An Interface for Visualizing the Evolution of Enterprise Meta-models

Francisco Miguel Grilate Ferreira

Thesis to obtain the Master of Science Degree in

Information Systems and Computer Engineering

Supervisors: Prof. Miguel Leitão Bignolas Mira da Silva, Prof. Pedro Manuel Moreira Vaz Antunes de Sousa

Examination Committee

Chairperson: Prof. João Emílio Segurado Pavão Martins
Supervisor: Prof. Pedro Manuel Moreira Vaz Antunes de Sousa
Member of the Committee: Prof. Daniel Jorge Viegas Gonçalves

November 2016

Acknowledgments

My acknowledgments start by emphasizing the important role of both my supervisors, Prof. Miguel Mira da Silva and Prof. Pedro Sousa, during development the of my master thesis. Their advice and expertise motivated and inspired me during this new phase, and without their guidance all of this could not be possible.

I want to express my gratitude to Link Consulting organization, which provided a great environment for the development of my thesis project. A special thank you to the EAMS's development team for being always helpful and patient when I needed their support.

I would also like to thank all my friends and colleagues at IST and INOV for their helpfulness, experiences shared together, debates of ideas and leisure moments. A particular acknowledgment for Nuno Silva, my research partner, to who I would like to express my gratitude for the supervision, encouragement and constructive criticism.

I would also like to recognize the importance of my friends outside IST. They were an important part of my academic journey, who were always there for me in both good and bad moments.

Last but not least, a huge acknowledgment to my family, for the unconditional love, inspiration and strength.

Abstract

The practice of Enterprise Architecture (EA) has proven its utility in the alignment of business and IT. Organizations incorporate EA as a fundamental practice, representing it through a set of models and/or representations. Models are designed, using as a guideline EA Frameworks, depicting the organization's meta-model. EA Frameworks are very complete and generic, making it difficult for organizations to incorporate all concepts into their organizational structure. Viewing a meta-model as a living artifact, enriching it as the organizational needs grow, becomes then appealing. Managing EA evolution when new requirements arise is not a trivial task, thus, threatening the success and efficiency of EA development projects. In this thesis, the author proposes an interface that provides a consistent environment for visualizing and managing the evolution of EA meta-models. The interface used as demonstration an EA evolution project from a large European financial organization, and was evaluated using a metric-based validation for the proposed information system (IS) artifact, encompassing specific techniques to evaluate interfaces.

Keywords: Enterprise Architecture, EA Management, EA Project, EA Tools, EA Transformation, Interface

Resumo

A prática de Arquitetura Empresarial (AE) já provou a sua utilidade para as organizações no alinhamento entre os processos de negócio e as tecnologias de informação. As organizações integram a AE como uma prática fundamental, representando-a através de modelos e/ou representações. Os modelos são criados usando como diretriz frameworks de AE, que definem o meta-modelo da organização. As frameworks de AE existentes são muito completas e genéricas, dificultando a representação de todos os seus conceitos na estrutura de uma organização. Idealizar um meta-modelo como um artefacto vivo, incorporando novos conceitos à medida que as necessidades de uma organização crescem, torna-se então apelativo. Gerir a evolução de uma AE quando novos requisitos emergem não é uma tarefa trivial, pondo em risco o sucesso e eficiência do desenvolvimento de um projeto de AE. Nesta tese, o autor propõe uma interface capaz de garantir um ambiente consistente para visualizar e gerir a evolução de meta-modelos em AE. A interface utiliza como demonstração os dados de um projeto de AE de uma grande organização financeira e é avaliada usando uma técnica de validação baseada em metricas para artefactos de sistemas de informação (IS), incorporando técnicas específicas para a avaliação de interfaces.

Palavras-Chave: Arquitetura Empresarial,Gestão de AE, Projecto de AE, Ferramentas de AE, Transformação de AE, Interface

Contents

List of Tables

List of Figures

Acronyms

ACF Architecture Capability Framework. 10, 11

ADM Architecture Development Method. 10, 11

CIM Computation Independent Model. 12, 13

CWM Common Warehouse Metamodel. 12

DSML Domain-Specific Modeling Languages. 17

DSRM Design Science Research Methodology. 2, 23, 51

EA Enteprise Architecture. 1, 2, 5–9, 11, 14–17, 21–24, 26, 28–33, 35, 37, 38, 41–43, 45, 46, 48, 49, 51, 52

EAF EA Framework. 1, 5, 8, 9, 14, 21

EAM EA Management. 1, 2, 5–9, 13–16, 21–23, 26, 28, 29, 48, 51

EAMS Enterprise Architecture Management System. 24–26, 41, 48

IS information system. 1, 2, 6, 45, 49, 51

IT Information technology.. 1, 7, 8, 15

MDA Model-Driven Architecture. 12, 13

MDD Model-Driven Development. 17

MOF Meta Object Facility. 12

OMG Object Management Group. 12

PIM Platform Independent Model. 13

PSM Platform Specific Model. 13

TOGAF The Open Group Architectural Framework. 10, 11, 21

UML Unified Modeling Language. 12

Chapter 1

Introduction

Enteprise Architecture (EA) is defined as a coherent whole of principles, methods, and models used in the design and realization of an organization's structure, business processes, information system (IS) and infrastructure (Lankhorst, 2009). EAs aim to capture the essentials of the business, IT and its evolution using enterprise models and does so by consolidating architecture decisions along with migration and implementation plans to address strategic aspects of an organization. Thus, the practice of EA Management (EAM) within organizations has proven its relevance in the business and IT alignment.

An effective EAM follows not only a consistent baseline of information concerning the *as-is* landscape of the organization, but also a planning process of the envisioned *to-be* state. The planning and implementation of the organizations *to-be* landscape, from its *as-is*, is done within the scope of an EAM process such as EA projects. An EA project is supported by three central tasks: *documenting* the current state of the organization's architecture, *envisioning* the future perspective of the architecture, and *planning* different intermediate architecture scenarios deciding on an architecture roadmap (Buckl et al., 2009).

The organization's architecture is defined by an EA meta-model, specifying the information regarding the elements and relationships used to represent and/or model the organization (Buckl et al., 2007a).

Since current EA Frameworks (EAFs) and notations are very complete, organizations might struggle to incorporate all existing EA meta-model concepts into their organizational structure, thus would profit from using a simplified meta-model initially, and then complement it while the organization's needs grow over time. Therefore, starting an EA project with a simplified EA meta-model and enriching it while growing within the organization turns out to be appealing. However, addressing these projects implies that EA meta-models are supported as living artifacts evolving according to the organization's needs (Tribolet et al., 2014).

During an EA project, the planning and implementation of the *to-be* landscape is performed as an iterative process. Each iteration may raise a number of issues resulting from uncoordinated, ad hoc decisions concerning EA components and their interrelationships. Issues like the duplication of efforts and resources, poor coordination and control, management and business performance issues, and inefficiencies in operation (Kaisler et al., 2005) are the outcomes of such decisions. Consequently, the meta-models and concepts used to model the first *as-is* are no longer enough to model the current *as-is* of even the *to-be*.

There are a variety of EA tools that support numerous EAM tasks. These tools support the EA architect in the design of EA models as structured, object-oriented models, reflecting the organization's landscape, which are stored in EA repositories. Existing EA Tools features are complete when defining and representing the organization's EA, allowing EA architects to choose one according to their specific needs. Nonetheless, although EA tools also support flexible information models, adapted to the organization's needs, the support for managing them as an iterative process, requiring the migration of affected EA instances after changes are performed, is still limited.

Therefore, the lack of support from EA tools in regard to EA evolution as an iterative process leads to our research problem, which can be identified as the **inefficient implementation of EA development projects**, which poses as a limitation when trying to develop an EAM program.

To address the identified problem, the author propose an EA meta-model evolution interface that supports a set of EA transformations as an innovative, purposeful IS artifact using Design Science Research Methodology (DSRM) (Von Alan et al., 2004).

The steps of the DSRM are correlated to the chapters of this thesis. The "Problem" and "Related Work" chapters, present the motivation behind the research (Identify Problem and Motivate) by initially presenting a description of the identified problem, followed by an overview and discussion on the state-of-the-art concerning the topics of EA, EAM problems, EAFs, enterprise transformation, EA tools, meta-model evolution, and information visualization, as found in literature. In the "Proposal" chapter, the author provides a detailed description of the developed IS artifacts, that corresponds to DSRM's Design and Development phase. Then, an evolution scenario is illustrated in the "Demonstration" chapter, showing the application of the proposed artifact by applying it to a large European financial organization's EA project. Next, the "Evaluation" section outlines a metric-based validation of the proposed IS artifact, validated by the demonstration itself, user tests and a usability questionnaire. Finally, the communication, research conclusions and themes for future work are presented in the "Conclusion" chapter.

Chapter 2

Research Methodology

The thesis follows the DSRM, in the context of Information Systems, which is an iterative process that produces an innovative product, also known as a design artifact (Von Alan et al., 2004).

This methodology was chosen since the author's motivation is to solve a problem in a specific organizational context, having explicit outcome objectives. The design science research method goal is utility, extending the boundaries of human and organizational capabilities by creating new and innovative artifacts, in contrast to behavioral science research goal which is to find the truth, seeking to develop and verify theories (Von Alan et al., 2004).

The DSRM iterative process is based on building and evaluating the artifact repeatedly, since knowledge and understanding of a design problem and its solution are only acquired in the building and application of an artifact (Von Alan et al., 2004). The DSRM can produce four kinds of artifacts to solve problems: constructs (vocabulary and symbols), models (abstractions and representations), methods (algorithms and practices) and instantiations (implemented and prototype systems) (Von Alan et al., 2004).

In this thesis the author proposes to create a new **instantiation artifact**, by implementing a new module for an EA tool, for a not yet solved problem in information systems.

The DSRM is composed by six steps: problem identification and motivation (defines the specific research problem and justify its value), definition of the objectives for a solution (describe the solution objectives

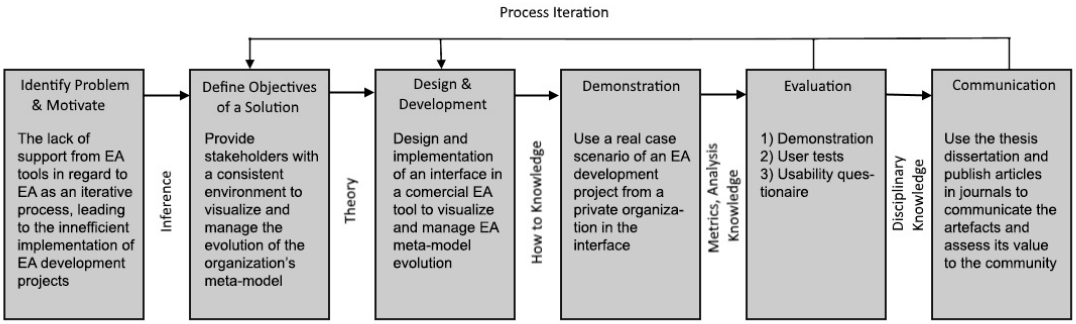


Figure 1: DSRM process (adapted from (Peffers et al., 2007))

from the problem definition and knowledge of what is possible and feasible), design and development (creates a new artifact by implementing the proposed solution), demonstration (demonstrate how the artifact can solve our problem), evaluation (observe and measure how well the artifacts supports the solution) and communication (communicate the contribution to solving the problem and its importance) (Peffers et al., 2007).

In order to apply the methodology we mapped the six steps to our specific problem as illustrated in Figure 1. In the thesis, step 1 corresponds to Introduction (Chapter 1), Research Problem (Chapter 3) and Related Work (Chapter 4). Step 2 and 3 are described in the Research Proposal (Chapter 5). Step 4 is addressed in Demonstration (Chapter 6), step 5 correlates to Evaluation (Chapter 7) and step 6 is presented in Conclusion (Chapter 8).

Chapter 3

Problem

EA is a holistic approach to the successful development and execution of a strategy for an organization, performing a complete enterprise analysis, design, planning and implementation. EA guides organizations through the changes necessary to execute their strategies.

EA development projects start with the definition of a set of concepts that will be used to represent and/or model the organization. These concepts, such as the ones presented in EAFs as TOGAF® (Josey, 2011) or notations like ArchiMate® (Josey et al., 2013), are used or proposed, or as in domain-specific languages (Frank et al., 2009), which define the EA meta-model. EA meta-models specify the information about EA concepts, i.e., the elements and relationships, which will be used to represent and/or model the organization.

Although EAFs offer guidance in identifying which areas of business and technology should be considered when creating an EA, they provide little assistance in creating the architectural artefacts themselves (Lankhorst, 2009).

Even though an architecture captures the relatively stable parts of business and technology, any architecture will need to accommodate and facilitate change, and architecture products will therefore only have a temporary status. Architectures change because the environment changes as new technological opportunities arise, and because of new insights as to what is essential to the business.

Since EA projects are very ambitious, the idea of starting with a simplified meta-model and enriching it while growing within the organization is appealing. However, addressing these projects implies that EA meta-models are supported as living artifacts that evolve according to the organization's needs (Tribolet et al., 2014).

Furthermore, each EA project, being an EAM process, requires the definition of different scenarios, from the current state of the organization's architecture to envisioned future states. Documenting the organization's current state and planning possible future states, helps in deciding the architecture roadmap (Buckl et al., 2009). Consequently, the transformation from the *as-is* state of the EA to the envisioned *to-be* state can be derived.

This transformation occurs in cycles, over and over again. Usually, the meta-models and concepts used to model the first *as-is* are no longer enough to model the current *as-is* or even the current *to-be*. For example, an organization wants to reformulate its business model by changing, deleting and

creating new business concepts. The current information systems architecture that supports the *as-is* and, supposedly, the *to-be* business cannot be adapted in any way. Thus, the organization's EA need to be drastically re-designed, not only business-wise, but also in terms of the information systems that will support the *to-be* business. Migrating all these changes can lead to many issues, being a manual and time consuming task that can affect the information's coherence.

Hence, managing the evolution of the EA meta-model, as well as deriving the necessary transformations to migrate the associated EA models, still poses resistance to the incremental approach of EA within organizations. The lack of support from EA tools regarding EA evolution as an iterative process leads to our research problem, which can be identified as the **inefficient implementation of EA development projects**, which poses as a limitation when trying to develop an EAM program.

Through the creation of an IS artifact that solves our research problem, the author aims to answer the following questions regarding the evolution of EA meta-models:

- Which are the differences between the EA meta-model *as-is* and *to-be* states?
- How many EA Transformations create/update/delete EA classes or EA properties?
- What EA operation type is applied to a specific EA meta-model component?
- Which are the EA requirements and EA transformations associated to a specific EA meta-model component?
- What is the most common EA operation type in the EA meta-model evolution?

Chapter 4

Related Work

In the next subsections, the author presents a literature review with reference to contributions on the topics of EA, EAM problems, EAFs, enterprise transformation, EA tools, meta-model evolution, and information visualization, as well as a brief discussion of the conclusions taken from the state-of-the-art analysis. This section corresponds to the ”Identify Problem & Motivate” step of the research method.

4.1 Enterprise Architecture

Current organization’s business practices find indispensable to have an integrated view of the entire enterprise, aligning its business and IT (Lankhorst, 2009). Performing changes in a company’s strategy and business goals has significant consequences in all domains of the enterprise. These domains comprise the organization’s structure, business processes, software systems, data management and technical infrastructures. Stakeholders, within and outside the organization, require information from each of these domains to be presented effectively. Providing an overview of each domain’s changes, and their impact on each other, to support the stakeholder’s decision process with the correct information is not always a trivial task.

Creating a complete overview of an organization, comprising all its domains and relations between them, can be achieved using an architecture.(Hilliard, 2000) defines an architecture as ”*the fundamental organization of a system embodied in its components, their relationships to each other, and to the environment, and the principle guiding its design and evolution*”. Architectures are then able to provide an integrated view of the system being designed or studied, helping to improve communication between architects and stakeholders through architecture models, views, presentations, and their analysis (Lankhorst, 2009).

The architecture definition can be extended to the enterprise’s entire domain, allowing to represent an organization’s business, IT and its evolution. (Lankhorst, 2009) defines EA as “*coherent whole of principles, methods and models that are used in the design and realization of the enterprise’s organizational structure, business processes, information systems, and infrastructure*”.

