs in the field of Enterprise Architecture, each with its unique notation and focus. Some popular examples include:

- **ArchiMate** is a graphical language developed by The Open Group that provides a standardized notation for describing, analyzing, and visualizing Enterprise Architecture. ArchiMate is designed to be aligned with the TOGAF framework and covers the business, application, and technology layers of architecture, as well as their relationships and dependencies[17].
- **Unified Modeling Language (UML)** is a general-purpose modeling language that can be used to represent and visualize various aspects of an organization's architecture, including its structure, behavior, and interactions. UML provides a rich set of diagrams, such as class diagrams, sequence diagrams, and activity diagrams, that can be used to model different architectural components and relationships[2].
- **Business Process Model and Notation (BPMN)** is a graphical notation specifically designed for modeling business processes. Although BPMN focuses primarily on the business layer of architecture, it can be used in conjunction with other ADLs to represent the relationships between business processes and other architectural components, such as applications and data. BPMN is restricted to process modelling[6].

2.1.5 Enterprise Architecture Management Tools

Enterprise Architecture Management tools support the activities and practices associated with managing an organization's architecture, including the creation, analysis, and maintenance of architectural artifacts, as well as the implementation of governance processes.

EAM Tools offer a range of functionality to help enterprise architects in the planning, analysis, design, implementation, and maintenance of their architecture. EAM tools serve the following purposes[17]:

1. EAM tools help to standardise the semantics and notation of architecture models. If the use of tools is accompanied by proper training and education, a company-wide introduction of a tool (or set of tools) is a big step towards standardisation of the architecture languages and practice within the organisation.

2. EAM tools can support the design of correct and consistent models through automated constraint checking and application of architecture principles.

3. EAM tools can support migration paths from the current situation to a newly designed 'to be' situation.

EA tools allow organizations to examine both the need for, and the impact of, change. They capture the interrelationships and interdependencies within and between an ecosystem of partners, operating models, capabilities, people, processes, information, and applications and technologies. They provide a central repository to capture data and metadata about the artifacts that an enterprise cares about and their related life cycles.[10]

Conceptually, tools are built for creating models in a specific modelling domain, and not for modelling relations to models or objects outside that domain.

Depending on the starting point for setting up tool support for enterprise architecture, a number of tool categories are of interest:

- Enterprise architecture modelling tools
- IT management tools
- Software design and development tools
- Business process design tools

- Business process management tools

For the purposes of this study, we will specifically focus on Enterprise architecture modeling tools, as they directly support the creation, analysis, and management of enterprise architecture models, which are integral to the scope of our research.

Features of EAM Tools

Gartner, a leading research and advisory firm, provides recommendations for features and capabilities that Enterprise Architecture Management (EAM) tools should include to meet the evolving needs of organizations. According to Gartner, an effective EAM tool should encompass the following key capabilities[10]:

- **Support for multiple EA Frameworks**: EAM tools should be able to support various Enterprise Architecture frameworks, such as TOGAF, Zachman, and FEAF, allowing organizations to adopt and customize the framework that best suits their needs and context.
- **Architectural Modeling**: EAM tools should offer robust modeling capabilities for creating, editing, and managing architectural artifacts using different Architecture Description Languages (ADLs), notations, and techniques.
- **Repository and Metadata Management**: EAM tools should include a centralized repository for storing, organizing, and managing architectural artifacts and metadata, enabling users to access, search, and reuse architectural information efficiently.
- **Collaboration and Communication**: EAM tools should facilitate effective collaboration and communication among architects, stakeholders, and other users through features such as version control, change tracking, commenting, and notifications.
- **Analysis and Decision Support**: EAM tools should provide various analytical capabilities to help users assess architecture, identify dependencies, risks, and opportunities for improvement, and evaluate the impact of potential changes. This may include gap analysis, impact analysis, scenario planning, and performance metrics.

- **Governance and Compliance**: EAM tools should support the establishment and enforcement of architectural governance, including processes, roles, and responsibilities for decision-making, communication, and coordination. Features may include managing architectural principles, standards, policies, and guidelines, as well as tracking and reporting compliance with these requirements.
- **Integration and Interoperability**: EAM tools should offer integration capabilities with other systems and tools within the organization, such as project management, IT service management, and software development tools, to enable seamless data exchange and ensure consistency across different tools and processes.
- **Reporting and Visualization**: EAM tools should provide various reporting and visualization features to help users create, customize, and share architectural views, diagrams, and dashboards, enabling stakeholders to gain insights into the organization's architecture and make informed decisions.
- **Scalability and Customization**: EAM tools should be scalable and customizable to support organizations of different sizes and complexity levels, with features for customization, extensibility, and configuration to meet specific organizational needs and preferences.
- **Cloud and SaaS Support**: As organizations increasingly adopt cloud-based solutions, EAM tools should offer cloud-based deployment options, including Software-as-a-Service (SaaS) models, to provide flexibility and scalability.

These key capabilities, as recommended by Gartner, ensure that EAM tools are well-equipped to support the diverse and evolving needs of organizations in managing their Enterprise Architecture effectively.

Meta Model of an EAM Tool

The meta model of an enterprise architecture management tool is a conceptual model that defines the structure and relationships of the elements that make up an organization's enterprise architecture, and serves as a blueprint for organizing and understanding the various components of the architecture, and helps to ensure consistency, completeness, alignment, and evolution[13].

A meta model is important for enterprise architecture management because it provides a common understanding and language for describing the architecture, and it helps to ensure consistency and completeness across the different layers and components of the architecture. The meta model can be used to define the types of artifacts that can be created and stored in the architecture repository, and it can also be used to define the relationships and dependencies between those artifacts. This helps to ensure that the architecture is well-aligned with the organization's goals and objectives, and that it is responsive to changing business needs.

The meta model is also important for managing the evolution of the architecture over time. As the organization's needs change, the meta model can be used to ensure that the architecture remains consistent and aligned with the organization's goals. It can also be used to identify areas of the architecture that may need to be modified or updated in order to support new business requirements.

Data Collection in an EAM Tool

There are several ways to report data to Enterprise Architecture Management tools, depending on the tool and the type of data being reported. Here are a few methods:

- **Manual Data Entry**: One of the simplest ways to report data to an EAM tool is to manually enter it into the tool using forms or data entry screens. This method is typically used for small amounts of data or for data that changes infrequently.
- **Data Import**: Many EAM tools include the ability to import data from external sources, such as spreadsheets, databases, or other tools. This method is typically used for larger amounts of data or for data that changes frequently. Some of the common import formats are CSV, Excel, JSON and XML.
- **Data Synchronization**: Some EAM tools can be configured to automatically synchronize data with external sources, such as systems of record, service management tools, or other EAM tools. This method is typically used for large amounts of data that changes frequently and where data is already stored in another system.
- **Data Collection and Discovery**: Some EAM tools include the ability to automatically discover and collect data from different systems and applications, such as servers, databases, and other IT assets. This method is typically used for large

amounts of data that is spread across multiple systems and applications, and that needs to be consolidated for reporting and analysis.

- **APIs**: Some EAM tools provide APIs (Application Programming Interface) that allow for the automation of data collection and updating. This method is particularly useful when data needs to be updated in real-time, or when a large number of updates need to be made, such as when a system is integrated with other systems.

For the purpose of integrating an EAM Tool into DevOps it is preferable to use completely automated method of collecting data, such as via API or Data Synchronization.

2.1.6 DevOps

DevOps is a set of practices that automates the processes between software development and IT teams, in order that they can build, test, and release software faster and more reliably. The concept of DevOps is founded on building a culture of collaboration between teams that historically functioned in relative silos. The promised benefits include increased trust, faster software releases, ability to solve critical issues quickly, and better manage unplanned work[5].

DevOps practices include agile methodologies, continuous integration and delivery (CI/CD), continuous testing, and infrastructure as code (IAC). Agile methodologies focus on rapid iteration and delivery of software, while CI/CD automates the process of building, testing, and deploying code. IAC allows for the provisioning and management of infrastructure through code rather than manual configuration, which enables more efficient and consistent management of infrastructure[14].

2.2 Usage of Enterprise Architecture Tools in DevOps Scenarios

IT-landscape modelling, as a sub-area of EAM, tries to discover and document the IT-landscape of an organization. It aims to generate and maintain a virtual representation – a digital twin – of the whole organization[20]. With this digital twin, it becomes much more convenient for the enterprise architects to assess the as-is EA. This is important for decision support especially when it comes to IT transformation and modernization.

Many EAM tools support the modelling of the EA digital twin, however, the creation of these models is a time-consuming task[22]. Enterprise architects struggle to collect EA-relevant information to unveil the current business- and IT-landscape[9]. This leads to out-of-date EA models and, in turn, to decisions made on wrong or obsolete data basis[18].

A proposed solution to this problem is the maintanence of the architecture model via runtime IT discovery[15]. The proposed tool automates the EA model maintenance by implementing a new integration layer that synchronizes static and runtime data from different data sources.

In this study we further investigate the possibility of automated EA Model maintanance in relation to specific tools and provide an overview to what extent it is possible to accomplish with each of them.

3 Methodology

3.1 Research Design

In this this study, we focus on the practical aspects of using Enterprise Architecture Management tools in DevOps scenarios. The primary goal is to explore the capabilities of EAM tools' interfaces and understand how they can be leveraged to access essential information and maintain up-to-date EA models automatically via their APIs.

To achieve this goal, the research will follow the following methods:

- **Interface analysis**: A systematic analysis of the EAM tools' interfaces will be conducted to identify the available features and assess their suitability for integration in DevOps scenarios. This analysis will focus on the APIs provided by the EAM tools, exploring their functionality, ease of use, and flexibility.
- **Documentation review**: The study will involve a review of the EAM tools' documentation, focusing on the technical aspects of their APIs, including data models, request and response formats, authentication mechanisms, and any limitations or constraints that may impact their use in DevOps contexts.
- **Example API Requests**: We will provide some sample requests to the Tools' APIs. This will demonstrate the practical usage for typical tasks that a user or developer might perform while interacting with the EAM for the sake if integrating it into DevOps.

This approach will help us to understand the EAM tools' usability, integration capabilities, and the challenges and benefits associated with maintaining EA models through their APIs.

3.2 Interface Analysis

Interface analysis is the first method employed in this study to evaluate the usability and integration capabilities of EAM tools in DevOps scenarios. This method involves a systematic examination of the EAM tools' interfaces, particularly focusing on their APIs, to gain insights into their features, functionality, and ease of use. The primary objective of the interface analysis is to assess how well the EAM tools can support the effective implementation of DevOps practices and maintain up-to-date EA models through their APIs.

3.2.1 Objectives of Interface Analysis

The main objectives of the interface analysis in this study include:

1. Identifying the key features and functions available through the EAM tools' APIs.
2. Evaluating the ease of use, flexibility, and customization options offered by the APIs.
3. Assessing the compatibility and alignment of the EAM tools' APIs with DevOps principles and practices.

3.2.2 Execution Steps

The interface analysis will be conducted through the following steps:

1. **Exploration of the APIs**: The research of the APIs provided by the selected EAM tools, focusing on their capabilities, documentation, and any available sample code or examples.
2. **Description of Meta Model**: we will describe the meta model of each EAM Tool to provide readers with a better understanding of the underlying structure and concepts of each tool.
3. **Features and Functionality**: We will describe the features and functionality available via the tool's REST API.

4. **Ease of Use**: We will assess the usability of the EAM tools' APIs by examining the comprehensiveness and clarity of their documentation and available resources such as sample code or example requests. This analysis will help determine how easily developers can interact with and integrate the APIs.

5. **DevOps Integration**: We will evaluate the APIs' compatibility and alignment with DevOps practices by identifying relevant features and functionalities.

6. **Summary**: We will synthesize the key findings from the previous sections, highlighting the strengths and weaknesses of the EAM tools' APIs in terms of usability and integration capabilities within DevOps scenarios.

3.3 Documentation review

The documentation review is the second method employed in this study to assess the usability and integration capabilities of the selected EAM tools in DevOps scenarios. This method involves an examination of the tools' official documentation, including user guides, technical manuals, and online resources, focusing on the aspects that are most relevant to their use in DevOps contexts. The primary objective of the documentation review is to gain a deeper understanding of the EAM tools' features, functionality, and any limitations or constraints that may impact their use in DevOps environments.

3.3.1 Objectives of Documentation Review

The main objectives of the documentation review in this study include:

1. Identifying the key features and functionality offered by the EAM tools.

2. Evaluating the ease of use, customizability, and extensibility of the EAM tools based on the information provided in their documentation.

3. Assessing the ability to complete a scenario, such as authenticating via the REST API using the documentation.

3.3.2 Execution Steps

The documentation review will be conducted through the following steps:

1. **Accessing documentation**: We will locate and access the official documentation for the selected EAM tools, including user guides, technical manuals, and online resources such as help centers or knowledge bases.

2. **Following a scenario**: During documentation review, we will follow a scenario, as to what we want to accomplish using the documentation. By focusing on specific objectives and tasks we aim to accomplish using the documentation, we systematically evaluate the information provided, its organization, and the effectiveness of the guidance.

3. **Evaluating the documentation**: We will assess the quality and clarity of the documentation, considering factors such as the depth and accuracy of the information provided, the organization of the content, and the availability of examples and sample code.

By conducting a thorough documentation review, this study aims to provide valuable insights into the practical aspects of using EAM tools in DevOps contexts, thereby contributing to the overall understanding of their potential benefits and challenges.

3.4 Example API Requests

In the literature there has been proposed a method of automated maintenance of Enterprise Architecture Model via Runtime IT Discovery[15]. This method, which we can refer to as the "Automated EA Model Maintenance Method," aims to leverage runtime service instrumentation of existing IT architecture to automatically create, update, and enhance static EA models with runtime information.

In the Example API Requests method, we aim to demonstrate practical use cases of the selected EAM tools' APIs by providing sample requests for typical tasks that a user or developer might perform in the context of integrating EAM tools within DevOps scenarios. This method helps to further illustrate the capabilities, ease of use, and integration potential of the EAM tools' APIs and to provide readers with a better understanding of

how they can be utilized effectively in the siituation of Automated EA Model Maintenance.

3.4.1 Objectives of Example API Requests

The main objectives of this method in this study include:

1. Illustrating practical applications of the EAM tools' APIs by providing examples of common tasks and operations.
2. Demonstrating the ease of use and integration potential of the APIs through real-world scenarios.
3. Offering readers a clear understanding of how the APIs can be used to perform specific tasks, supporting their integration with other tools and platforms in a DevOps environment.

3.4.2 Execution Steps

The Automated EA Model Maintenance Example Implementation will be conducted through the following steps:

1. **Choose appropriate API endpoints**: For each identified task, we will select the appropriate API endpoints to demonstrate how the desired actions can be achieved using the EAM tools' APIs.
2. **Provide clear explanations**: Alongside each example request, we will offer a clear explanation of the request's purpose and functionality, helping readers understand the steps involved in using the APIs to accomplish specific tasks.

3.5 Tool Selection

3.5.1 Overview of Existing Tools

There are several Enterprise Architecture Management (EAM) tools currently available in the market. This section presents a brief summary of some of the leading EAM

tools available in the market, as identified in the Gartner Magic Quadrant for Enterprise Architecture Tools report[12]. Here are a few examples:

- **LeanIX** is a modern, cloud-based EAM tool that offers a comprehensive set of features for managing enterprise architecture. It is designed to support a wide range of use cases, including IT portfolio management, technology risk management, and DevOps integration. LeanIX's REST API allows for easy integration with other tools and systems, which makes it a suitable candidate for DevOps scenarios. The tool has gained popularity among organizations of various sizes and industries for its user-friendly interface, flexibility, and extensive reporting capabilities.