An EA captures a holistic view of the organization, focusing on their essentials rather than their specific problems, encompassing the necessary flexibility and adaptability to translate corporate strategy into

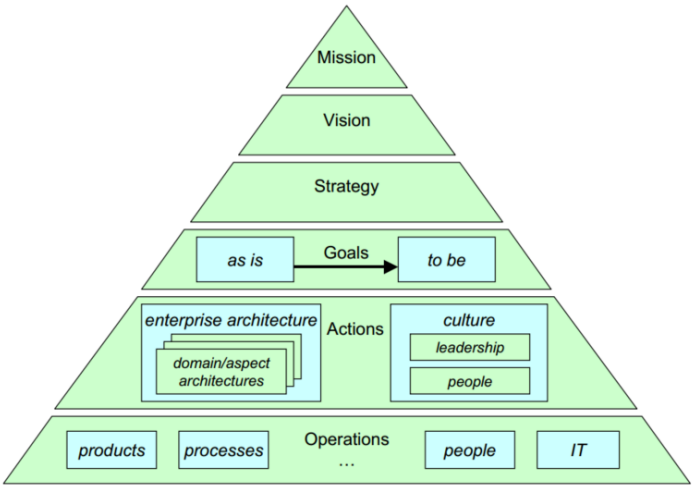


Figure 2: Enterprise architecture role in an organization (from (Lankhorst, 2009))

daily organizational operations. Choosing what to represent defines the architecture's quality, requiring a rational perspective to achieve the essential business objectives.

An EA provides valuable assistance in the process of managing the enterprise, helping to align business and IT, as Figure 2 illustrates. The enterprise's mission represents the reasoning behind its existence. The vision depicts the values the enterprise holds, and how the enterprises wants to be seen in the future. Bellow there is the strategy, defining the path the enterprise will follow to achieve their mission and vision. The path is divided into goals, guiding the enterprise's strategy execution. These goals translate into changes in the company's operations, which are represented through EA. EA provides a holistic perspective of the current and future operations, and on the actions followed to achieve the company's goals. The EA is complemented by the company's culture, composed by its people and leadership. In the pyramid's bottom the daily operations of the company are represented, which are governed by the all other layers.

Using EA as a management instrument helps enterprises determine their needs and priorities from the business perspective, as well as assess how the company may benefit from technological and business innovations (Lankhorst, 2009). Besides using EA to effectively achieve an organization's strategy and optimize its operations, EA can support the communication of the organizations structure and operations to external drivers, such as regulatory authorities.

4.2 EA Management Problems

(Ahlemann et al., 2012) defines EAM as *"a management practice that establishes, maintains and uses a coherent set of guidelines, architecture principles and governance regimes that provide direction and practical help with the design and development of an enterprise's architecture in order to achieve their vision and strategy"*. However, the ability to integrate EAM into existing programs, organizational structures and governance regimes, as well as considering specific cultural aspects, establishes the EAM program's sucess (Ahlemann et al., 2012).

Organizations struggle when starting their own management programs. (Buckl et al., 2008) identified as the main problems the fact that organizations have to start their programs from scratch, EAF are

too abstract and therefore not implementable, organizations do not properly assess their informational needs, documentation for already implemented approaches is nonexistent, and EAM approaches are introduced as a single piece instead of being incremental. Overcoming these problems is crucial for a successful EAM program.

(Winter et al., 2010) performed an investigation on the state-of-art of EAM methodologies, in both literature and practice. Their analysis suggests that is very important to adapt the EAM approach to the organization's specific needs, although literature stated otherwise, and that performance indicators are not often used, and can be a point of improvement for EAM.

After implementing an EAM program, maintaining EA models can also be a challenge due to the impact of social, technological and economic changes. (Fischer et al., 2007) analysis highlighted several reasons for these:

- Increasing complexity of business transactions, due to customization of products and services
- Growing globalization with respect to service development, creation and distribution
- Accelerated rate of change in business models due to competition
- Growing regulatory framework
- The dependency on information technology, which results in technology related risks

Overcoming these issues is not easy, but (Fischer et al., 2007) suggests that integrating existing models from specialized architectures strongly influences the acceptance of EA as a management tool, providing valuable insights in the current and future architecture, and the integration of model data from specialized architectures into the EA repository is an ongoing process rather than a one-time effort.

EA repositories represent information through the instantiation of information models. An EAM information model can be defined as a model that specifies which information about the enterprise architecture, its elements and their relationships, should be documented, and how the respective information should be structured (Buckl et al., 2007a).

In current practice there is a tendency to develop new models when starting an EAM, instead of improving existing ones. The same is true for visualization techniques for EAM specific information and for methodologies that prescribe procedures for working with this information and its visualizations. This is an outcome from the lack of results in the field that are further developed and verified by other practitioners (Buckl et al., 2007a). Existing information models in research are neglected, unless a joint project with academia is executed or an information model is incorporated in the EAM tool used.

4.3 EA Frameworks

Using EA as a holistic perspective of an enterprise requires the employment of techniques in order to describe architectures in a coherent manner and to communicate them with all relevant stakeholders. Moreover, EA architectures are not static, changing over time when new development plans arise, demanding methods for analyzing the impact of these changes.

EAFs provide a set of methods and techniques to specify, analyze and communicate EAs, fulfilling the needs of the different stakeholders involved. EAFs structure the architecture description through the identification and association of different architectural viewpoints, and related modelling techniques

(Lankhorst, 2009). Adopting EA frameworks can accelerate the implementation of an EAM program, reduce the risk of EAM failure and make EAM more efficient and effective (Ahlemann et al., 2012).

An EAF comprises two main components (Abdallah and Galal-Edeen, 2006): a methodology and a framework. The methodology defines the standardized processes for developing an EA, whereas the framework provides a standardized classification tool for the EA deliverables or artifacts.

EA modeling commonly uses high level abstractions called views. EAFs use viewpoints to create views that represent different perspectives of a system model. Commonly used viewpoints are business architecture, process architecture, integration architecture, software architecture, and technology (or infrastructure) architecture (Winter and Fischer, 2006).

Although EAFs improved EA by providing systematic approaches to EA development, still many aspects of the architecture remain ambigous (Tang et al., 2004):

- What should be the scope of the architecture?
- What is the role of the EA architect?
- What should be the outcome of architecture activities?
- Architecture activities involve design and modeling, but what level of detail belongs to architecture and when do detailed design activities start?
- To what extent should and could outcomes of architecture be measured, verified or validated?
- A system of what size and complexity would require architecture?
- What is the relationship between enterprise architecture and stand-alone system architecture?

The next subsections provide an analysis on three commonly used EAFs.

4.3.1 The Zachman Framework

The first EAF, (Zachman, 1987), was introduced by John Zachman in 1987. The framework comprises a logical structure for classifying and organizing the descriptive representations of an enterprise, establishing a common vocabulary and a set perspectives for describing the enterprise's system. These representations characterize the management of the enterprise as well as the development of the enterprise's systems.

The framework architecture (Figure 3) is a matrix of 30 cells, where the rows represent the different perspectives and the columns the possible framework abstractions, subjects or objects, for each perspective.

A perspective represents how a stakeholder views the system. The various stakeholders are: *Planner, Owner, Designer, Builder and Sub-contractor*. Each perspective produces a specific outcome defining the *Scope, Enterprise or Business Model, System Model, Technology Model, and Detailed representation*.

The different types of information, for each perspective, are characterized by: *What* (information and data), *How* (function and process), *Where* (location hardware/software), *Who* (people in terms of allocation), *When* (timing requirements of business process), and *Why* (motivation).

The Zachman Framework provides a concise way to structure and model an enterprise and system architecture (Tang et al., 2004). It addresses the enterprise as a whole, is defined independently of tools

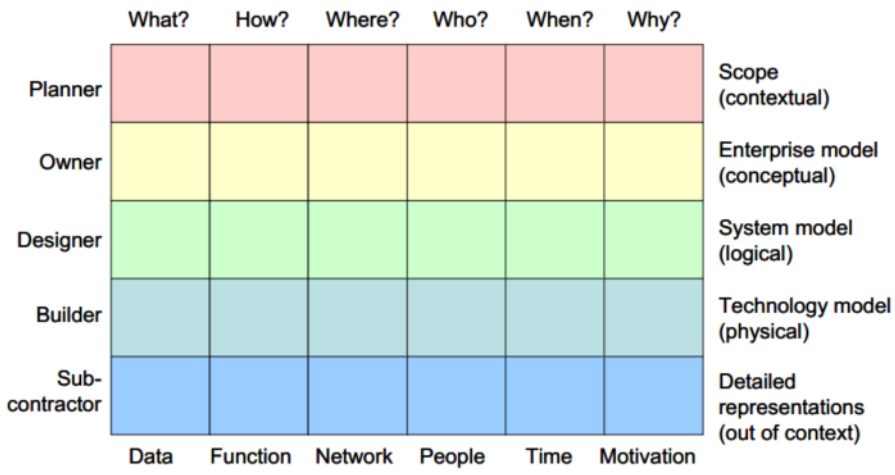


Figure 3: The Zachman Framework (from (Lankhorst, 2009))

or methodologies, and any issues can be mapped against it to understand where they fit. Nevertheless, the large number of cells can be an obstacle for the pratical applicability of the framework (Lankhorst, 2009).

As for the architecture's evolution aspect, the Zachman Framework, being a description-oriented framework, does not provide any rules or guidelines regarding the evolution of the enterprise architecture (Urbaczewski and Mrdalj, 2006).

4.3.2 The Open Group Architectural Framework (TOGAF)

The Open Group Architectural Framework (TOGAF) (Josey, 2011) is a framework for developing an enterprise architecture, providing assistance in the acceptance, production, use and maintenance of architectures. TOGAF covers four different types of architectures: business, data, application, and technology. A description of each of these types is presented in Table 1.

Architecture Type	Description
Business	The business strategy, governance, organization, and key business processes.
Data	The structure of an organization's logical and physical data assets and data management resources.
Application	A blueprint for the individual applications to be deployed, their interactions, and their relationships to the core business processes of the organization.
Technology	The logical software and hardware capabilities that are required to support the deployment of business, data, and application services. This includes IT infrastructure, middleware, networks, communications, processing, and standards.

Table 1: TOGAF's supported architectural types (adapted from (Josey, 2011))

Figure 4 depicts TOGAF's Capability Framework. The Architecture Development Method (ADM) is the core of TOGAF's framework. This method is operated by the Architecture Capability Framework (ACF),

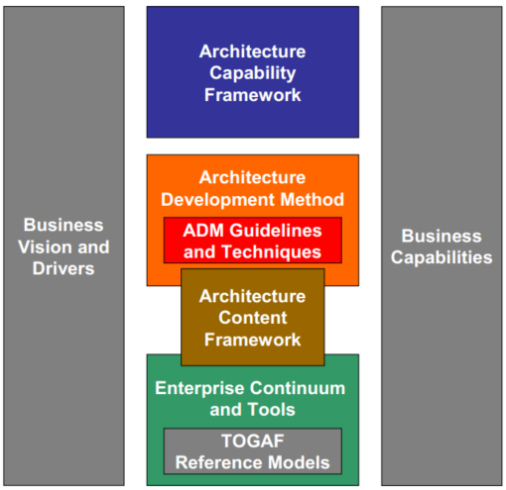


Figure 4: TOGAF 9.1 Capability Framework (from (Lankhorst, 2009))

which provides the enterprise with a set of reference materials on how to establish an architecture function. ADM's Guidelines and Techniques support the application of the ADM.ACF acts as a guide to help the architect establish an architecture practice within an organization. The resulting content is then stored in the Architecture Content Framework, which is classified according to the Enterprise Continuum and Tools. The repository initially is populated with TOGAF's Reference Models.

Architecture Development Method

The ADM, in Figure 5, describes a method for developing and managing the lifecycle of an enterprise architecture. ADM is an iterative, over the whole process, between phases and within phases. In each iteration a decision must be taken as to:

- The breath of coverage of the enterprise to be defined
- The level of detail to be defined
- The extent of the time period aimed at, including the number and extent of any intermediate time periods
- The architectural assets to be leveraged, including assets created in previous iterations of the ADM cycles, and assets available elsewhere in the industry

ADM comprises a number of architecture development phases, in cycles, as an overall process template for the architecture development activity. Each phase represents a narrative, describing it in terms of objectives, approach, inputs, steps, and outputs. Cross-phase summaries cover the requirements management.

TOGAF addresses the evolution of EAs by introducing three phases of the ADM that describe the architecture roadmap based on a gap analysis, the guidelines to move from the baseline to the target architecture presented in a detailed implementation and migration plan, and the architectural oversight of the implementation according to the target architecture. Those phases are: Opportunities and Solutions, Migration Planning, and Implementation Governance.

As a generic method, ADM is intended to be used by enterprises in a wide variety of different geographies and applied in different vertical sectors/industry types. As such, it may be, but does not necessarily have to be, tailored to specific needs. For example, it may be used in conjunction with the set of de-

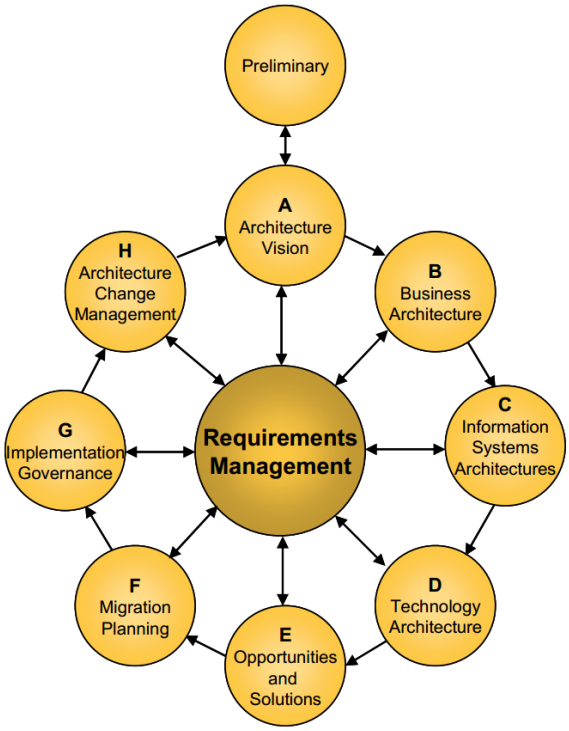


Figure 5: Architecture Development Method (from (Lankhorst, 2009))

liverables of another framework, where these have been deemed to be more appropriate for a specific organization.

4.3.3 Model-Driven Architecture (MDA)

The Model-Driven Architecture (MDA), from the Object Management Group (OMG), addresses the complete development lifecycle, covering the analysis and design, programming, testing, component assembly, as well as deployment and maintenance (Truyen, 2006). MDA contributes with an approach for software development, enabled by OMG specifications such as the Unified Modeling Language (UML) (Rumbaugh et al., 2004), the Meta Object Facility (MOF) (omg, 2006) and the Common Warehouse Metamodel (CWM) (Poole et al., 2002).

With the continuous emergence of new platforms and technologies, MDA aims to support the rapid development of specifications and centralize the process of their integration. Hence, MDA provides a comprehensive, structured solution for application interoperability and portability into the future. Precise modeling of the solution domain in UML provides the added advantage of capturing its inherent intellectual property in a technology neutral way.

A model-driven approach aims to support software development by using models as the primary source for documenting, analyzing, designing, constructing, deploying and maintaining a system. MDA uses 3 viewpoints, composed by one or more models, focusing on particular set of concerns while suppressing all irrelevant details. These viewpoints, in Figure 6, are computation independent, platform independent, and platform specific.

The Computation Independent Model (CIM)represents a system, and expected behavior, hiding all infor-

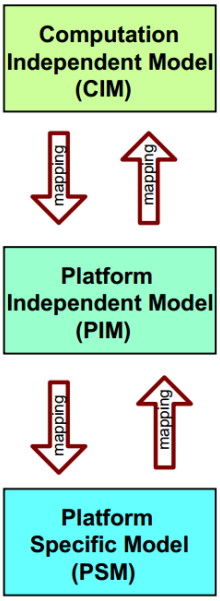


Figure 6: Model-Driven Architecture models (adapted from (Lankhorst, 2009))

mation technologies related specifications. Thus, remaining independent of the systems current or future implementation. CIM can also be referred as business or domain model since it uses vocabulary familiar to subject matter experts. CIM bridges the gap between the business domain experts and information technologists responsible for implementing the system. In an MDA specification, the CIM requirements should be traceable to the platform independent and platform specific constructs that implement them (and vice-versa).

The Platform Independent Model (PIM) accomplishes an abstraction level to enable its mapping to one or more platforms. A set of services is defined with abstraction to its technical details. Other models then specify the realization of these services in a platform specific manner.

The Platform Specific Model (PSM) incorporates the necessary details to describe how a system uses a particular type of platform, complementing the PIM. If the PSM does not incorporate all necessary details to produce an implementation of a platform is considered abstract, relying on other explicit or implicit models with the necessary details

MDA aims to ease the creation of machine-readable models, with the goal of long term flexibility in terms of: technology obsolescence, portability, productivity and time-to-marker, quality, integration, maintenance, testing and simulation, and return on investment. This methodology does not encompass the topic of model evolution.