- **ADOIT** is an established EAM tool that provides a powerful suite of features for modeling, analyzing, and optimizing enterprise architecture. It supports a wide variety of industry-standard frameworks, such as TOGAF, ArchiMate, and ITIL, making it a popular choice among organizations seeking a robust and comprehensive EAM solution. ADOIT's REST API enables seamless integration with other tools, systems, and platforms, facilitating its use in DevOps contexts. The tool is known for its advanced visualization capabilities, intuitive user interface, and customizable reporting features.

- **LUY** is a relatively newer entrant to the EAM market, offering a modern and innovative approach to enterprise architecture management. It emphasizes ease of use, collaboration, and scalability, making it an attractive option for organizations of all sizes. LUY's REST API allows for straightforward integration with other DevOps tools, systems, and processes, supporting the maintenance and updating of architecture models in real-time. The tool offers a unique combination of simplicity and depth, with a focus on enabling cross-functional collaboration and effective decision-making.

- **Avolution ABACUS** is a flexible and scalable EAM tool that supports multiple frameworks, such as TOGAF, ITIL, and COBIT. It provides a range of features, including enterprise architecture modeling, IT portfolio management, and analytics capabilities. ABACUS is known for its advanced algorithms and calculations, enabling organizations to assess the impact of various architectural changes and decisions. Its REST API allows for straightforward integration with other tools and systems, making it suitable for use in DevOps environments.

3.5.2 Choice of EAM Tools for this study

The selection of EAM tools for this study was primarily influenced by the availability of licenses and the potential for academic collaboration. **LeanIX**, **ADOIT**, and **LUY** were chosen as the focus of this research due to their willingness to provide university and community licenses, as well as test accounts with full access. These tools represent a diverse range of solutions in the EAM landscape, offering valuable insights into their usability and integration capabilities in DevOps scenarios.

- **LeanIX** was selected for this study because of its well-established presence in the EAM market and its provision of a university license for academic research purposes. The tool's modern, cloud-based approach, user-friendly interface, and extensive feature set make it a candidate for evaluating usability and integration in DevOps contexts.
- **ADOIT** was chosen for this research due to its robust and comprehensive suite of features, as well as its availability through a Community Edition license. The tool's support for various industry-standard frameworks and its advanced visualization capabilities make it a good candidate for assessing the potential of EAM tools in DevOps scenarios.
- **LUY** was included in this study because of its innovative and modern approach to enterprise architecture management and its provision of a test account with full EAM access for academic research. Its focus on simplicity, scalability, and cross-functional collaboration make it an interesting option for evaluating the integration and usability of EAM tools in DevOps environments.

The selection of these three tools allows for a well-rounded exploration of the EAM landscape. By analyzing their interfaces, reviewing their documentation, and interacting with their APIs using a simulation program, we hope to provide valuable insights into their potential in DevOps scenarios.

In the following chapter, each of these EAM tools will be thoroughly examined through the application of the chosen research methods: interface analysis, documentation review, and example API requests will be provided. The insights gained from this evaluation will provide an understanding of the tools' strengths, weaknesses, and overall suitability for integration with DevOps processes.

4 Evaluation

The primary objective of this chapter is to assess the usability and integration capabilities of the selected EAM tools - LeanIX, ADOIT, and LUY - within the context of DevOps scenarios. Through a combination of interface analysis and documentation review, this chapter aims to provide a comprehensive understanding of each tool's strengths, weaknesses, and overall suitability for integration with DevOps processes.

The insights gained from this evaluation will serve as the basis for identifying best practices and providing recommendations for organizations looking to effectively leverage EAM tools in their DevOps environments. We will systematically evaluate each of the selected EAM tools following the methodology described in the previous chapter. After that, a comparative analysis will be conducted, drawing attention to the respective strengths and weaknesses of the EAM tools within the context of DevOps scenarios.

4.1 LeanIX Analysis

4.1.1 LeanIX Interface Analysis

LeanIX API Description

LeanIX provides a comprehensive RESTful API that allows developers to interact with the tool programmatically[1]. This API provides access to various functionalities, such as creating, updating, and retrieving information related to EA models, including applications, interfaces, data objects, and more.

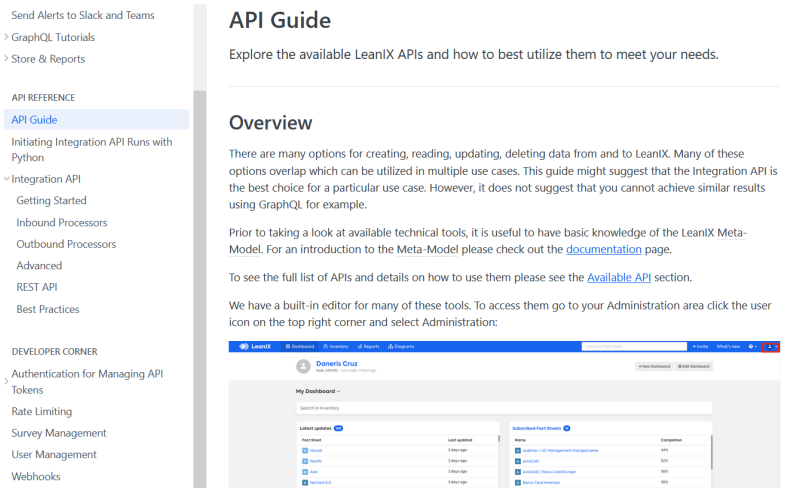

Figure 4.1: Main page of LeanIX API Guide

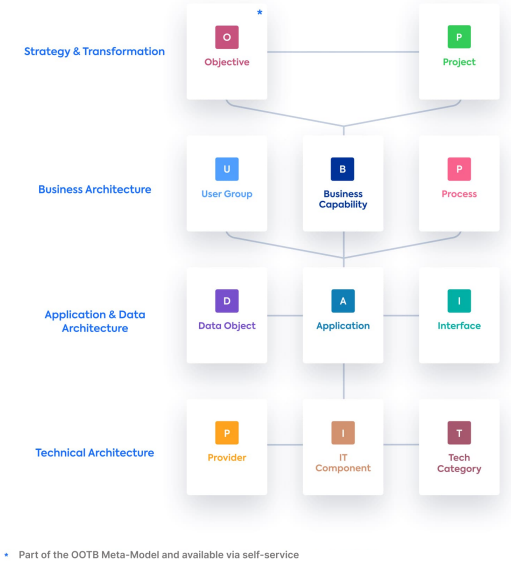

Figure 4.2: Meta Model of LeanIX

LeanIX Meta Model

The meta-model of LeanIX[2] is a collection of different entities or "fact sheets" that represent the different components of the organization's IT landscape, such as applications, systems, services, and infrastructure, as well as the relationships between them.

The meta-model of LeanIX is based on the ArchiMate standard, which is an open standard for enterprise architecture modeling that provides a common language and framework for modeling and analyzing enterprise architecture. ArchiMate is based on three different layers: the Business layer, the Application layer, and the Technology layer.

The Business layer represents the business processes and functions of the organization, the Application layer represents the applications and systems used to support the business processes, and the Technology layer represents the infrastructure and technology that supports the applications.

Each "fact sheet" in LeanIX's meta-model is based on one of the ArchiMate elements and has a predefined set of attributes and relationships to other fact sheets. For example, an "Application" fact sheet would have attributes such as name, description, and owner, and it could be related to "Business Processes", "Services", "Systems" and more. This allows for a clear representation of the IT landscape and the relationships between different components.

LeanIX also allows for custom attributes and relationships to be added to the fact sheets, which helps in capturing additional information and context that is specific to the organization.

Features and Functionality

The LeanIX API offers a wide range of features and functionalities, including:

- CRUD Operations: The API supports all the basic Create, Read, Update, and Delete (CRUD) operations for managing EA model elements, such as applications, interfaces, and data objects.