4.4 Enterprise Transformation

Enterprise transformation is known in the enterprise engineering community as a set of initiatives that change the organization's domain (its structure and dynamics) from its current *as-is* state to a predetermined *to-be* state (Sousa et al., 2009; Tribolet et al., 2014). These initiatives are part of the EAM practice within organizations.

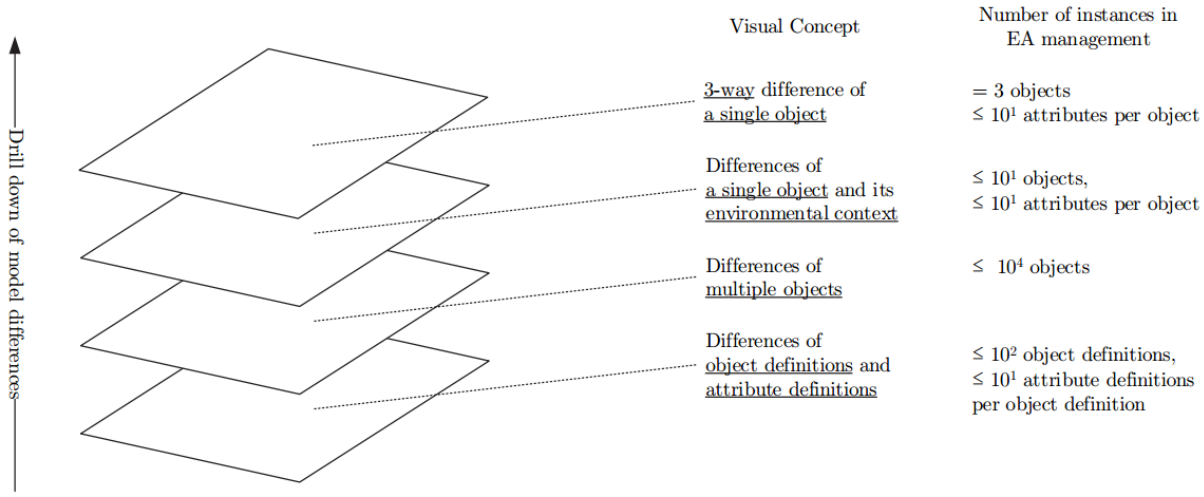


Figure 7: Four-layered conceptual design (from (Roth and Matthes, 2014)

To address the changes within the organization's domain, a set of principles were proposed concerning the evolution of architectural views (Sousa et al., 2009; Tribolet et al., 2014). Keeping up-to-date architectural views with organization's changes has brought up issues in EA development projects, such as being time-consuming, error-prone, and even cause some misalignments between both business and IT spectrums. In order to mitigate this, rather than using a versioning schema to handle architectural views evolution, was proposed that all organizational artifacts have four fundamental invariant states (gestating, alive, dead, and retired) in their lifecycle as if each organizational artifact was a living entity.

An organizational artifact is conceived as the future result of an EA transformation project, thereby entering the gestating state. An artifact remains in gestation throughout the project's timeline becoming alive after the project ends. The artifact dies after the completion of a decommissioning project or when a transformation project is canceled. In the end, the dead artifact is then retired when a retirement project explicitly removes it from the organizational infrastructure.

Taking into consideration the importance of models in EA, (Buckl et al., 2007b) performed a research about software cartography, in which they develop an approach for EA modeling. Their approach proposes a method for the automatic creation of EA models and visualizations. They identified creating visualizations for EA models as an error prone and time-consuming task, leading to incoherent models with aged data, due to not being updated. Also contributing is the fact that EAM tools commonly provide the user with the capability of introducing visual elements without defined semantics, disconnecting the visualization from the respective data. To overcome these issues, their approach focuses on connecting the information model to the visual representation through transformations.

Furthermore, a viewpoint was presented for road mapping the development of EA over time, complementing it with a conceptual model explaining the information demands for such roadmap plans (Buckl et al., 2009). Both artifacts were drawn from (Denert-Stiftungslehrstuhl, 2008), a pattern-based approach for EAM, meaning both can be integrated into a multitude of organization-specific EAM approaches. The presented conceptual model emphasizes the role of the IT project as a driver to accomplish an EA transformation, performing the transformation from the current to the planned EA.

Also, a four-layered conceptual design of an interactive visualization to drill down and analyze model differences in EA meta-models and respective models was proposed (Roth and Matthes, 2014), illus-

trated in Figure 7. The visualization was motivated from (Roth and Matthes, 2014) research, where was stated that no common standard has been established to visualize and analyze differences in models and in particular EA models. The conclusions drawn were that EA models shared many similarities with software models with respect to complexity, the number of instances, etc. Therefore, an EA model could be perceived as a software model.

(Roth and Matthes, 2014) conceptual design first layer ”Meta-model Differences” evidences through a graph the differences between two meta-models, including their object definitions and attribute definitions. The second layer ”Instance Overview” contributes with an overview of the meta-model object definition's instances, and their attributes, navigating from a single object definition selected in the first layer. The third layer ”Instance Neighborhood” represents an object's impact to its environment. The last interactive layer ”Three-Way Differences” depicts the object's versions, including it's original version if existent. Throughout all the layers a color code is used were orange means an object as been modified, blue when as no changes, red if the object is deleted, and green if created.

Another study identified the challenges of documenting EAs as being a manual, time-consuming and costly task (Buschle et al., 2012). A vendor-specific enterprise service bus was reverse-engineered to understand the degree of coverage to which data of a productive system could be used for EA documentation. From there, a set of model transformations for automating EA documentation was derived.

4.5 EA Tools

In order to understand the actual state of progression regarding EA transformation support, analysis, and management within the EA industry, an assessment on the current EA Tool's features was made.

(Matthes et al., 2008) performed an analysis to 9 EA tool, which were evaluated by 3 different teams. The evaluation process was based on the analysis of functional criteria and EAM task criteria. Figure 8 and Figure 9 illustrate the minimum and maximum achieved results from (Matthes et al., 2008) for each criteria, respectively. For each criteria different test scenarios were created, complemented by online questionnaires. The findings were then processed and presented in spider diagrams, according to their ratings for each task. The ratings range from 0 to 7, being 7 the highest score possible. Later on, (Matthes et al., 2014) complemented the EA tool analysis, using the same evaluation process, by evaluating 4 additional EA tools.

Their functional criteria analysis concluded that improvements can be done in “Creating visualizations”, since scenario generation is limited and flexible models are not supported. Moreover, “Importing, Editing, and Validating” has no standard exchange format for EA models and a common terminology to define information models is missing. Improvements can also be made regarding ”Interacting with Editing of and Annotating Visualizations , Editing of, and Annotating Visualizations” since semantic changes could provide an improvement towards graphical modeling.

More specifically, in the context of EA meta-model evolution, the functional criteria also assessed the ”flexibility of information model”. From the 13 analyzed tools, only 7 supported features to change or adapt the information model. From these 7 tools, only 6 were rated according to their functional support. The analysis showed that 4 tools scored above 5, incorporating a complete solution to manage the information model. Although most EA tools already show concerns about the information model flexibility, none of the tools mentioned any considerations regarding the migration of the EA models

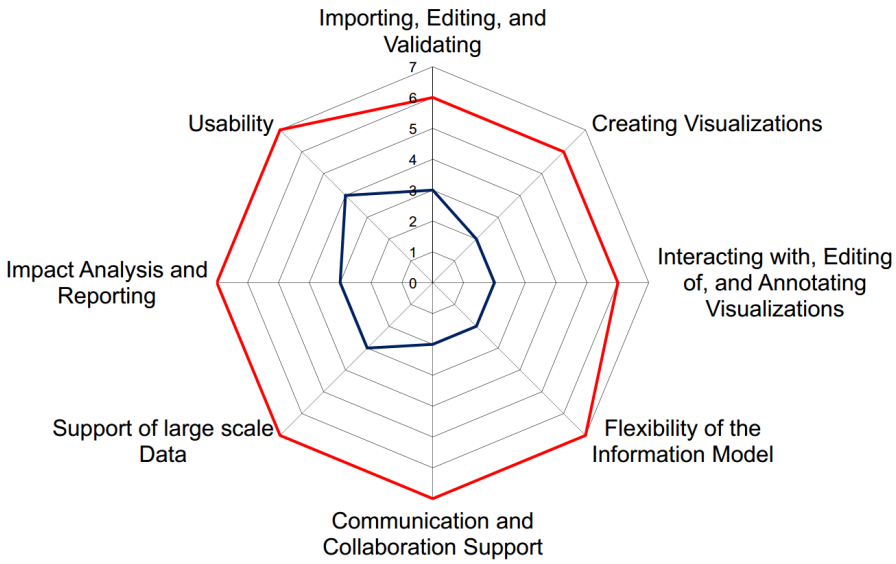


Figure 8: Functional criteria spider diagram (from (Matthes et al., 2008))

affected by those changes.

The task analysis showed that the tasks in ”Landscape Management” could still be improved in EA Tools. More specifically, versioning application landscapes retains potential for improvement, and not all tools provide methods for deriving the planned landscape from the planned project portfolio. ”Synchronization Management” is also limited, as no tool directly supports the concept of project delay. ”SOA Transformation Management” lacks tool support for identifying services, and in ”Infrastructure Management” not all tools provide concepts for lifecycle aspects of infrastructure components.

4.6 Meta-model Evolution

In the Model-Driven Development (MDD) area, software applications can be described through a set of models using Domain-Specific Modeling Languagess (DSMLs). These DSMLs are thoroughly used with the purpose of increasing software quality and productivity by serving as input for code generation and as validation and testing tools. In order to preserve such advantages, DSMLs are subject to change given a set of changing requirements. This causes the adaptation of existing models according to the language changes.

DSMLs are defined by meta-models, i.e., models that define the structure of a modeling language used to describe a model (da Silva, 2015). Thus, meta-model evolutions must be followed by model migrations, i.e., co-evolving both the meta-model and respective models.

The following requirements must be considered when dealing with model migration (Mantz et al., 2015):

- Migrated models must belong to the evolved modeling language
- All models of the original modeling language can be migrated to the evolved language
- Model migration should be automatically deduced from its meta-model evolution or specified using a high-level language

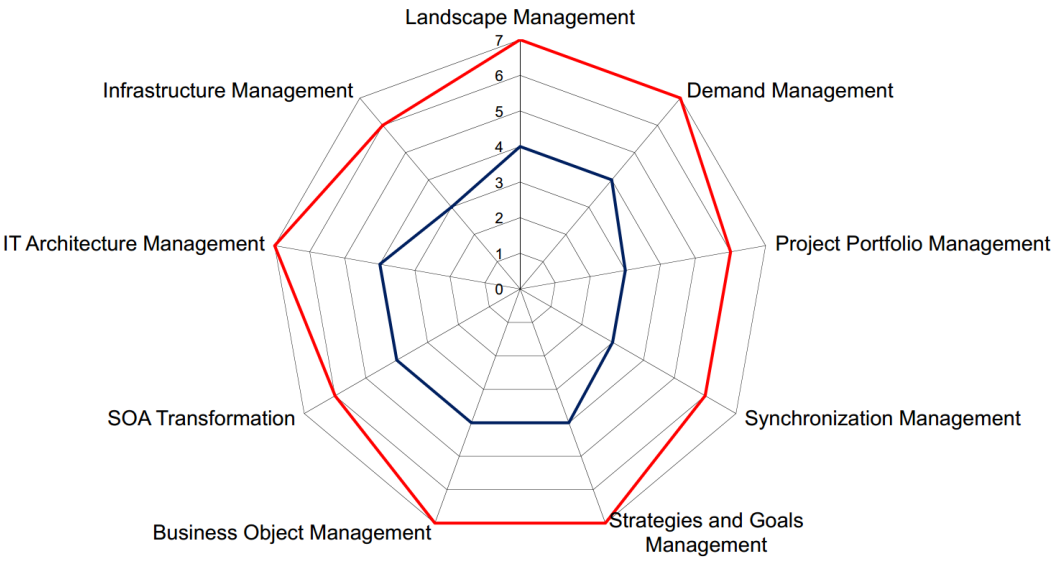


Figure 9: Task criteria spider diagram (from (Matthes et al., 2008))

– General strategies for model co-evolution are formulated independently of a specific meta-modeling approach

To address these requirements, research has been done on the subject of co-evolution and/or co-adaptation.

(Mantz et al., 2015) proposed a formal framework based on graph transformations to model co-evolution of meta-models and models. His research states that meta-model evolutions and migrations of their instance models are specified by coupled graph transformations. Furthermore, to specify model migrations on a high abstraction level and with the possibility of reusing them, Mantz introduced model migration schemes (Mantz et al., 2015). These schemes increase the level of automation during a migration step and are derived from given meta-model evolution rules. Once a model migration scheme is specified, it can then be applied automatically to instance models yielding well-typed migrated results.

Moreover, an evolutionary method was established to automate and secure modeling language evolution (Herrmannsdörfer et al., 2011). This method records the adaptations to the meta-model as coupled operations in a history model, which can later be used to automatically migrate models conforming to the meta-model.

(Wachsmuth, 2007) proposed a transformational setting for stepwise meta-model adaptation, where each meta-model transformation implements a typical adaptation step performed manually. Then, each transformation is classified according to specific preservation properties (Wachsmuth, 2007). Lastly, for each transformation, corresponding instant co-transformations (co-adaptations) of the meta-model instances are presented always conforming to the current version of the meta-model.

This transformational setting was adapted to the Enterprise Engineering area of research, more specifically, to the subject of enterprise transformation (Silva et al., 2016). A set of EA migration rules were proposed in order to mitigate time and coherence issues revolving around the migration of enterprise models when changes were made within the organization's structure, i.e., EA meta-model changes.

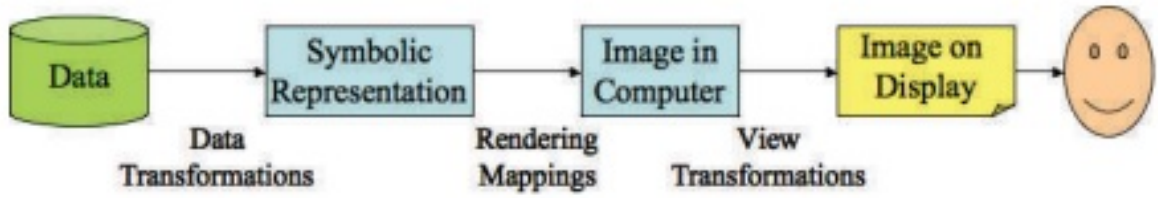


Figure 10: High level view of the visualization process (from (Ward et al., 2010)

4.7 Information Visualization

Organizations also struggle with the fact that they have to maintain numerous, redundant, information environments leading to inefficiencies and highs costs. Providing one information environment will allow an organization to meet strategic goals, anticipate and respond to changes in the global economy, and use information for sustained competitive advantage (Godinez et al., 2010). An analysis on information visualization techniques and principles was made in order to achieve a coherent information visualization environment.

There are three main concepts associated with visualization (Chen et al., 2009): data, information and knowledge . These terms can be misleading, and for our thesis proposal two main spaces are fundamental: **perceptual and cognitive** space and **computational** space. The **perceptual and cognitive** space, using Ackoff's definitions, refers to data as symbols, information as data processed to be useful (answering to "who", "what", "where" and "when" questions), and knowledge as the application of data and information, to achieve an answer to the questions (Chen et al., 2009). From the **computational** space, data is defined as the computerized representation of models and attributes of real or simulated entities, information as the data that represents the results of a computational process assigning meaning to the data, or the transcripts of some meanings assigned by human beings, and knowledge as data that represents the results of a computer-simulated cognitive process, such as perception, learning, association, and reasoning, or the transcripts of some knowledge acquired by human beings (Chen et al., 2009).

Visualization

Visualizations are crucial for acquiring knowledge since humans use vision, one of four senses, to understand information (Ward et al., 2010). Visualization can be defined as "the communication of information using graphical representations" (Ward et al., 2010).

The visualization process, described in (Ward et al., 2010), begins with an analysis on the type of data that is going to be used and the type of information that is expected to be extracted. The data values, or attributes derived from the data, are used to define graphical objects. Then, some interactive controls should be provided for viewing and mapping of the variables (attributes or parameters). This process is illustrated in Figure 10.

The visualization field is applied in several disciplines like human-computer interaction, perceptual psychology, databases, statistics and data mining, among others (Ward et al., 2010). A visualization should be used when a need to augment human capabilities exists, rather than replace people with computational decision-making methods (Van Wijk, 2005).