[1] https://docs-eam.leanix.net/reference/integration-api accessed on 2023-03-15
[2] https://docs-eam.leanix.net/docs/meta-model accessed on 2023-03-15

- Data Model Customization: The API allows users to customize their data model by adding custom attributes and relationships, enhancing the ability to tailor the tool to specific organizational needs.
- Versioning and History: The API supports versioning and history tracking, enabling users to view and manage different versions of the EA model and their changes over time.
- Access Control and Authentication: The API supports robust access control and authentication mechanisms, ensuring data security and providing granular control over data access.
- Webhooks: The API supports webhooks, enabling real-time notifications and integration with other tools and platforms.

Ease of Use

The LeanIX API is well-documented, with comprehensive API documentation and examples available on the LeanIX website. This makes it easy for developers to understand and interact with the API. Additionally, the API follows RESTful design principles, which promotes consistency and simplicity, making it easier for developers to integrate the tool with other platforms and services.

DevOps Integration

The LeanIX API's features and functionality facilitate seamless integration with various DevOps tools and practices. For example, webhooks enable real-time updates and notifications, while the API's support for CRUD operations allows for easy synchronization of data between LeanIX and other tools, such as CI/CD pipelines, configuration management systems, and monitoring platforms. This integration capability makes LeanIX a suitable choice for organizations aiming to implement DevOps practices while maintaining an up-to-date and accurate EA model.

Summary

The interface analysis of the LeanIX EAM tool reveals that its API offers a comprehensive set of features and functionalities, promoting usability and integration capabilities in DevOps scenarios. Its ease of use, supported by well-documented resources and RESTful design principles, facilitates seamless integration with other tools and platforms in a DevOps environment. Overall, the LeanIX API demonstrates strong potential to support effective implementation of DevOps practices while maintaining up-to-date EA models.

4.1.2 LeanIX Documentation Review

In this section, we present a documentation review of the LeanIX EAM tool to assess its usability and integration capabilities in DevOps scenarios. This review involves examining LeanIX's official documentation, including user guides, technical manuals, and online resources, focusing on aspects most relevant to DevOps contexts. Our primary objective is to gain a deeper understanding of the LeanIX's features, functionality, and any limitations or constraints that may impact its use in DevOps environments.

Accessing Documentation

We accessed the official LeanIX documentation from the LeanIX website[3], which offers various resources such as user guides, technical manuals, and online help centers. Our primary focus was accessing and use of LeanIX Integration API, so we followed a scenario in which we, as a LeanIX user, wanted to authenticate and send requests to the API programmatically. During this scenario we encountered a number of problems, which we will describe in this section.

Evaluating the documentation

First of all, we tried to accesss the documentation from the main page of the workspace. In the documentation window we could search for the API Guide that we needed. However, we could not follow the documentation in accessing the Integration API from our account, because the option was just not there. The documentation or interface failed

[3] https://docs.leanix.net/ accessed on 2023-03-15

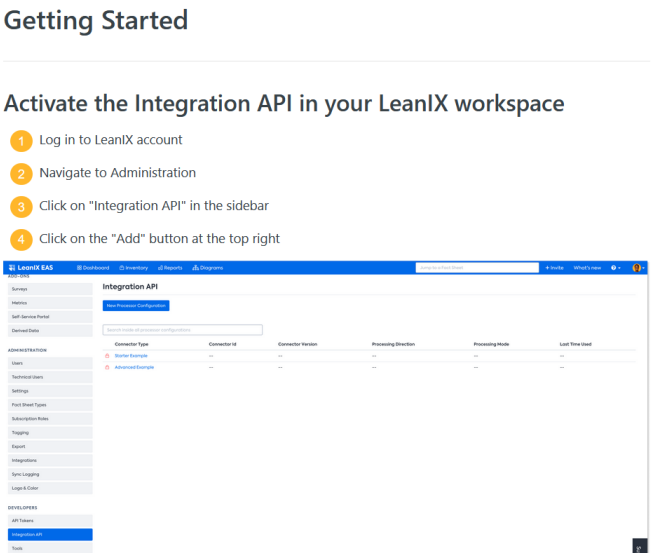

Figure 4.3: Instructions to activate the LeanIX Integration API

to provide the information about the missing access to some features depending on the license used by the user. We could only figure that out by contacting the support of LeanIX. The same with additional Developer Tools, such as GraphQL.

Nevertheless, we proceeded to examine the documentation for utilizing the REST API and observed that it provided a clear explanation of user authorization and request submission. The documentation featured a lot of example requests, tutorials, and included a Swagger description of the API. We encountered a few broken URLs within the text, which redirected to empty pages. However, by using the search functionality, we were able to locate the intended referenced pages.

LeanIX defines its own LeanIX Data Interchange Format (LDIF) for the API requests. The primary purpose of LDIF is to provide a standardized, structured, and consistent format for data import and export when interacting with the LeanIX API.

LDIF is based on JSON, a widely-used, lightweight data interchange format that is easy for humans to read and write and easy for machines to parse and generate. By leveraging JSON, LDIF ensures compatibility and interoperability with various programming languages, platforms, and tools.

In the context of LeanIX API requests, LDIF defines the schema and structure for the data payload, including the attributes, relationships, and metadata associated with Enterprise Architecture model elements such as applications, services, infrastructure com-

ponents, and more. This enables developers to work with a consistent data representation while interacting with the LeanIX API, making it easier to integrate the LeanIX EAM tool into DevOps scenarios and other systems.

4.1.3 LeanIX Example API Requests

For the authentication with the LeanIX Integration API, the API token is needed. It is possible to generate an API token from the LeanIX workspace administration area.

Next, we can construct an HTTP GET request to query the LeanIX Integration API, specifying the desired Fact Sheet type. In this example, we will retrieve all "Application" Fact Sheets: Here's an example request to the LeanIX Integration API using cURL:

```
curl -X GET \
"https://<LeanIX-instance-URL>/services/pathfinder/v1/factsheets?type=Applicat
  -H "Authorization: Bearer <api-token>" \
  -H "Content-Type: application/json"
```

In this example, <LeanIX-instance-URL> should be replaced with the URL of the LeanIX Instance and <api-token> with the token generated from the workspace administration area.

If the request is successful, the API will return a JSON response containing a list of "Application" Fact Sheets and their associated data.

Another example of API request would be updating a fact sheet:

```
curl -X PUT \
"https://<LeanIX-instance-URL>/services/pathfinder/v1/factsheets/<id>" \
  -H "Authorization: Bearer <api-token>" \
  -H "Content-Type: application/json" \
  -d '{
    "displayName": "New Application Name",
    "tags": ["newTag"]
  }'
```

In addition to the variables mentioned in the previous example, the <id> must be replaced with the ID of the fact sheet that we want to update.

This request specifically updates the displayName and tags of a fact sheet. It is also possible to update other attributes of a fact sheet. Full description of the attributes that can be updated is located in the Swagger UI[4].

4.2 ADOIT Analysis

4.2.1 ADOIT Interface Analysis

In this section, we provide an in-depth interface analysis of the ADOIT EAM tool[5], focusing on its Application Programming Interface (API) to evaluate its usability and integration capabilities in the context of DevOps scenarios. ADOIT is an Enterprise Architecture Management tool developed by BOC Group. We will assess its API features, functionality, and ease of use to understand its suitability for DevOps environments.

ADOIT Meta Model

The meta-model of ADOIT is based on the ArchiMate modeling language and conforms to the TOGAF framework, ensuring alignment with widely recognized industry standards. However, it also extends beyond these standards, allowing for customization and integration with other frameworks as needed.

User can choose between several available predefined metamodel profiles, reducing the number of available elements within the chosen metamodel[6]. The complete profile is called ADOIT for ArchiMate Champions and contains all object types and relationships of the ArchiMate framework[7].