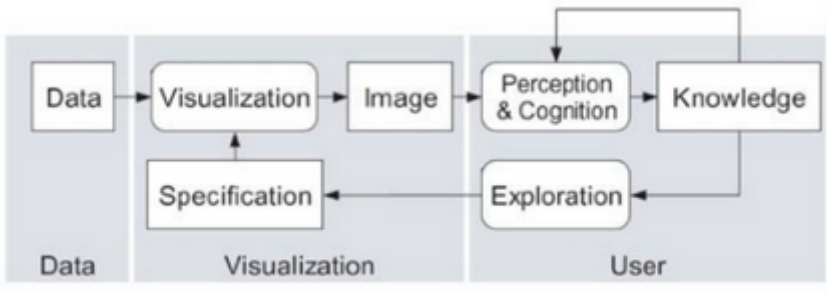


Figure 11: Visualization discovery process (from (Santos, 2014))

There is no formula for guaranteeing the effectiveness of a given visualization since different users will have differing opinions on each visualization (Ward et al., 2010). So, validating the effectiveness of a design in necessary but difficult (Munzner, 2014). A visualization can be analyzed through answering why the visualization is necessary for the user, what data is shown and how the idiom is designed (Munzner, 2014).

Information Visualization

The visualization of data allows obtaining insight in an efficient and effective way, using the human visual system to detect interesting features and patterns in a short time (Van Wijk, 2005). Information visualization is defined as "... the use of computer supported, interactive, visual representations of abstract data to amplify cognition" (Card et al., 1999).

Information visualization differs from a simple visualization as it allows to use techniques in order to visualize complex features in data, abstracting information from the data (Chen et al., 2009), represented in Figure 11. Some of these techniques use information captured in the visualization process to improve visualization efficiency and effectiveness. The information can be about the input data, the results, the process or the user's perceptions (Chen et al., 2009).

In the real world, this means that users can explore data in real time and discover specific patterns visually(Meyer, 2010). Information visualization applications are interactive, dynamic and embed details in context, providing the user with an overview first, then with zoom and filter the visualized information is reduced, and finally details are accessible on demand (Meyer, 2010). This describes the information visualization principle "overview first, zoom in and filter, then show details on demand" (Shneiderman, 1996).

Data types used in visualization techniques can be divided into six categories (Ward et al., 2010): One-dimensional, Two-dimensional, Multidimensional, Text and Hypertext, Hierarchies and Graphs, and Algorithms and Software.

Visualization techniques can be divided into the following groups (Ward et al., 2010): Standard 2D/3D, Geometrically Transformed, Icon-based, Dense pixel, and Stacked.

The use of these data types and techniques will improve visualizations if used properly. Some problems might arise if they don't follow the guidelines (Santos, 2014):

- **Color** — helps distinguish elements easier. Color maps should be used to represent data, avoiding

garish colors and wide vibrant areas.

- **Equivalency** — representations can be informationally and computationally equivalent. A representation is informally equivalent if is verifiable that another representation comprises all its information, and vice-versa. A representation is computationally equivalent if is verifiable that both representations are informally equivalent. This allows to draw conclusions effortlessly between different representations.
- **Interaction** — the user should interact with the visualization in order to ease the analytical process.
- **2D vs 3D** — the use of 3D representations can lead to spatial and occlusion problems. 2D should be used by default, and 3D only in special cases.

As a conclusion, with the increasing size of and complexity of data, the use of information has a support for visualizations will eventually become the standard approach (Chen et al., 2009).

4.8 Discussion

Based on the considered literature, the author identified the need for an innovative artifact that allows the visualization and management of EA transformations over time in a holistic manner as a way to support EAM.

In regard to the described EAFs, the author concluded that most frameworks do not stipulate any rule or guideline for handling the evolution of the architecture. Only TOGAF, address this issue in a more detailed manner but even TOGAF does not objectively specify any method, or artifact for undertaking such process.

After the EA tool analysis, some limitations were identified regarding the flexibility of information models in EAM. Most of current EA tools already address the concern of adapting the EA information model to the organization's specific needs, but do not comprise model migration scenarios. After performing changes to the information model, the affected EA models still need to be manually updated to ensure the information remains coherent. This concern still poses a limitation when addressing EA as an iterative process evolves over time. Therefore, proving the need for an innovative artifact that addresses these issues.

Integrating such artifact into an EA tool requires the specification and implementation of a structured EAM program. By coordinating the EAM program with the organization's needs as a continuous iterative process, a coherent informational environment can be reached. To achieve this alignment, it is necessary to consider an EA Meta-meta-model (described in the section 5) that provides the relevant concepts to address the evolution of the organization's structure over time, thus overcoming some of the problems associated to EAM failure (Denert-Stiftungslehrstuhl, 2008).

The EA Meta-meta-model incorporates any EA framework or notation, ensuring the EAM program success as it can be adapted to each organization's individual needs. The considered meta-meta-model is aligned with Buckl's information model concepts to manage EA transformation (Buckl et al., 2009), representing a transformation project as a set of EA Transformations.

(Roth and Matthes, 2014) four-layered conceptual design to visualize model differences in meta-models

accomplishes some of the author's proposed interface goals. Despite being thorough, the layered visualization has some limitations. The user can only visualize changes, in contrast to the interface the author proposes which aims to empower stakeholders, allowing them to perform changes in real time and oversee their impact. Another disadvantage is that the drill-down layered style is rather complex for everyday use, requiring navigation and filtering to visualize the changes. Hence, since some of the features already exist in current EA Tools, the proposed interface uses a single view to compare the differences between two meta-models (from the *as-is* to the *to-be* state).

Concerning the analysis made on the evolution of meta-models, the EA migration rules were considered for adaptation as a way of managing the EA evolution of an organization. (Silva et al., 2016) proposed set of EA migration rules to automate the migration process during an EA transformation project were incorporated. Besides (Silva et al., 2016) approach, the bibliography seems to be scarce within the EAM area of research as opposed to the model-driven community. (Silva et al., 2016) approach follows (Tribolet et al., 2014) lifecycle principle to manage EA objects, ensuring no EA object is removed but rather changes its state. (Tribolet et al., 2014) approach ensures any set of EA Transformations can be reversed if needed.

The use of information visualization techniques contributes with the necessary techniques and guidelines to design the new interface to visualize and manage EA evolution in organizations, as its main purpose focuses on making information more easily understood and accessible for the users, focusing on supporting task efficiency and effectiveness. Next, the author provides a specification of the proposal.

Chapter 5

Proposal

This chapter corresponds to the ”Define objectives of a solution” and ”Design and Development” steps in the DSRM process. A description of the author’s approach to solve the problem is explained, the proposal objective’s are defined, and an EA meta-model evolution interface to support EAM is proposed.

Considering the problem described in Section 3, the absence of support to visualize and manage EA evolution in organizations as an iterative process, the author suggests an interface to visualize and manage EA meta-model evolution over time. The interface’s contribution is to support organizations with an appropriate tool to manage EA evolution through time, presenting up-to-date consistent views of the past, and the ability to predict future changes, avoiding inconsistencies and improving the quality and reliability of enterprise data.

Following (Silva et al., 2016) research, the outcome of using the interface to manage EA evolution will be a set of EA migration rules, derived from the gap analysis between the *as-is* and *to-be* states of the meta-model, thus co-evolving the EA meta-model and respective EA model instances.

Using a single coherent environment to visualize and manage information within an enterprise, contributes to stakeholders with a consistent view of the information, and how it is communicated across the enterprise wide-context, using it as an asset to improve the enterprise’s value and, as a tool, for guidance and support.

Therefore, the proposed interface aims to accomplish the following objective: **provide stakeholders with a consistent environment to visualize and manage the evolution of the organization’s EA meta-model**, thus, turning EA development projects more efficient.

Initially, a description of the initial approach to solve the problem is presented. Then, the EA meta-meta-model to support the EA evolution interface is introduced, followed by the EA migration rules subsection which states the EA migration rules adapted by the proposed interface. Then, the section ”Design and Development” presents the reasoning behind the idioms and tasks chosen for the interface. Finally, the EA meta-model evolution interface is explained in detail.

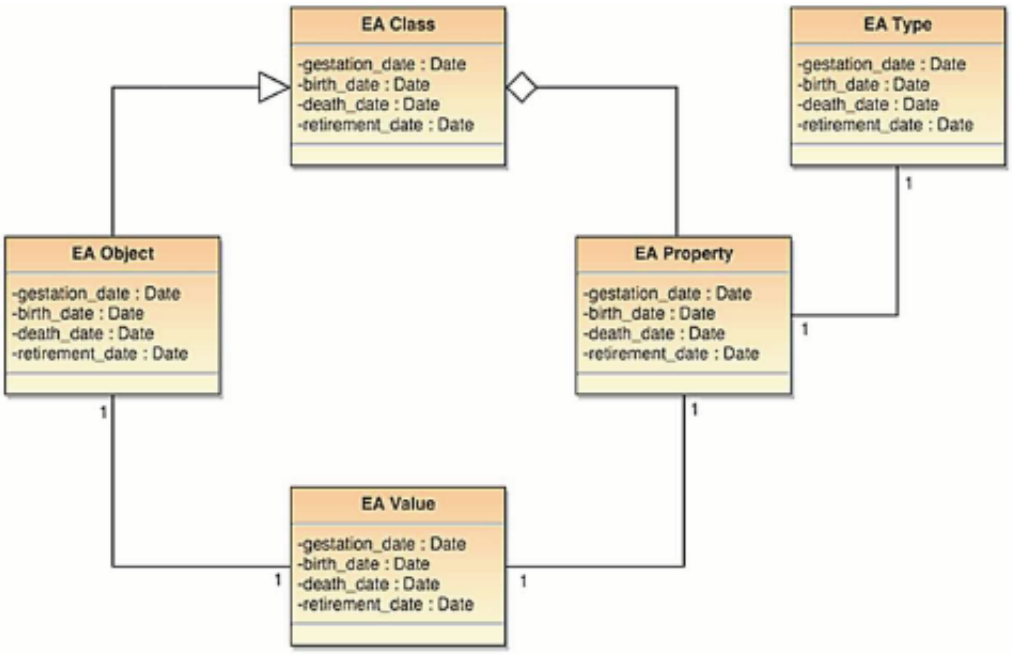


Figure 12: EAMS Meta-meta-model

5.1 Initial Approach

After the related work analysis (see Chapter 4), several limitations in EA tools were identified in regard to the lack of support for EA evolution. The author's initial approach tried to solve part of the problem through the creation of new viewpoints to visualize EA evolution over time.

The first step was to explore the existing features of the used EA tool. Enterprise Architecture Management System (EAMS) is an automation based solution to enable efficient management of EAs. EAMS provides a simplified environment for information harvesting and input, aggregating all needed EA information from the relevant sources inside organizations, so as to automatically generate EA blueprints.

The EA platform supports communication and decision making by providing simple interfaces and blueprints that comply to standards and contain all the business and technical information needed. EAMS capabilities allow to capture information from enterprise repositories, tools, files and human input into a consolidated repository, providing a conciliated view of the organization. Time-travel is supported by having all the information required to generate time related representations that include lifecycle information of new and discontinued assets, creating an integrated vision between the *as-is* and the *to-be* states.

To improve the author's knowledge about EA in practice, the initial objective encompassed the exploration of EAMS's features to solve the identified problem.

As a starting point, EAMS's meta-meta-model was incorporated into the EAMS repository. The model is composed by EA classes and EA types. Each EA class was instantiated into an EA object, with a list of EA properties, each one associated to an EA value from a specific EA type. Lifecycle states and dates are implicit in EA objects and EA properties. Figure 12 illustrates EA meta-meta-model used by EAMS.

Each of the objects that compose the model were then associated to a data type, defining the meta-meta-

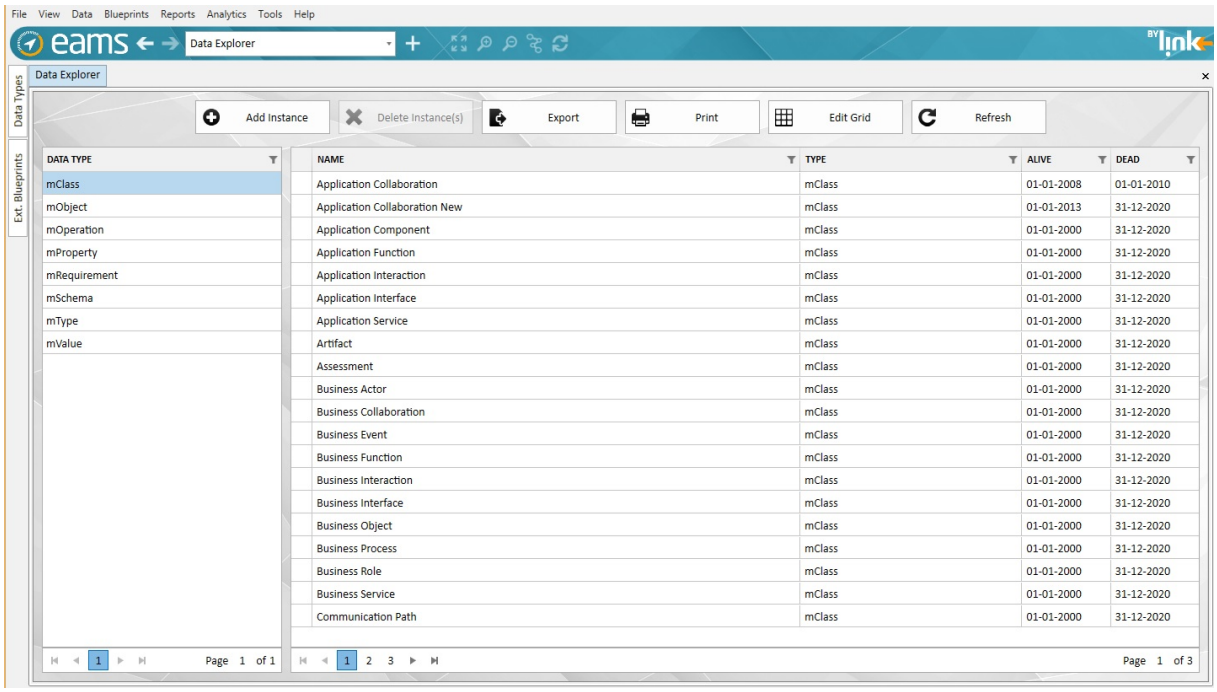


Figure 13: EA Meta-model repository instances

model strucutre in EAMS's repository. Thus, the meta-model can be fully represented by adding EA classes, EA objects, EA properties, and EA types as instances. For this initial approach, the Archimate framework (Jonkers et al., 2014), an architecture description language for enterprise architecture, was selected as the meta-model to instantiate the repository. Archimate was chosen since it is commonly accepted by the IS community and most architectural blueprints in EAMS already use it as a guideline.

The meta-meta-model instances were incorporated into the repository through EAMS's importation capabilities, using a XML file as the source. A text file with Archimate's specification was parsed, allowing to create new files following EAMS's specific XML document structure with the meta-model's classes, objects, properties, and types. The XML input files were generated resorting to XSLT language to perform the transformations. Figure 13 shows the meta-meta-model's instances in the EA repository by data type.

With all meta-model components instantiated, the next step was to design the blueprints to visualize the meta-model. A blueprint is composed by several containers, each one associated to query that retrieves the corresponding data instances from the repository. The blueprints incorporate the use of color to represent the data object's instances lifecycle state, taking advantage of EAMS's time-travel feature. Each state (gestation, alive, dead, and retired), folowing (Tribolet et al., 2014) principles, is associated to a date. Selecting different dates in the viewpoint changes the object's lifecycle color highlight accordingly.

The next goal was to map transformations, and their reasoning, to the meta-model components that had lifecycle changes. For that, we used projects to represent a meta-model transformation and requirements to describe the motivation that led to the change, correlating them with the affected meta-model components. These concepts were incorporated into EAMS's repository and then added to the meta-model blueprint.

The new meta-model views are represented in Figure 14 and Figure 15. In Figure 14 a blueprint from the Business Object Class is presented, where the property "acessed by" is going to be removed. In

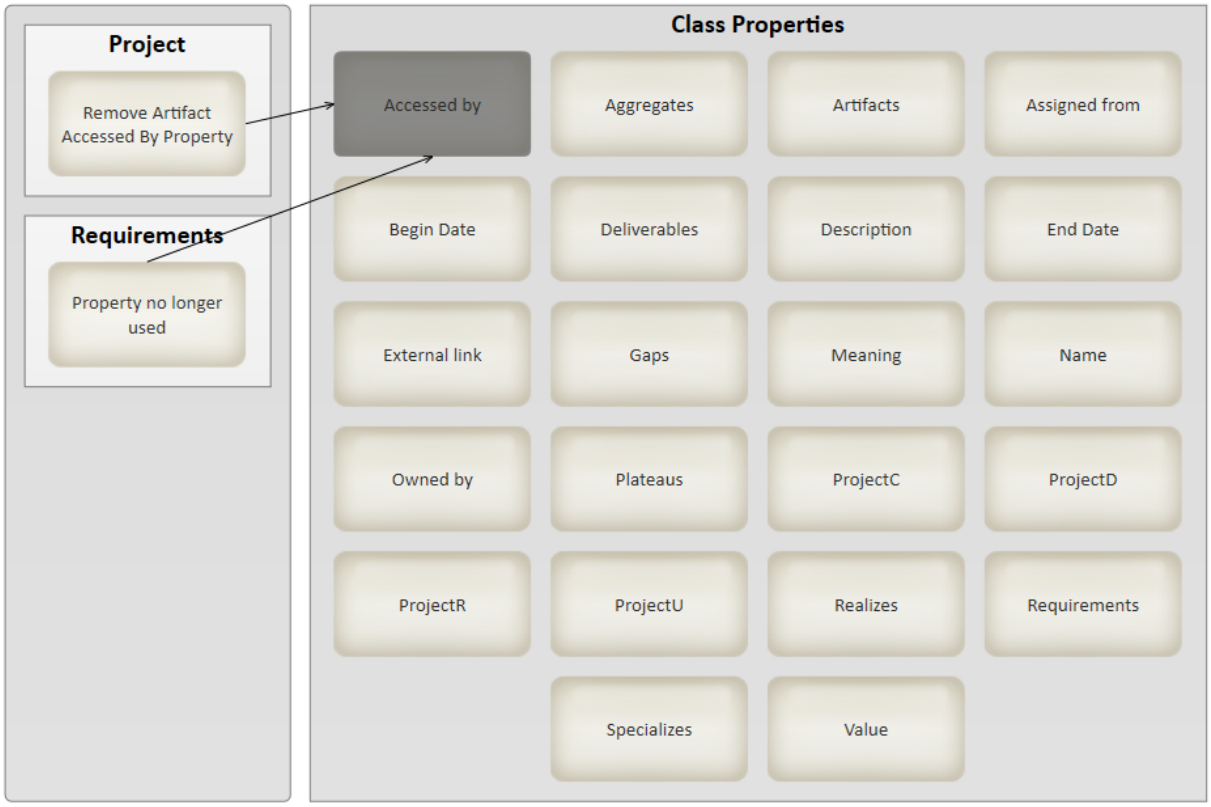


Figure 14: Meta-model properties blueprint with business perspective and lifecycles

Figure 15 two projects, representing the associated transformations, are connected to two meta-model components. The grey component is retired, no longer used, and the green one is gestating, about to be alive. An additional view of the project's perspective was created, showing only the meta-model changed components by project, as Figure 16 illustrates.

EAMS features enabled the author to visualize the transformations and requirements associated to each changed component in the meta-model, achieving the initial objective.

However, this first solution did not enable the management of EA meta-model evolution through time, only to visualize the changes performed and their reasoning, since all transformations had to be made beforehand in the repository.

Moreover, another problem found was that after creating and integrating a meta-model in EAMS, when posterior changes to the meta-model were required the repository had to be rebuilt from scratch. Performing changes to a meta-model becomes then a very time consuming and error prone task, since the new meta-model has to be imported again to an empty repository. Having to perform minor changes to the meta-model, and constantly rebuilding the repository, becomes then a task to avoid, being a constraint for a successful EAM program. These problems confirmed the need for a new interface in EAMS to facilitate these tasks.

In order to implement our solution, a specification for a generic model that integrates EA meta-model evolution concepts was required. Besides being able to visualize EA meta-model evolution, the meta-model models migration must also be taken into account, requiring an extension to incorporate EA transformation principles. The next section defines EA meta-meta-model specification for the interface with support for meta-model and meta-model's model evolution and migration.

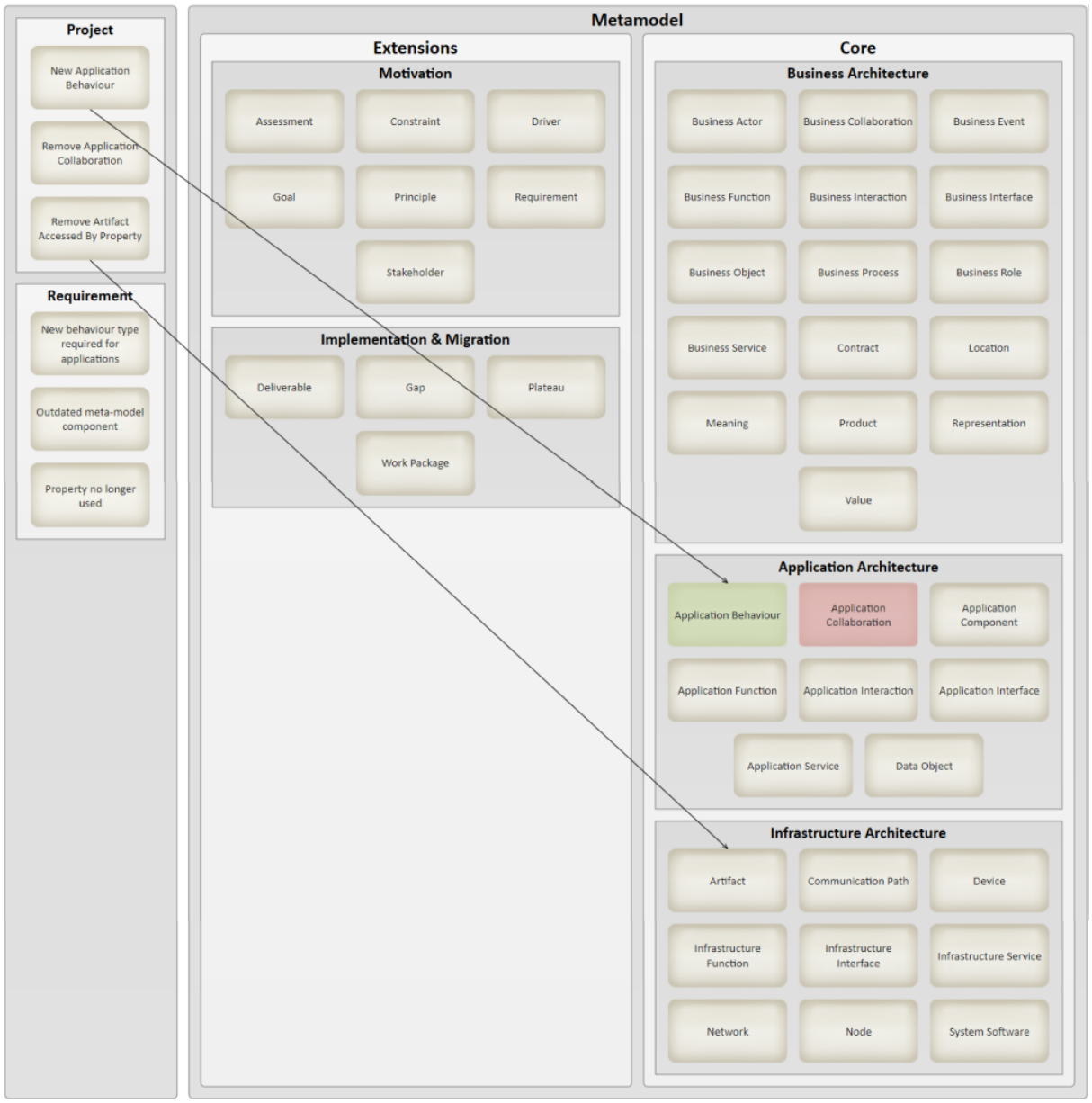


Figure 15: Meta-model blueprint with business perspective and lifecycles

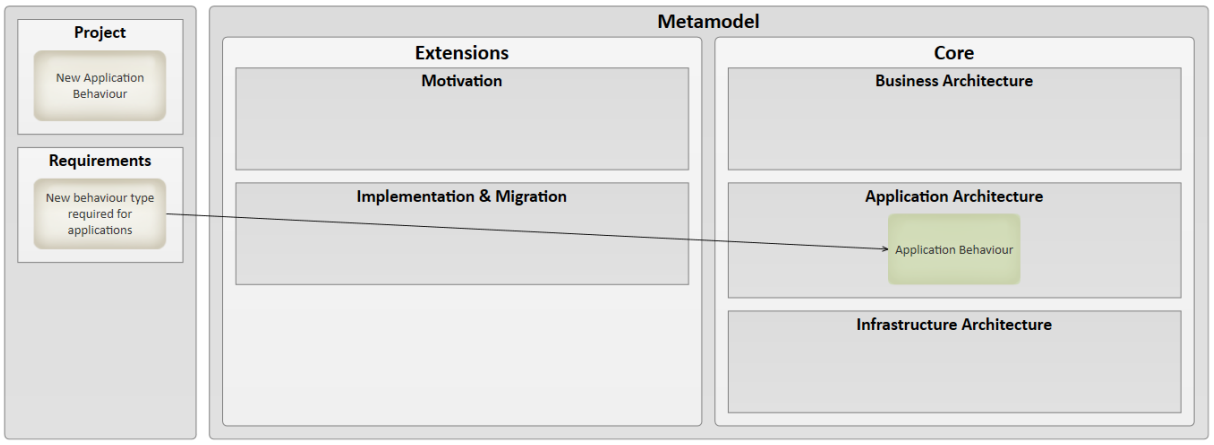


Figure 16: Business perspective blueprint with meta-model changes

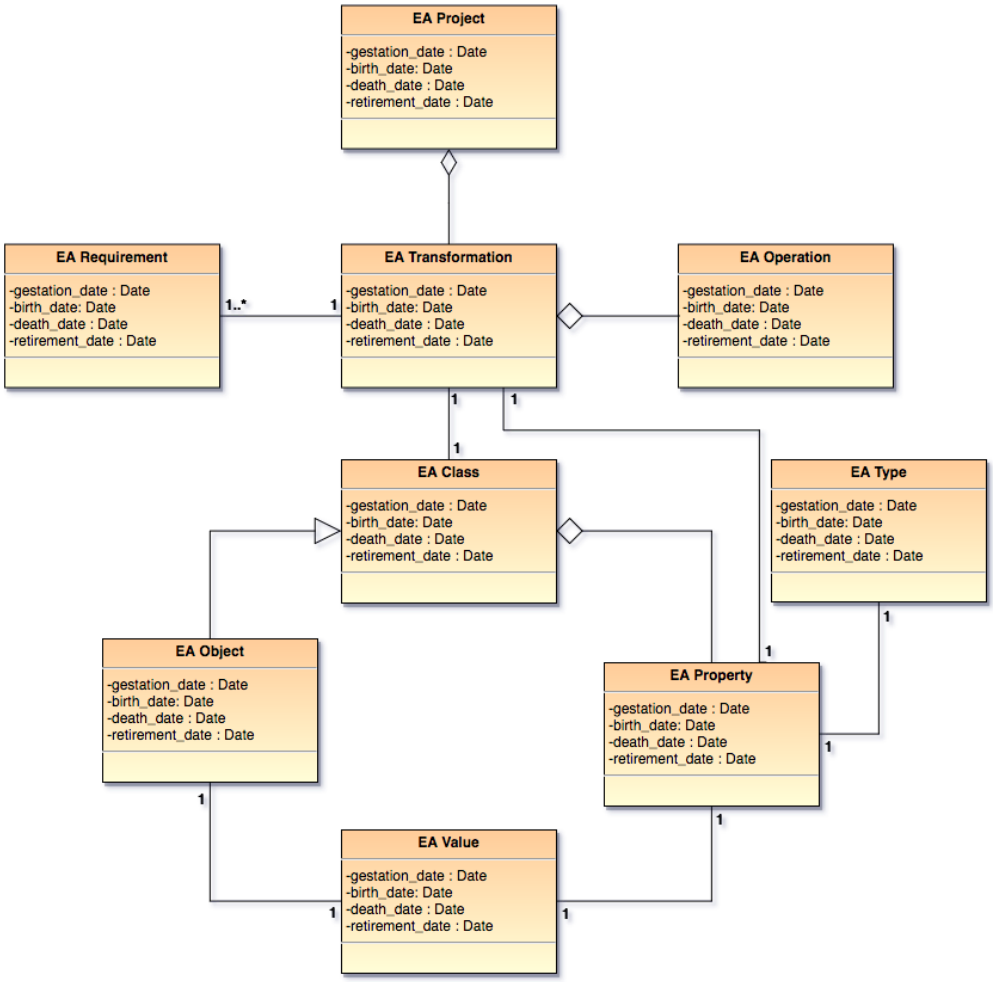


Figure 17: EA meta-meta-model (Complete view)

5.2 EA Evolution Meta-meta-model

The integration of EA evolution into an EA tool requires the specification and implementation of a structured EAM program. After considering all problems associated with EAM, analyzed in more depth in Section 4, one of the most important conclusions was that an EAM approach should be adapted to the organizations needs, make it a continuous iterative process, so that it provides a coherent informational environment. Therefore, a generic EA meta-meta-model to incorporate different organization's meta-models seemed as a prerequisite for success.

The EA evolution meta-meta-model, illustrated in Figure 17, defines the set of concepts that will be used by the proposed interface. The model incorporates the necessary concepts to represent any EA meta-model, being a generic approach for an EA evolution representation.

EA meta-models and respective models are specified by a set of EA Classes and EA Types. An EA Class has a variable list of EA Properties, each from a given EA Type. EA Objects are instantiated from an EA Class that bounds the EA Properties to what each EA Object may hold at each moment. Finally, EA Values are always assigned to a pair <EA Object, EA Property>. EA Properties can hold sets of EA Values, in particular, EA Properties of an EA Type. The lifecycle states and their respective dates are kept for each concept.

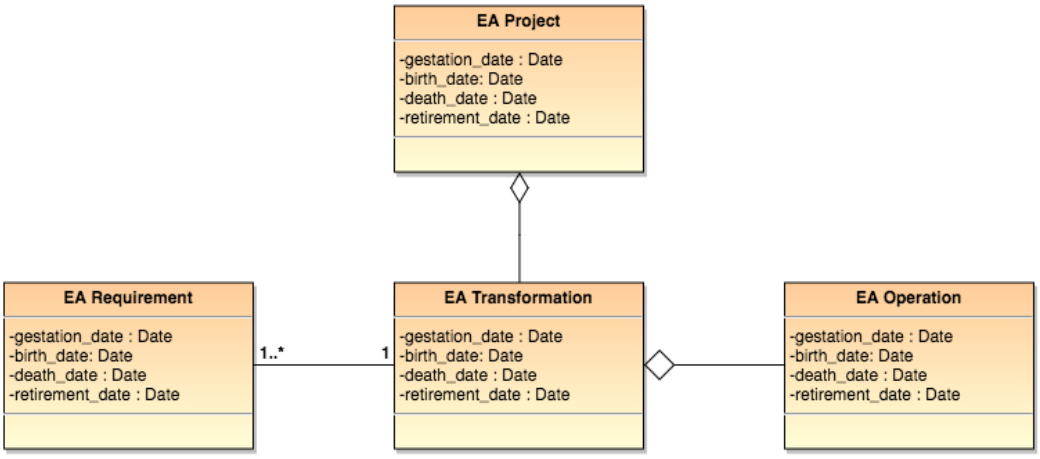


Figure 18: EA meta-meta-model (EA evolution concepts)

In addition, to manage the evolution of an organization's meta-model over time, four additional concepts were added to the EA meta-meta-model, as illustrated in Figure 18.

An EA Operation represents a change made to the meta-model, from a pre-defined set of operations. The EA Operation defines a single transformation operation (or migration rule), e.g. Introduce EA Property, Eliminate EA Class, Eliminate EA Property, etc. An EA Transformation is composed by one or more EA Operations. Since organizations may require several transformations to be applied simultaneously, the concept of EA Project was incorporated. Each EA Transformation is related to a business requirement for the organization, so EA Requirement incorporates the reasoning behind any EA Transformation. Each EA Transformation can be associated to an EA Class or EA property.

Although we can now accomplish a complete and comprehensive specification of the meta-model, and associated transformations, the transformation of the meta-model itself, and migration of the associated models, still need to be accounted for. The next section describes the set of rules to migrate EA models which will be used as support for the interface.