The ADOIT meta-model is organized into several layers, which represent different aspects of the enterprise architecture:

[4] https://eu.leanix.net/openapi-explorer/#/factSheets/updateFactSheet accessed on 2023-03-15
[5] https://developer.boc-group.com/adoxx/en/API-Guide/ accessed on 2023-03-15
[6] https://docs.boc-group.com/adoit/en/docs/16.0/user_manual/mmconf-000000/ accessed on 2023-03-15

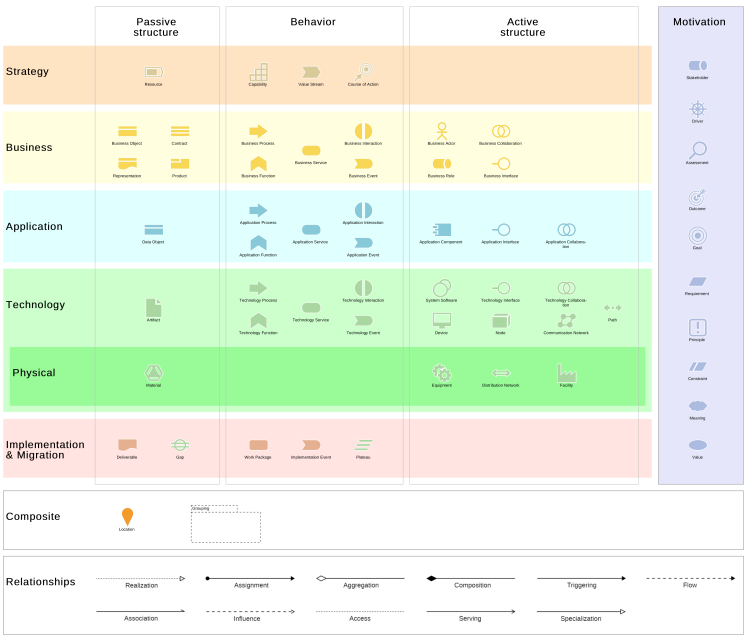

Figure 4.4: Meta Model of ADOIT (for ArchiMate Champions profile)

- **Business Layer**: This layer focuses on business processes, organizational structures, and functions. It includes elements such as business actors, roles, processes, services, and products.
- **Application Layer**: This layer describes the applications, services, and interfaces that support the business layer. It includes components like application components, services, interfaces, and data objects.
- **Technology Layer**: This layer represents the underlying technology infrastructure, including hardware, software, and networks. It encompasses elements like devices, system software, network components, and communication protocols. It also contains a physical sub-layer.
- **Implementation and Migration Layer**: This layer focuses on the management of projects, programs, and portfolio elements, which are necessary to transition from the current state to the desired future state of the architecture. It includes components like projects, programs, and migration plans.

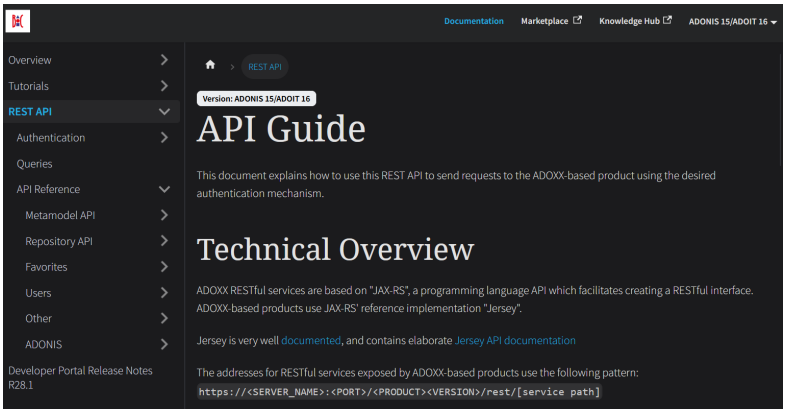

Figure 4.5: Main page of ADOIT API Guide

- **Motivation Layer**: This layer captures the strategic drivers, goals, objectives, and requirements that underpin the enterprise architecture. It consists of elements like drivers, goals, objectives, requirements, principles, and constraints.
- **Customization Layer**: ADOIT also supports the customization of the meta-model, allowing organizations to add their own elements and relationships to accommodate their specific needs and preferences.

The ADOIT meta-model captures the relationships between these elements across layers, enabling a holistic view of the enterprise architecture. By providing a comprehensive representation of architectural components and their interdependencies, the ADOIT meta-model supports effective decision-making, planning, and analysis in EAM contexts.

Features and Functionality

The ADOIT API provides the following features and functionalities:

- CRUD Operations: The API supports basic Create, Read, Update, and Delete (CRUD) operations for managing EA model elements.
- Data Model Flexibility: ADOIT allows for a flexible data model that can be adapted to specific organizational needs.
- Access Control and Authentication: The API supports several ways of authentication, ensuring data security and providing granular control over data access.

Ease of Use

The ADOIT API is well-documented, with detailed API documentation available on the BOC Group website. The documentation provides examples and sample code snippets, making it easy for developers to understand and interact with the API. The RESTful design principles of the API ensure consistency and simplicity, facilitating integration with other platforms and services.

Unfortunately, the "Try it!" function is missing from Swagger UI of the ADOIT, but they promise to add the functionality in future releases.

DevOps Integration

The ADOIT API's features and functionality enable seamless integration with various DevOps tools and practices. The API's support for CRUD operations allows for easy synchronization of data between ADOIT and other tools, such as CI/CD pipelines and configuration management systems. Additionally, the API's integration capabilities make ADOIT a suitable choice for organizations looking to implement DevOps practices while maintaining an up-to-date and accurate EA model.

Summary

The interface analysis of the ADOIT EAM tool reveals that its API offers a comprehensive set of features and functionalities, promoting usability and integration capabilities in DevOps scenarios. Its ease of use, supported by well-documented resources and RESTful design principles, facilitates seamless integration with other tools and platforms in a DevOps environment. Overall, the ADOIT API demonstrates strong potential to support the effective implementation of DevOps practices while maintaining up-to-date EA models.

4.2.2 ADOIT Documentation Review

In this section, we present the documentation review of the ADOIT EAM tool to assess its usability and integration capabilities in DevOps scenarios. We will examine the tool's

official documentation, including user guides, technical manuals, and online resources, focusing on aspects relevant to their use in DevOps contexts.

Accessing Documentation

The official documentation for ADOIT is available on the BOC Group website. It's accessible from the main page on the ADOIT workspace. This documentation includes user guides, technical manuals, and online resources such as help centers or knowledge bases. The documentation is accessible for registered users, providing a comprehensive source of information on the tool's features, functionality, and integration capabilities.

Evaluating the Documentation

During this review, we followed the same scenario as we did with evaluating the LeanIX documentation, which is accessing and connecting the ADOIT REST API to our program.

ADOIT documentation is divided into "User Manual" and "Administrator Manuals". It is done to cater to different roles and responsibilities within an organization that uses the ADOIT EAM tool. The primary difference between these two types of manuals lies in their target audience and the scope of the content they cover is that the User Manual is designed for end-users, architects, and analysts who will be using the ADOIT EAM tool on a day-to-day basis. This manual covers topics related to navigating the tool, understanding its features and functionalities, and using it to model, analyze, and manage enterprise architecture artifacts. It generally includes step-by-step guides, use case examples, and best practices for working with the tool in various scenarios. The Administrator Manuals are targeted towards system administrators, IT managers, and other technical personnel responsible for the installation, configuration, maintenance, and overall management of the ADOIT EAM tool within an organization. This set of manuals focuses on topics such as system requirements, installation procedures, configuration settings, user and access management, customization, and troubleshooting. The content in the Administrator Manuals often covers more advanced and technical aspects of the tool, ensuring that it operates effectively within the organization's IT environment.

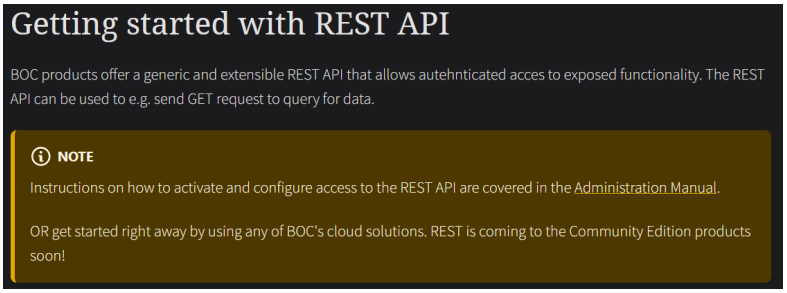

Figure 4.6: ADOIT Availability of REST for Community Edition

The information about REST API is covered in the BOC Developer Portal[7], which is common documentation for three BOC products: ADONIS, ADOIT and ADOGRC.

The API Documentation is divided into different sections and provides clear examples on commands written in Java language. Unlike the LeanIX documentation, the information about accessibility of different features for specific licenses is provided where needed.

4.2.3 ADOIT Example API Requests

In addition to the Developer Portal, the BOC Group provides a Github repository with a lot of example requests to access the API[8].

There are currently 4 ways to authenticate against the API: Token-Based Authentication, Basic Authentication, Oauth 2.0 and JWT authentication. For this example, we will use the Basic authentication method.

First, the Basic authentication must be enabled in the administrator settings area.

Example cURL request to retrieve all available repositories:

```
curl -X GET "https://<adoit-base-url>/rest/3.0/repos/" \
-u <username>:<password>
-H "Content-Type: application/json
```

The following variables must be replaced for the request: <adoit-base-url> with the URL of the ADOIT instance, <username> and <password> with the user data for the authentication.

[7] https://developer.boc-group.com/adoxx/en/ accessed on 2023-03-15
[8] https://github.com/BOC-Group/developer-examples accessed on 2023-03-15

Example cURL request to create new repository objects:

curl

```
curl -X POST "https://<adoit-base-url>/rest/3.0/repos/<repo-id>/objects" \
-u <username>:<password> \
-H "Content-Type: application/json" \
-d '{
  "attributes": [
    {
      "metaName": "A_DESCRIPTION",
      "value": "Description value to be set"
    },
    {
      "metaName": "A_SHOW_SYMBOL_IF_COTROLS_EXIST",
      "value": true
    }
  ],
  "groupId": "{227f1a76-dd6f-4c03-aed9-36e07e82a14c}",
  "metaName": "C_RISK",
  "name": "New Risk"
}'
```

In addition to the variables for the previous request, the variable <repo-id> must be replaced with the UUID of the repository, where the new objects will be created.

Full spec of the Objects API can be found in the Swagger UI of BOC products[9].

[9] https://developer.boc-group.com/adoxx/en/rest-objects/#/RepositorywriteAPIs/createObject accessed on 2023-03-15

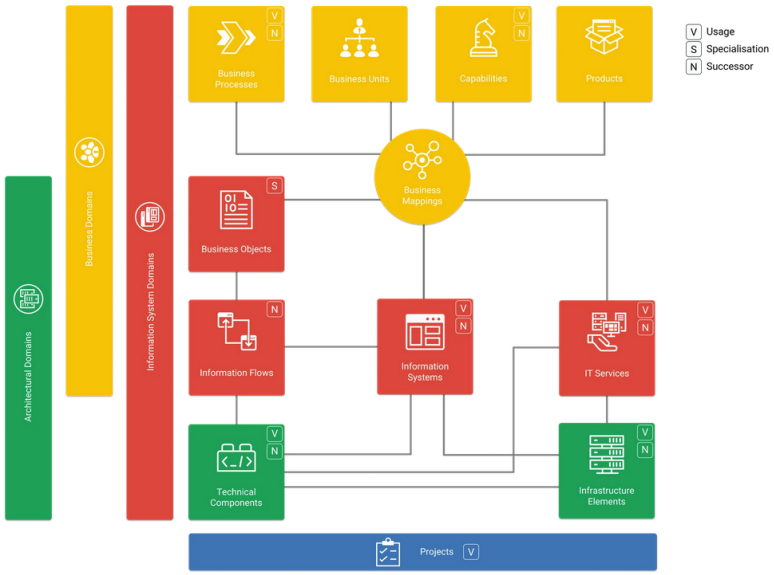

Figure 4.7: LUY Meta Model

4.3 LUY Analysis

4.3.1 LUY Interface Analysis

LUY Meta Model

The description of the meta model of LUY is available in the official LUY Documentation on their website[10].

The LUY meta model consists of the following customizable building block types:

- Business domain: A structural element that serves to group associated building blocks in the business landscape. A business domain describes the core business units within the company.
- Business process: A business process is a sequence of logically related activities contributing to value creation for the company. It has a predefined beginning and an end, is usually performed repeatedly, and is oriented toward the customer.
- Capabilities: A self-contained and coherent business activity, e.g., create customer or change product configuration.

[10] https://help.luy.app/doc/7.3/cloud/meta-model accessed on 2023-03-15

- Product: The outcome or deliverable of an enterprise's service or manufacturing process. Products can be either tangible (e.g., goods such as cars or computers) or intangible (services).
- Business unit: A logical or structural unit of an enterprise, such as a department, site or plant; it also encompasses user groups such as "field sales team" or "internal administration".
- Business object: A real entity - abstract or concrete - that is closely related to an organization's business activities (e.g., customer, product, or task). Business objects can be connected by relationships and are used by business processes and capabilities.
- Information system: A software or software package for associated functionalities which is logically and technically distinct from other areas of functionality and which can be supported entirely or to a large extent by IT.
- Information system domain: A group of Information Systems with common features. IS Domains are commonly used to organize the IS landscape and the responsibilities for landscape planning into related units.
- Information flow: The exchange of business objects between two information systems. The exchange can be directed. Technical components can be assigned to implement the exchange. The endpoint of the information flow on the information system can be called the interface.
- IT service: A service provision of IT to its users to perform its tasks (capabilites). The IT service is usually a bundle of services and offers of IT (information systems, technical devices, infrastructure) and is described by an agreement.
- Architectural domain: The structuring of the blueprint, the standardization catalogue for technical landscape. An architectural domain might be used to group technical components.
- Technical component: Information about the technical realization of information systems or information flows. Standardization is part of IT architecture management. Its result is a catalog of standardized technical components, also called "technical blueprint". In addition to standardized technical components, non-standardized technical components can also be managed as part of the documentation of the current status quo.

- Infrastructure element: Contains the logical hardware and network units on which information systems run. Infrastructure elements are generally either captured at the rough logical level or imported by a CMDB. A CMDB provides all backed-up and up-to-date information about the configuration items of the IT infrastructure (applications, clients, network, server, storage) and their relation to each other as well as the basic data for supporting the service management processes.
- Project: A purposeful activity or undertaking that is essentially characterized by the uniqueness of the conditions in its entirety, e.g., target setting, time, financial, personnel and other limitations, differentiation from other projects, and project-specific organization.

LUY REST API

The REST API of LUY is described in their official documentation [11]. The documentation section contains information on the authentication. For the complete documentation of all REST Endpoints Swagger is used. The Swagger documentation provided by the LUY EAM tool is more extensive and detailed compared to LeanIX and ADOIT. This enhanced documentation enables users to not only understand the structure and functionality of the API endpoints but also to execute example API calls directly from the documentation.

Features and Functionality

The LUY REST API supports standard Create, Read, Update, and Delete (CRUD) operations on both building blocks and the entire data model. This functionality is critical for effectively managing and updating enterprise architecture models, ensuring accurate and up-to-date information is available for decision-making and analysis.

In addition to supporting CRUD operations, the LUY REST API provides access to historical data associated with building blocks, if the feature is enabled. This feature enables users to track changes and monitor the evolution of the enterprise architecture model over time, providing valuable insights for strategic planning and continuous improvement initiatives.