5.3 EA migration rules

Recent approaches seek to automate and improve EA practice within organizations by employing EAM tools. Thus, evolving the organization's EA meta-model is a consequence of fulfilling such initiatives. Currently, the migration of EA models, that conform to a specific EA meta-model transformation, is a manual task in which EA data that corresponds to the actual models is gathered and the models redesigned. This results in an error-prone and time-consuming task.

To address the migration concerns of EA models, a set of nine migration rules were defined. Each rule aims to automate the EA migration process as well as providing a complete set of steps throughout the migration of EA data.

These migration rules aim to achieve two main objectives:

1. Automate the EA migration process
2. Provide a complete set of steps throughout the migration of EA data

Each migration rule followed an adaptation of Wachsmuth's co-adaptation transformations (Wachsmuth, 2007) for MOF compliant meta-model evolutions combined with Tribolet's research (Tribolet et al., 2014) on enterprise transformations and principles for understanding the dynamics of organizations.

By consolidating both approaches, (Silva et al., 2016) contributed to an automatic migration process involving various organizational elements, affected during an EA development project, based on their respective lifecycle states (gestating, alive, dead, and retired). Thus, the issue of manually migrating EA models with all its inconvenience (error-prone and time intensive) can be mitigated through the implementation of the proposed migration rules within any EA documentation tool that follows an object-oriented approach. A description of each rule is presented below :

- **Introduce EA Property** — Creates an EA property with a relationship to an EA class. It also sets a gestation and a birth date for the property corresponding to the dates when the property changes its state to GESTATING and ALIVE respectively. The creation of EA values corresponding to the introduced EA property is not obligatory. However, if the introduced EA Property is marked as mandatory, the introduction of the respective EA values is required. EA values referencing the introduced EA property will have a default value according to the property's EA type. For example, if the property's EA type is of type integer, the default value is 0.
- **Eliminate EA Property** — Assigns a death and a retirement date to an EA property. In addition, it sets the respective death and retirement dates of all EA values referencing that property.
- **Eliminate EA Class** — Assigns a death date and a retirement date to an EA class. Then, it sets the respective death and retirement dates of all EA objects referencing the EA class and executes the Eliminate EA Property migration rule on each EA Property with a reference to that class.
- **Change EA Object Class** — Renames an EA class. Then, updates all EA objects, EA properties and EA values referencing that class as well as the gestation and birth dates of all the involved elements.
- **Change EA Property Type** — Changes the EA type referencing an EA property. The rule verifies if the changed EA Type exists. If so, updates the property's EA Type with the existing one; otherwise it creates a new EA Type instance and associates the new type to the property. Finally, the rule updates all EA values associated with the EA property accordingly as well as the gestation and birth dates. If the new EA type is not a reference, each EA value is assigned a default value.
- **Move EA Property** — Moves an EA property from a source EA class to a destination EA class. Statement-wise, it involves executing the Eliminate EA Property migration rule followed by the Introduce EA Property rule.
- **Inline EA Class** — Collapses all EA classes that are referenced by an EA property into the EA class containing that same EA property. Then, move all properties of each referenced EA class to the target EA class. In the end, it eliminates each referenced EA class and all related instances (EA Objects, EA Properties, and EA Values).
- **Association To EA Class** — Transforms a relationship property between an EA class and a group of EA classes into a new EA class. Then, it updates all values referencing that property with new EA objects, which are instances of the new EA class.
- **EA Class To Association** — Transforms an EA class into a set of EA properties, each one being a relationship between that EA class and all related EA classes. In the end, it eliminates the source EA class and all related instances (EA Objects, EA Properties, and EA Values).

Besides the statements that automate the migration of EA models, each migration rule is composed of a set of pre- and post-conditions that guarantee the correctness and coherence of the data throughout the migration process.

The author adapted five of those migration rules to define six meta-model EA operations used by the proposed interface. The adapted migration rules were: **Introduce EA Property, Eliminate EA Property, Eliminate EA Class, Change EA Object Class, Change EA Property Type**.

These rules can be decomposed into 6 low level operations: **Create Class, Update Class, Delete Class, Create Property, Update Property, and Delete Property**. The interface, described in the section 5.5, restricts users to these low level operations as they correspond to the transformations applied when manipulating the meta-model's classes and properties.

Next, the author presents the design and implementation phase, describing the initial work, the design and implementation choices, and the proposed interface.

5.4 Interface Specification and Design

This subsection describes the specification of the proposed interface, defining the tasks and idioms chosen to represent our information, associating each task's purpose with the questions they intend to answer, and depicting the initial mock-up sketches of the proposed interface. In the end, the final version of the implemented interface is portrayed.

The interface is going to incorporate information visualization principles as a mean to diminish the data analysis complexity and understanding, as people gather more information through the visual analysis of information. In our organizational business context, incorporating these features will also assist and improve the decision making process.

For the suggested interface to achieve a proper information visualization environment, an assessment using (Munzner, 2014) high-level framework was performed. The framework analyses the visualization trying to answer to three central questions: **what** data the user sees, **why** the user intends to use a visualization tool, and **how** the visual encoding and interaction idioms are constructed in terms of design choices. The next subsections describe the dataset, tasks and idioms chosen.

5.4.1 Data set

In order to represent EA meta-model evolution, the interface uses as dataset the EA meta-meta-model data instances. Since the interface will be incorporated into an EA tool, the data set will be accessed through the EA tool's repository.

The interface comprises the following data types: **items, attributes** and **links**. EA Projects, EA transformations, EA requirements, EA Operations and EA meta-model components are represented as **items**, defining an individual discrete entity. A **link** represents a relation between two items, connected through a reference between EA meta-model components and related transformations and requirements. Each item, EA meta-meta-model component, has **attributes** to describe them. These attributes include the EA object's name, description and associated dates. In the particular case of an EA Class, a clas-

sification attribute is also required to define the architectural type. The EA meta-model components' name, classification and description are from the categorical type, in contrast to dates which are ordinal. Relating different items, and their attributes, through links, define the data set type as a **network**.

The dataset availability is dynamic since the dataset information trickles in over the course of the visualization. In our case, a dynamic change can be to add or remove an item (EA transformation or EA requirement).

5.4.2 Tasks

Task abstraction is a generic approach that allows visualization tools to be compared regardless of their domain. Without domain-specific language descriptions, visualization tools share most of the same user goals, sharing a lot of similarities. These tasks can be divided into actions and targets. Actions and targets describe the reasoning behind why the visualization tool is being used, distinguishing different user goals.

The proposed interface aims to accomplish the following tasks:

- **Task 1**: Analyze → Consume → Discover/Present/Enjoy

  Analyze EA meta-model evolution using the visualization interface. Consume existing information, with the goal of discover new knowledge, present information succinctly and effectively, and enjoy the visualization.

- **Task 2**: Analyze → Produce → Derive

  Derive the number of EA Transformations by EA operation type, through the analysis of each EA transformation operation type.

- **Task 3** Search → Lookup ; Query → Identify

  Find an EA meta-model component EA operation type knowing beforehand its classification and name.

- **Task 4**: Search → Lookup; Attributes → One → Distribution

  Assess how EA Transformations are divided by EA operation, depicting the frequency each one occurs.

- **Task 5**: Search → Locate; Query → Identify

  Find a specific EA meta-model component, assessing its EA operation type when the EA meta-model classification is not known beforehand.

- **Task 6**: Search → Browse; Query → Compare.

  Compare several EA meta-model components operation type from a specific classification.

- **Task 7**: Search → Explore; Query → Summarize;

  Overview of the EA meta-model evolution through the analysis of all changes to EA classes and EA properties that compose the EA meta-model.

- **Task 8**: Search → Explore; Query → Summarize; Attributes → All Data → Trends

  Analyze which is the most common EA operation type in the EA meta-model evolution overview.

With the support of these tasks, the author intends to answer the following questions using the proposed interface:

- Which are the differences between the EA meta-model *as-is* and *to-be* states? **(Task 1,6,7)**
- How many EA Transformations create/update/delete EA classes or EA properties? **(Task 1,2,4)**
- What EA operation type is applied to a specific EA meta-model component? **(Task 1,3,5)**
- Which are the EA requirements and EA transformations associated to a specific EA meta-model component? **(Task 1,3,5)**
- What is the most common EA operation type in the EA meta-model evolution? **(Task 1,7,8)**

Answering these questions provides a complete analysis on EA meta-model evolution, accomplishing a detailed description of the changes perform and the reasoning that led to them. The following subsection analyzes how the idioms of the visualization were selected out of a set of design choices.

5.4.3 Idioms

In order to accomplish the objectives set by the tasks, the following idioms were selected: a **Point Chart**, presenting an overview of the EA Transformations by EA Operation type, a **Blueprint viewpoint** with a comparison between the *as-is* and *to-be* state of the EA meta-model, and **two data grid containers** representing the EA transformations and EA requirements.

The point chart represents each possible EA Transformation by EA Operation in a determined date. A point seemed appropriate to represent an EA transformation, as applying an EA operation to the EA meta-model is an atomic operation. The point chart uses as encoding a color hue, determining the EA operation purpose by category, and the data point's size for the number of EA Transformations that apply the EA Operation. All EA operation data points are aligned by their date. Manipulation is supported, selecting a data point will highlight all related EA objects.

The blueprint's viewpoint allows to compare the EA meta-model *as-is* and *to-be* states. Each EA meta-model class is organized by their classification, and EA properties by the class they belong to. Both EA Classes and EA properties are ordered by alphabetically. A color hue is used to encode the EA operation category associated to the EA meta-model component. The viewpoint allows to select EA meta-model components, highlighting related EA objects, and to navigate, using zooming and exploration capabilities. Comparison between the *as-is* and *to-be* state is achieved by juxtapose both states or partitioning the viewpoint into two views (one for each state).

The data grid containers enumerate all EA transformations and EA requirements, using a text label as encoding. Data from both data grids can be reduced, using as a filter specific properties. Manipulation is supported by selection, highlighting related EA objects in the other idioms.

5.4.4 Mock-up sketch

After the initial specification, some mock-up sketches were drawn in order to find the best design to incorporate every idiom. Figure 19 and Figure 20 illustrates the initial and final interface mock-up sketches, respectively.

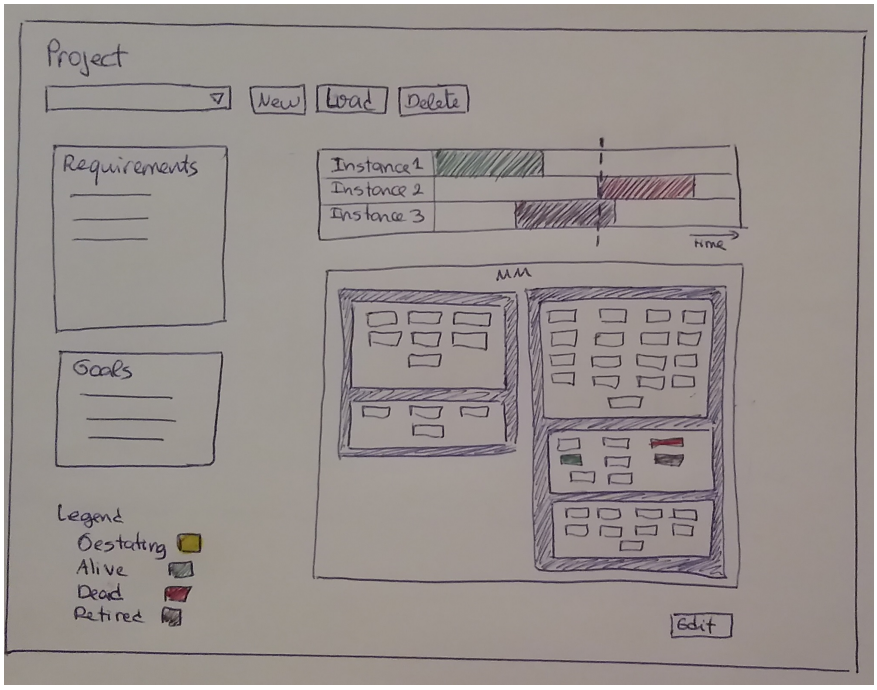


Figure 19: Initial mockup sketch

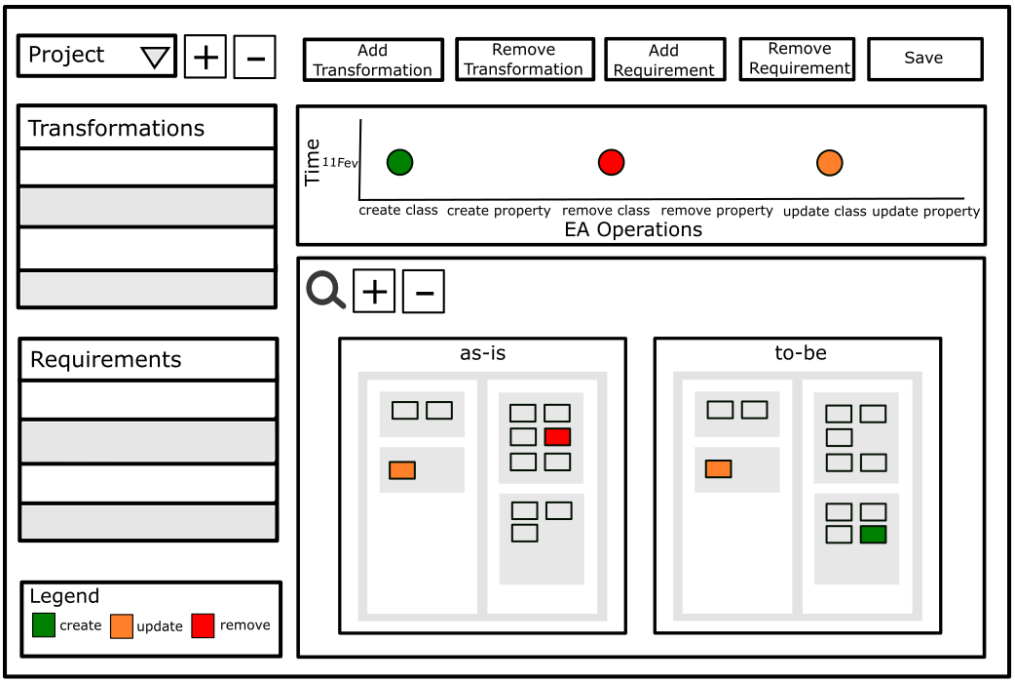


Figure 20: Final mockup sketch

The final mock-up sketch incorporates all idioms presented in the previous subsection. From the first sketch, some components were added to manipulate EA evolution more effectively.

The EA Project selection box was complemented with the respective add and remove buttons, allowing to manage EA evolution as EA projects. Also, two additional buttons were added to create and remove EA requirements and EA transformations separately. This design choice takes into account the highlight feature between all components. Thus, a generic add/remove button would be ambiguous. The viewpoint required navigation capabilities, so two zoom buttons were incorporated as well as two buttons to change between the gap analysis and single EA meta-model views.

5.5 EA Meta-model Evolution Interface

In this subsection a complete description of the proposed EA meta-model evolution interface is presented. Initially an overview of the interface is depicted, afterwards an explanation of some of the design choices to incorporate the interface in the vendor-specific EA tool is given. Finally, an overview of the interface, and a description of each of its components features, is presented.

The proposed interface addresses the management of EA evolution by using a set of EA transformations that alter the current state of an organization's EA meta-model. Those transformations are composed of specific operations that change the lifecycle stages of the meta-model elements: classes and properties. Afterwards, the impact of those modifications can be seen in the meta-model's viewpoint.

First, the EA meta-meta-model, described in subsection 5.2, was incorporated into the EA tool, creating a data type for each element, with the necessary properties. Each data type instantiation depicts a meta-model element. The initial data importation defines the initial EA meta-model, providing a basis for the EA meta-model evolution. The data importation followed the same steps as in the initial approach, described in section 5.1.

The initial interface view is empty, without any of its components being initialized. Figure 21 illustrates the initial view.

The interface allows to create, delete and save EA Projects, thus allowing the representation of different scenarios regarding the organization's future landscape. Initially, after creating an EA project, since no transformations exist, the viewpoint presents the original EA meta-model, not showing any differences between the *as-is* and *to-be* states, as expected.

Each EA project is associated to EA Transformations and EA Requirements, presented in two data grid containers. The top right buttons of the interface accomplish the EA evolution, allowing to add or remove EA transformations and EA requirements. Figure 22 shows the interface, populated with an EA project, after adding several transformations as an example.