[11] https://help.luy.app/doc/7.3/cloud/rest-api accessed on 2023-03-15

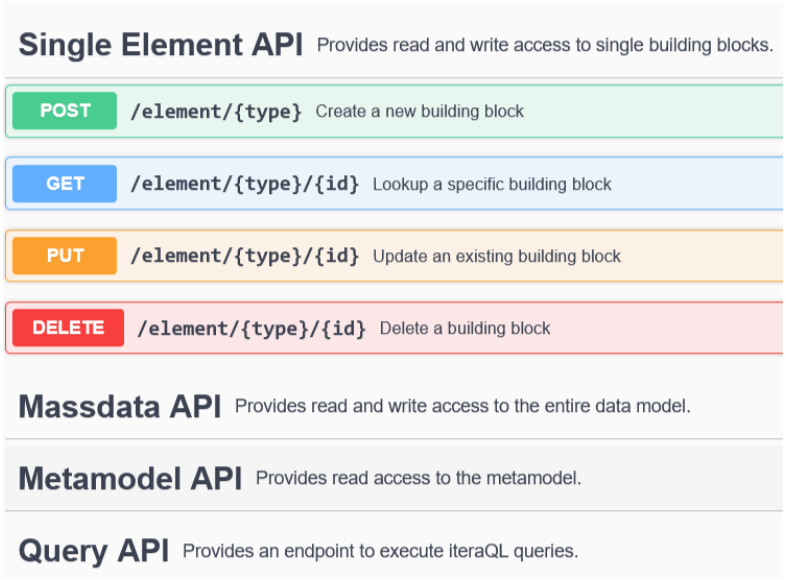

Figure 4.8: Swagger description of LUY REST endpoints

The LUY REST API also offers capabilities for managing users, assigning roles, and performing other administrative tasks. This functionality is particularly important for organizations that require robust user management and access control mechanisms to ensure the appropriate individuals have access to the enterprise architecture model and its associated data.

Ease of Use

Our analysis revealed that the LUY REST API documentation is notably concise. While it provides a walkthrough on authentication and offers endpoint descriptions, the overall content is relatively limited in scope. This may limit users' ability to fully grasp and utilize the API's capabilities effectively. We found that the LUY REST API documentation could greatly benefit from more detailed scenarios and additional code examples. By providing users with clear, real-world use cases and accompanying sample code, the documentation would be better equipped to support developers in understanding the API's features and functionality, ultimately enabling more effective integration within their DevOps environments.

DevOps Integration

The support for CRUD operations on building blocks and the entire data model allows for easy synchronization of data between LUY and other tools, such as CI/CD pipelines, configuration management systems, and monitoring platforms. These API's features enable integration with various DevOps tools and practices.

Summary

In conclusion, while the LUY REST API offers a range of valuable features and capabilities, its documentation requires improvement to enhance user understanding. By expanding the content to include more detailed scenarios and code examples, the documentation can better support developers in using the API to its fullest potential within their DevOps ecosystems.

4.3.2 LUY Documentation Review

For the documentation review of the LUY EAM Tool, we employed a similar approach as for the LeanIX and ADOIT EAM Tools. Our objective was to access the documentation and locate information about connecting our program to LUY's REST API. The documentation was readily accessible from within the workspace, and a link was also provided when we received our login credentials for the tool.

Navigating the documentation was efficient, as we were able to quickly find the relevant sections using the search bar. This ease of navigation allowed us to quickly locate the information needed to understand and integrate with the LUY REST API, leading to a positive user experience when working with the LUY EAM Tool's documentation.

However, we encountered some difficulties when attempting to authenticate with the LUY REST API using the existing session. The method outlined in the documentation proved unsuccessful for us, as we consistently received a "Permission denied" response. Despite this setback, we were able to find an alternative authentication method within the same documentation page. This alternative method involved creating a new session for each API request, which ultimately allowed us to proceed with our evaluation of the LUY EAM Tool.

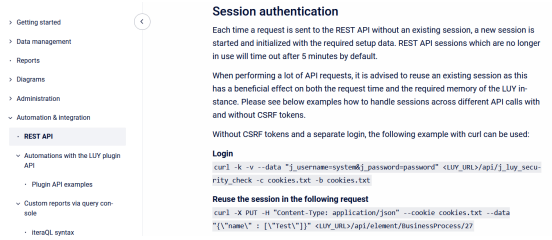

Figure 4.9: LUY REST Authentication page

4.3.3 LUY Example API Requests

In the Swagger UI for LUY it is possible to try out the endpoints[12].

To authenticate, a Basic Auth method can be used. A cookies.txt file can be created to reuse the authentication session (useful for many requests). In our example we will use the Basic Auth method.

Example cURL request to list all building blocks with their IDs, names and other data:

```
curl -X GET "https://<LUY-URL>/luy/api/data" \
-u "<system>:<password>" \
-H  "accept: application/json" \
--cookie-jar cookies.txt
```

The following variables must be replaced: <LUY-URL> with the URL of the LUY instance, <system> and <password> with the user data for the authentication.

To create a building block the following example cURL can be used:

```
curl -X POST "https://<LUY-URL>/luy/api/element/BusinessDomain" \
-u "<system>:<password>" \
-H  "accept: application/json" \
-d '{"name": ["new element"],
"description": ["description containing brief information about the eleme
--cookie-jar cookies.txt
```

[12] https://demo.luy.eu/api/7_3/ accessed on 2023-03-15

The full description of the Single Element API and other APIs can be found in the Swagger UI of LUY[13].

4.4 Comparative Analysis Summary

During this chapter we analysed the Application Programming Interfaces of the three EAM Tools: LeanIX, ADOIT and LUY. Our evaluation focused on the quality, comprehensiveness, and clarity of the APIs provided by these tools, as well as their suitability for integration with DevOps practices. Here are our findings:

ADOIT API

Among the three EAM tools, ADOIT's API stands out as the most well-documented. The documentation contains numerous examples and step-by-step guides, which enhance the understanding and implementation of various scenarios. As a result, users can easily grasp the capabilities of the API and use less effort to integrate it into their DevOps processes.

LeanIX API

While the documentation to the LeanIX API also provides clear guidance, it falls short in certain areas. For instance, the documentation does not cover essential information, such as the different licensing restrictions. This omission can make it challenging for users to follow the provided guides and fully leverage the API's capabilities. Moreover, LeanIX employs a unique Data Interchange Format, which can require additional effort to comprehend and utilize effectively. Despite these limitations, the LeanIX API remains a valuable option for organizations seeking to integrate EAM tools with their DevOps practices.

[13] https://demo.luy.eu/api/7_3/#/SingleElementAPI/createElement accessed on 2023-03-15

LUY API

The LUY API documentation is clear and concise, yet it could benefit from providing more examples for various scenarios. A broader range of examples would help users better understand the API's capabilities and integrate it more efficiently with DevOps processes. Additionally, the troubleshooting page is limited in scope and could be expanded to cover more cases, such as potential issues with reusable sessions during authentication. By addressing these gaps, the LUY API could become a more comprehensive and user-friendly solution for organizations looking to implement EAM tools in their DevOps environments.

5 Discussion

5.1 Recommendations for Organizations Integrating EAM Tools in DevOps

- **Prioritize Integration Capabilities**: When selecting an EAM tool for your organization, prioritize those with robust, well-documented APIs. Since most EAM tools provide similar basic functionality and support popular EA frameworks, a tool with a thoughtfully designed and documented API can make a significant difference in the process of integrating it into DevOps.
- **Use External Expertise**: Many EAM tool producers offer consulting services to help organizations integrate their tools effectively. However, maintaining an EA model may require additional effort, and the accessibility and comprehensibility of the tool will be crucial when training new team members or transitioning responsibilities.
- **Early Integration**: To minimize additional effort and potential challenges, it is advisable to integrate the EAM tool into your organization's DevOps environment during the early stages of system development. It can prevent complications that may arise from trying to integrate it into an already complex system.
- **Ongoing Training and Support**: Ensure that your organization invests in ongoing training and support for team members working with the EAM tool. This will help when onboarding new staff members, such as EA architects, and support the continuous and effective use of the EAM tool within the organization and in its DevOps environment.

5.2 Future work

This study provides a comparison of various EAM tools within the context of DevOps environments. However, there are several areas that could be explored further in future studies. One potential avenue for future work is developing a simulation program to test the DevOps integration. In subsequent studies, researchers could expand the choice of EAM tools under investigation to cover a broader range of the EAM market. This wider selection would enable a more comprehensive understanding of the EAM landscape. Additionally, acquiring the appropriate licenses for each EAM tool would allow for running complete simulations against their APIs, providing a more accurate assessment of their performance, functionality, and compatibility within controlled DevOps settings. Furthermore, future research could delve deeper into the design and execution of the simulation programs, exploring various scenarios, performance metrics, and integration challenges that may arise in real-world DevOps contexts.

6 Conclusion

This thesis has presented a comparison and evaluation of three EAM tools - LeanIX, ADOIT, and LUY - in the context of DevOps integration. Through a multi-method approach, including Interface Analysis, Documentation Review, and providing Example API Requests, the study has provided insights into the usability, integration capabilities, and potential challenges associated with each tool's API.

The findings suggest that among the tools analyzed, ADOIT's well-documented API stands out, offering numerous examples and step-by-step guides for various scenarios. However, each tool has its strengths and weaknesses, and the most suitable choice ultimately depends on an organization's specific needs and requirements.

In this study we also provided some recommendations to the organizations looking to integrate EAM tools into their DevOps processes. They are advised to prioritize tools with robust, well-documented APIs, use external expertise when needed, integrate the tools early in the development process and invest in ongoing training and support. By following these recommendations, organizations can effectively implement DevOps practices while maintaining up-to-date and accurate EA models.

In conclusion, this study contributes to the understanding of EAM tool integration in DevOps environments. As DevOps practices continue to evolve in the industry, the integration of EAM tools will remain an important aspect of managing complex IT environments.

Bibliography

[1] Ahlemann, Frederik ; Stettiner, Eric ; Messerschmidt, Marcus ; Legner, Christine: *Strategic enterprise architecture management: challenges, best practices, and future developments*. Springer Science & Business Media, 2012

[2] Booch, Grady: UML in action. In: *Communications of the ACM* 42 (1999), Nr. 10, S. 26–28

[3] Buckl, S. ; Ernst, A. M. ; Lankes, J. ; Matthes, F. ; Schweda, C. M.: State of the Art in Enterprise Architecture Management - 2009. (2009)

[4] Clements, Paul ; Garlan, David ; Bass, Len ; Stafford, Judith ; Nord, Robert ; Ivers, James ; Little, Reed: Documenting Software Architectures: Views and Beyond. (2002), 06. ISBN 0201703726

[5] Courtemanche, Meredith ; Mell, Emily ; Gillis, Alexander S.: *What is DevOps? The ultimate guide*. – URL https://www.techtarget.com/searchitoperations/definition/DevOps. – Accessed on: 2023-03-15

[6] Group, Object M.: *About the Business Process Model and Notation Specification Version 2.0.2*. 2014. – URL https://www.omg.org/spec/BPMN/2.0.2/. – Accessed on: 2023-03-15

[7] Group, The O.: *ArchiMate Language Structure*. – URL https://pubs.opengroup.org/architecture/archimate3-doc/chap03.html. – Accessed on: 2023-03-15

[8] Group, The O.: *The TOGAF® Standard, 10th Edition*. 2022. – URL https://publications.opengroup.org/c220. – Accessed on: 2023-03-15

[9] Hauder, Matheus ; Schulz, Christopher ; Roth, Sascha ; Matthes, Florian: Organizational factors influencing enterprise architecture management challenges. In: *21st European Conference on Information Systems (ECIS), Utrecht, Netherland*, 2013

[10] Heiden, Gilbert van der ; Jhawar, Akshay ; Hart, Nolan: *Magic Quadrant for Enterprise Architecture Tools*. 2021. – URL https://web.archive.org/web/20220328093124/https://www.gartner.com/doc/reprints?id=1-27ZV75UX&ct=211109&st=sb. – Accessed on: 2023-03-15

[11] ISO Central Secretary: Software, systems and enterprise — Architecture description / International Organization for Standardization. URL https://www.iso.org/standard/74393.html, 2022 (ISO/IEC TR 42010:2022). – Standard. Accessed on: 2023-03-15

[12] Jhawar, Akshay ; Heiden, Gilbert van der ; Gianni, Andrew ; Frangou, Andreas: *Gartner Magic Quadrant for Enterprise Architecture Tools*. – URL https://www.gartner.com/doc/reprints?id=1-2C044J8W&ct=221214&st=sb. – Accessed on: 2023-03-15

[13] Jonkers, Henk ; Lankhorst, Marc ; Doest, Hugo ter ; Arbab, Farhad ; Bosma, Hans ; Wieringa, Roel: Enterprise architecture: Management tool and blueprint for the organisation. In: *Information Systems Frontiers* 8 (2006), 02, S. 63–66

[14] Kim, Gene ; Debois, Patrick ; Willis, John ; Humble, Jez: *The DevOps Handbook: How to Create World-Class Agility, Reliability, and Security in Technology Organizations*. IT Revolution Press, 2016. – ISBN 1942788002

[15] Kleehaus, Martin ; Matthes, Florian: *Automated Enterprise Architecture Model Maintenance via Runtime IT Discovery*. S. 247–263, 01 2021. – ISBN 978-3-030-49639-5

[16] Langenberg, Kerstin ; Wegmann, Alain: Enterprise Architecture: What Aspects is Current Research Targeting. (2004), 01

[17] Lankhorst, Marc: *Enterprise Architecture at Work: Modelling, Communication and Analysis*. 4th. Springer Publishing Company, Incorporated, 2017. – ISBN 3662539322

[18] Lucke, Carsten ; Krell, Sascha ; Lechner, Ulrike: Critical issues in enterprise architecting–a literature review. (2010)

[19] Management, US O. of ; Budget: *Federal Enterprise Architecture Framework version 2*. 2013. – URL https://obamawhitehouse.archives.gov/omb/e-gov/FEA. – Accessed on: 2023-03-15

[20] Tao, Fei ; Zhang, He ; Liu, Ang ; Nee, A. Y. C.: Digital Twin in Industry: State-of-the-Art. In: *IEEE Transactions on Industrial Informatics* 15 (2019), Nr. 4, S. 2405–2415

[21] West, D. ; Bittner, K. ; Glenn, E.: *Ingredients for Building Effective Enterprise Architectures*. 2002. – URL http://www-106.ibm.com/developerworks/rational/library/content/RationalEdge/nov02/EnterpriseArchitectures_TheRationalEdge_Nov2002.pdf. – Accessed on: 2023-03-15

[22] Winter, Katharina ; Buckl, Sabine ; Matthes, Florian ; Schweda, Christian: Investigating the State-of-the-Art in Enterprise Architecture Management Methods in literature and Practice., 01 2010, S. 90

[23] Winter, Robert ; Fischer, Ronny: Essential Layers, Artifacts, and Dependencies of Enterprise Architecture. In: *http://www.alexandria.unisg.ch/Publikationen/67123* 3 (2006), 01

[24] Zachman, J. A.: A framework for information systems architecture. In: *IBM Systems Journal* 26 (1987), Nr. 3, S. 276–292

Erklärung zur selbstständigen Bearbeitung

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne fremde Hilfe selbständig verfasst und nur die angegebenen Hilfsmittel benutzt habe. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich gemacht.

Ort	Datum	Unterschrift im Original