Adding a transformation opens a new window, with the required fields to properly define any type of EA transformations. The Figure 23 shows the view to add a transformation. Every field's information is verified before creating the transformation. Each transformation may require different attributes according to their operation type. Requirements are associated to transformations through this window, needing to be created beforehand. Figure 24 shows the window to create a new requirement. Removing a transformation or a requirement is achieved by selecting it in the data grid and then, selecting the remove button

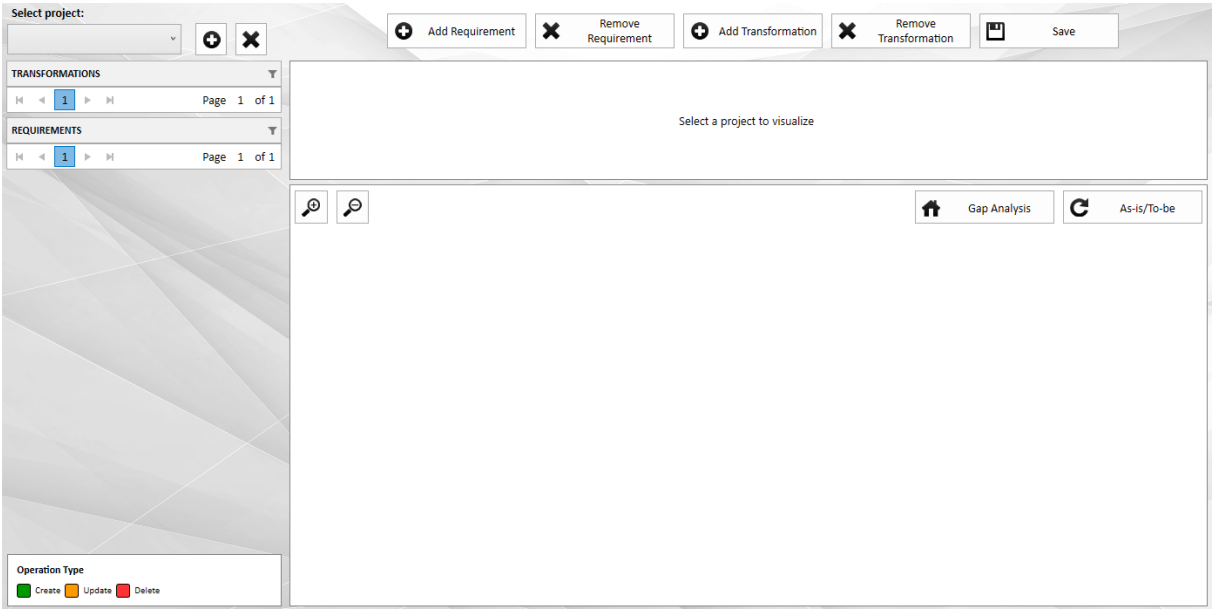


Figure 21: EA Meta-model evolution interface (empty)

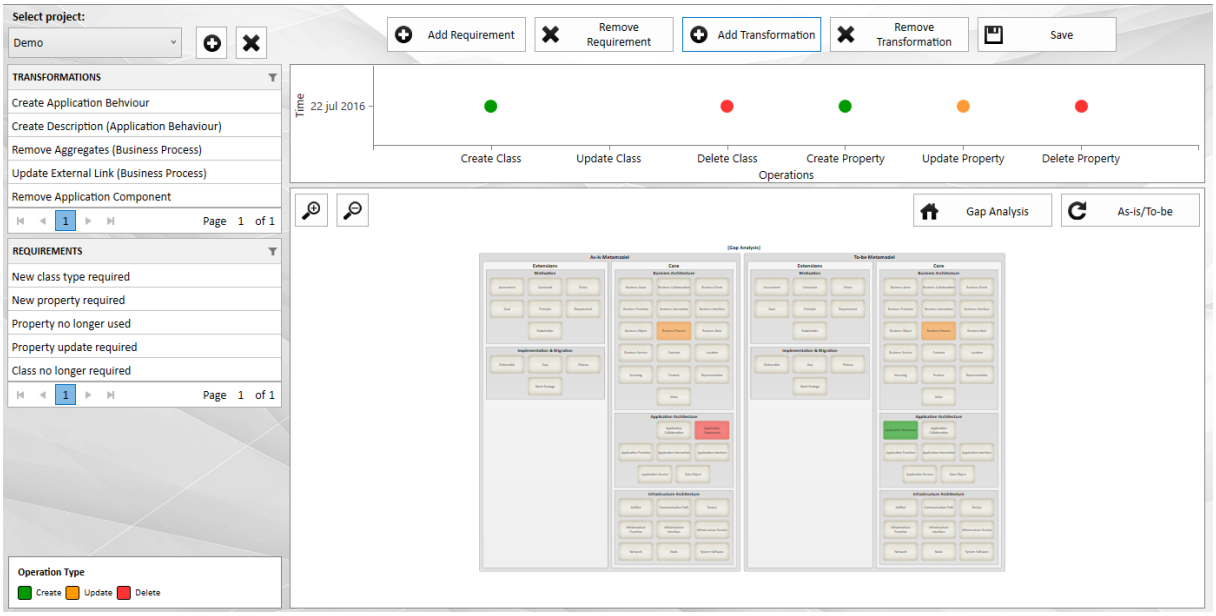


Figure 22: EA Meta-model evolution interface

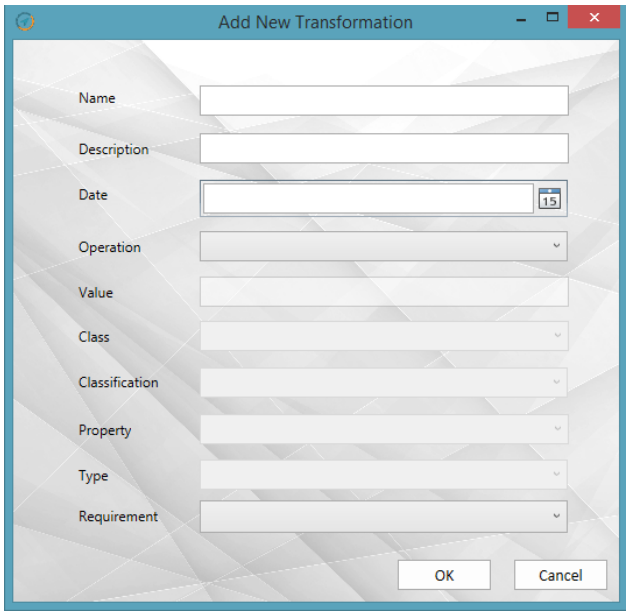


Figure 23: New transformation window

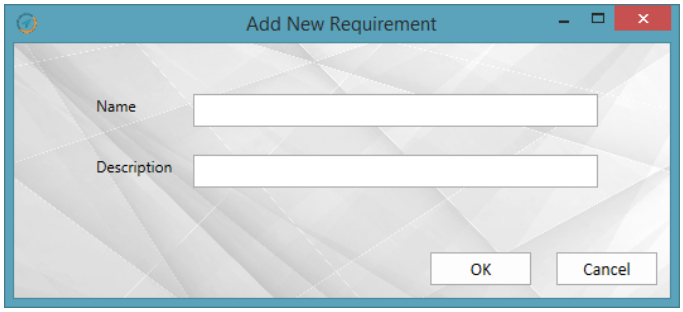


Figure 24: New requirement window

accordingly. Separate buttons to remove transformations and properties were required, as both can be simultaneously selected.

Each operation that creates a class or property, through a transformation, is updated immediately to the repository. In contrast, to maintain the environment coherence, updating or removing a class or property has no immediate impact, changing only their state afterwards through the application of the migration rules, previously specified in the subsection 5.3. To ensure that after removing a transformation from the interface the meta-model has no incorrect classes or properties instances, the save button performs a verification to ensure the data coherence. This verification compares the interface's local meta-model instances with the repository instances, removing any incoherent information from the repository.

Each EA Transformation is associated to one of six EA Operations (Create EA Class, Create EA Property, Update EA Class, Update EA Property, Remove EA Class and Remove EA Property). Those operations are represented in the Point Chart (Figure 25) at the top of the interface.

The Point Chart lays out an overview of the number of EA Transformations that will be applied to the EA meta-model in a predetermined date. The x-axis represents each possible operation type, whereas the y-axis determines the date when each operation will take place. Each data point is associated to a color, illustrating its purpose when applied to an EA Class or EA Property, i.e. red color means removal, green corresponds to creation and orange signifies an update will occur. The user can visualize the

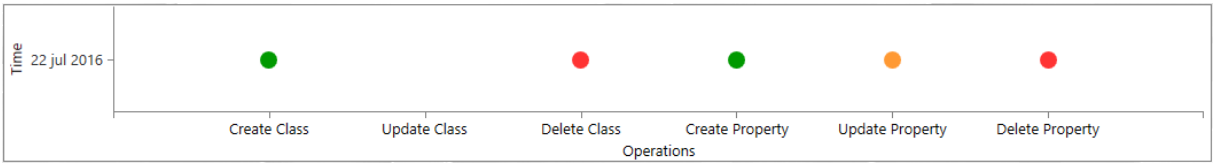


Figure 25: Point chart

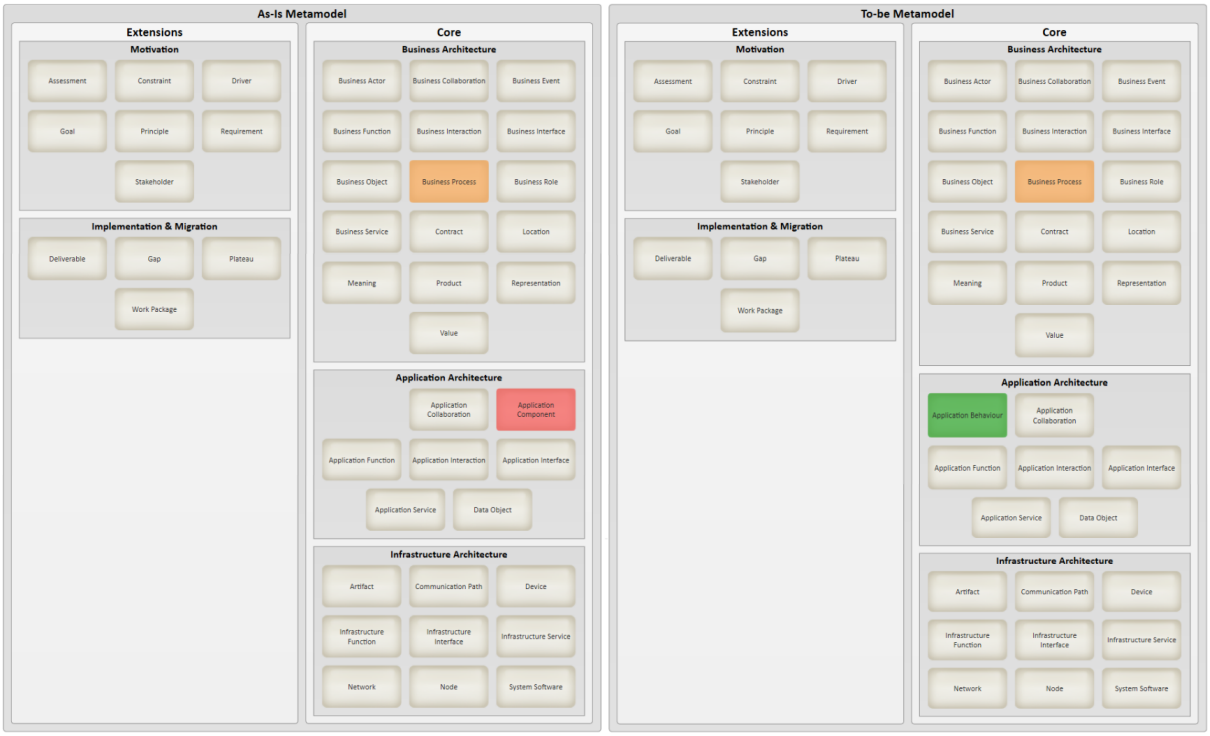


Figure 26: EA Meta-model viewpoint

categories through the legend in the left bottom corner. The data point's size pictures the number of EA Transformations from the same operation type that will occur in that date. The concrete number of transformations in each data point is provided in detail trough a tooltip.

In the right bottom, a viewpoint illustrates a comparison between the EA meta-model *as-is* state, i.e., before the execution of the EA transformations, and the EA meta-model *to-be* state, i.e., after applying the EA Transformations. The viewpoint (Figure 26) is dynamic, since adding or removing EA Transformations will update the viewpoint, displaying the impact of new transformations in real time.

The component's color is related to the EA transformation's operation type, using the same color scheme as the one described above. The viewpoint is also navigable. For example, a double click navigates to the class's properties viewpoint that compares the *as-is* and *to-be* states. To add a different perspective when comparing the the two states, the button "As-is/To-be" changes the side-by-side comparison to a single view, acting as a quick toggle button between them. The "Gap analysis" button redirects to the side-by-side view. Navigation between the class's views and class's properties views is eased by using the back button in the top left corner of the class's properties viewpoint. Exploration with more detail is supported using a zoom button, in the viewpoint's top right corner.

All interface components are related among them. By selecting one component's object, the related objects in other components are highlighted. This feature provides higher awareness to the stakeholders,

making the interface usage and information analysis easier.

Chapter 6

Demonstration

This section covers the Demonstration phase of the research method. After specifying and implementing the EA evolution interface, the author used the proposed interface in an EA project of a large European financial organization.

The EA project's goal was to plan a first version of the organization's EA, and from there iterate to newer versions according to specific EA requirements. An EA evolution path is stored, from the original specification of the organization's EA meta-model to the current iteration, allowing to justify decisions, through requirements, and the transformation process is eased, automatically updating the required instances.

The interface used the first version of the organization's meta-model as input for the set of EA transformations that were applied with respect to each requirement. Table 2 presents the number of EA transformations created by operation type that address each of the project's requirements in the first iteration.

EA Operations	EA Transformations
Create EA Class	1
Create EA Property	216
Update EA Class	-
Update EA Property	2
Delete EA Class	-
Delete EA Property	103

Table 2: EA transformations by operation type (First iteration)

The EA transformations were incorporated into the EA repository with an XML file, derived from the gap between the original meta-model and its first iteration, using the tool's "Import data" feature. Using EAMS importation capabilities to integrate information directly in the repository can be useful when managing large number of transformations. Then, the EA transformation and EA requirement instances were associated to the organization's EA project. Figure 27 illustrates the organization's EA project instantiated in the proposed interface. Each EA transformation and EA requirement is presented in the left-side data grids.

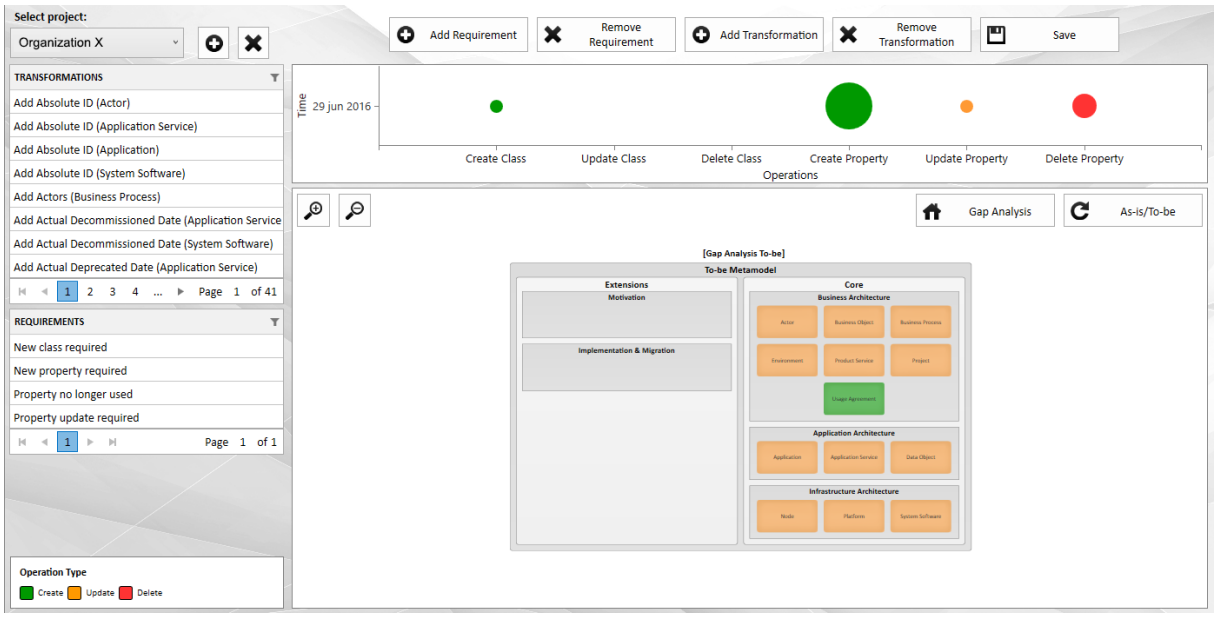


Figure 27: Interface with the organization's EA project (First iteration)

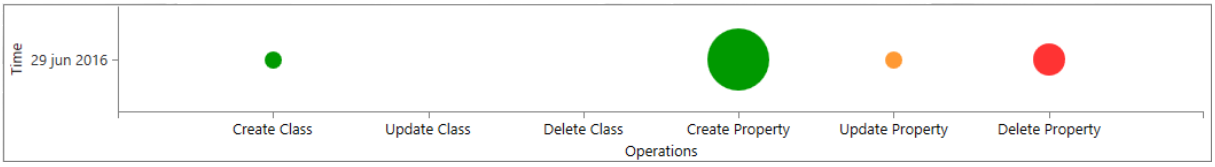


Figure 28: Organization's EA project point chart (First iteration)

An overview of the operations associated to the EA transformations can be visualized in the Point Chart (Figure 28). The data point's size shows the EA transformation's number by operation type. The operation "Create Property" and "Remove Property" comprise almost all of the required EA transformations. In this first iteration, there was a need to redefine the classes' domain, by creating and removing properties according to the organization's needs.

The viewpoint, depicted in Figure 29, clarifies the impact of the EA transformations to the meta-model.

The EA operation type can be deducted, by analyzing the highlights color on each meta-model component. The class highlighted in green (Usage Agreement) was added to the meta-model. All other

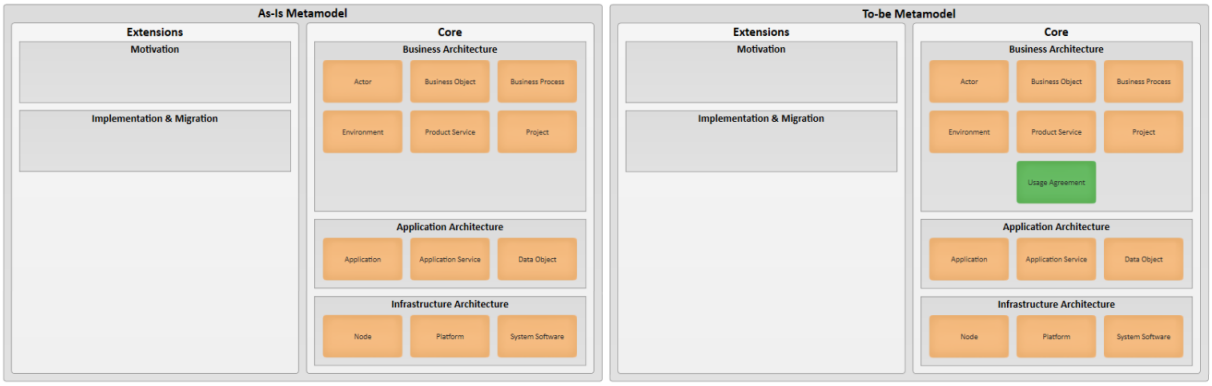


Figure 29: Organization's Meta-model Viewpoint (Gap Analysis)

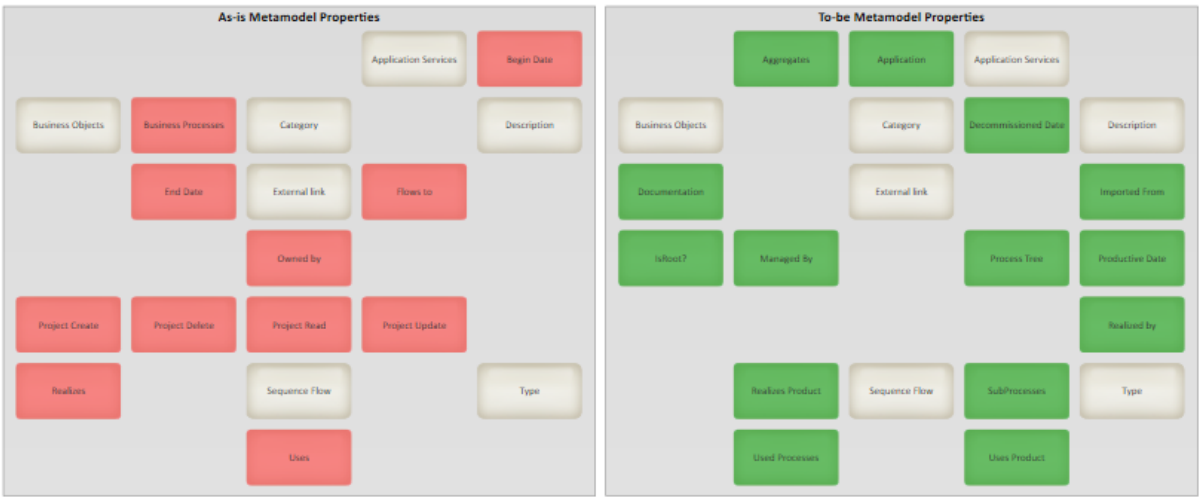


Figure 30: Business Process Class Properties Viewpoint (First Iteration)

classes are highlighted in orange, denoting changes in their properties. Correlating the Point Chart information with changes visualized in the meta-model, the architect can conclude that the majority of these operations are “Create Property” and “Remove Property”. This inference can be verified in Figure 30, displaying as an example the “Business Object” class properties viewpoint.

The EA project’s second iteration changed EA meta-model incorporating new requirements, derived from the organization’s feedback after analyzing the EA meta-model’s from the first iteration. The EA operations, depicted in 3, were applied with resource to the interface’s management features.

EA Operations	**EA Transformations**	
	First Iteration	**Second Iteration**
Create EA Class	1	-
Create EA Property	215	41
Update EA Class	-	-
Update EA Property	2	6
Delete EA Class	-	-
Delete EA Property	100	25

Table 3: EA transformations by operation type (Second iteration)

Table 3 also represents the changes made to the first iteration. Besides adding or removing new EA transformations, several EA transformations were also undone. Figure 31 illustrates the interface after the second iteration’s transformations were added. The two iterations can be compared in the Point Chart by EA operation type, as Figure 32 shows. The EA meta-model’s viewpoint was not affected, since all operations affected the existent classes’ properties, not creating, updating or removing any classes. The EA meta-model became more complete and coherent in this second iteration, evolving alongside the organization’s needs, as the Business Process properties viewpoint demonstrates in Figure 33.

With the interface support, the required changes, in each iteration, to meta-model were fully represented, accomplishing a visual representation for the transformations performed, comprising an overview of the operations applied, and the reasoning behind them, defining the project’s EA evolution.

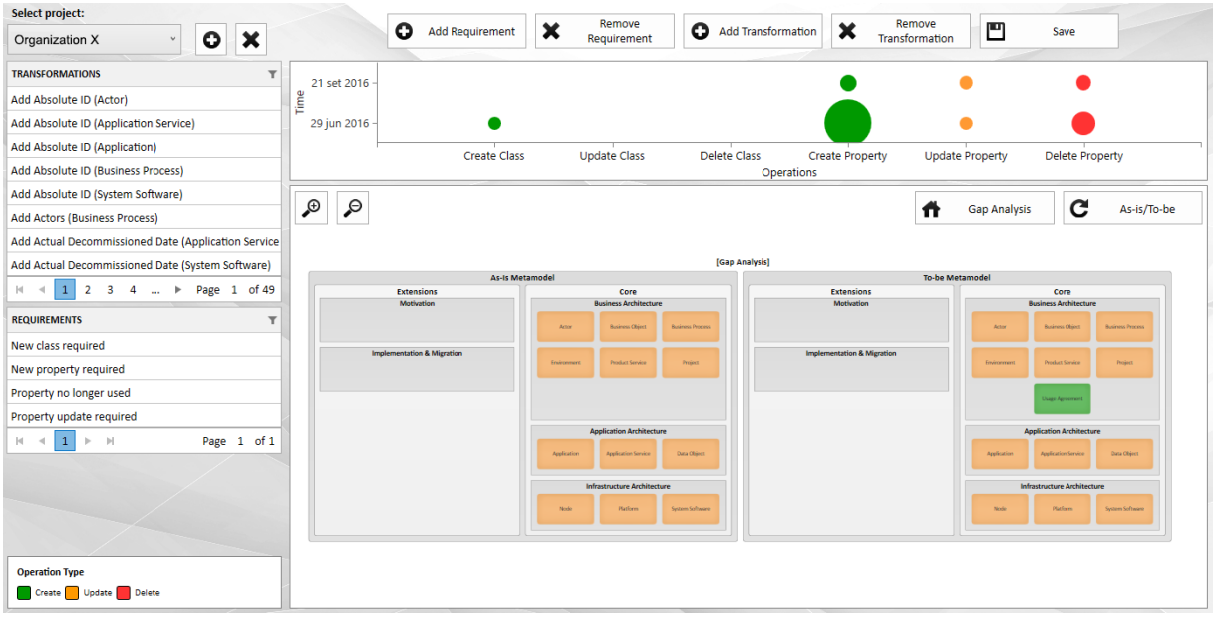


Figure 31: Interface with the organization's EA project (Second iteration)

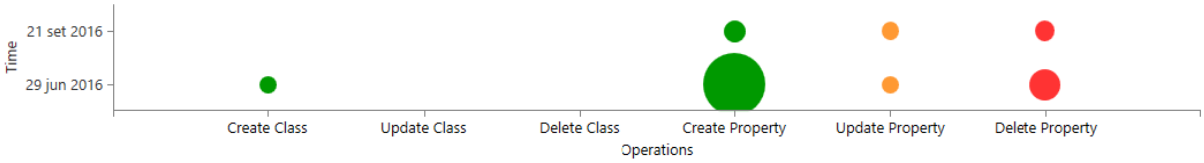


Figure 32: Organization's EA project point chart (Second iteration)

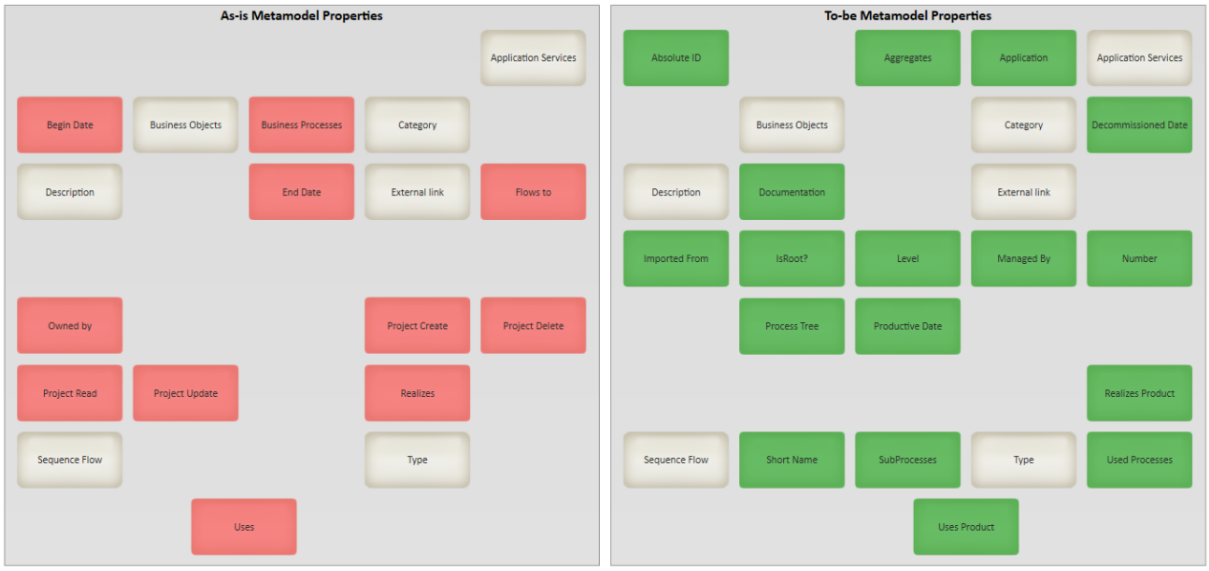


Figure 33: Business Process Class Properties Viewpoint (Second iteration)

Chapter 7

Evaluation

In order to evaluate the proposed EA meta-model evolution interface, a generic evaluation model for IS artifacts and evaluation criteria was used (Prat et al., 2014). This section corresponds to the ”Evaluation” phase of the research method.

(Prat et al., 2014) approach combines the design science research evaluation theory, from the literature analysis, with practice, using design-research papers as sample. The research intended to assess the criteria used in literature to evaluate IS artifacts, investigating which are actually used in published papers, and how should the criteria be structured. The final goal was to discover which evaluation method would emerge as a generic mean to asses IS artifacts.

(Prat et al., 2014) resulting artifact provided a hierarchy of evaluation criteria for IS artifacts, organized according to the dimensions of a system, a model providing a high-level abstraction of evaluation methods, and a set of generic evaluation methods which are instances of this model.

Several evaluation criteria were chosen, taking into account different evaluation contexts. As an evaluation of an instantiation artifact, an analysis on the goal (efficacy), environment consistency with people (utility/ease of use), and environment consistency with the organization (utility) were selected. Figure 34 highlights the criteria chosen from the IS evaluation criteria hierarchy.

With the use of the generic evaluation model to assess the previously described criteria, the following generic methods to evaluate the criteria were instantiated (Table 4):

Description	Assessed criterion	Form of evaluation	Secondary Participants	Level of evaluation	Relativeness of Evaluation
Demonstration of the use of the artifact with a real example	Goal / Efficacy	Analysis and logical reasoning		Instantiation Real example	Absolute
Measurement of the performance of the artifact with user tests	Environment / Consistency with people / Utility	Quantitative	Students Practitioners Researchers	Instantiation	Absolute
Qualitative feedback after the user tests	Environment / Consistency with people / Ease of use	Qualitative	Students Practitioners Researchers	Instantiation	Relative to absence of artifact
Qualitative feedback after the user tests	Environment / Consistency with the organization / Utility	Qualitative	Students Practitioners Researchers	Instantiation	Relative to absence of artifact

Table 4: IS artifact evaluation model instances (adapted from (Prat et al., 2014))

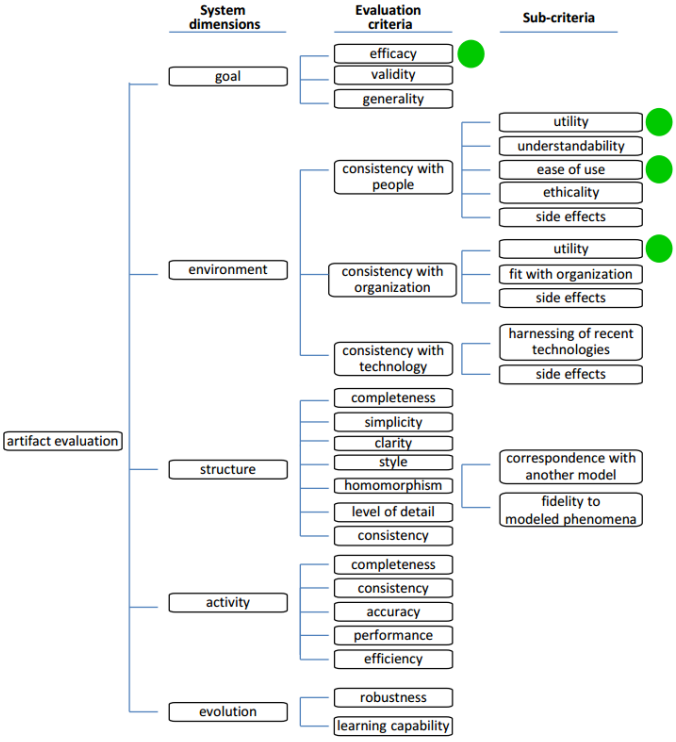


Figure 34: Criteria selected from the evaluation criteria hierarchy (adapted from (Prat et al., 2014))

7.1 Demonstration

From the demonstration the author can conclude that the artifact achieves the desired goal, providing one coherent environment to visualize the organization's EA meta-model evolution through time. The contribution to better communicate relevant meta-model information changes between stakeholders verifies the goal efficacy, as exemplified with the interface demonstration with a private organization described in the "Demonstration" section.

7.2 User Tests

To measure the quality of the artifact in practical use, the interface performance was measured with user tests.

7.2.1 Setup

The tests were executed by 20 users, including students, researchers and practitioners. The tests were performed in three separate sessions, all comprising the same environment conditions. The users were isolated in a quiet room and followed the same guidelines, starting from an empty setup in the interface. The user tests guideline is presented in Appendix A. The user tests started with a introduction to EA evolution, followed by the task's enumeration. The test comprised performing 16 tasks to test the interface capabilities. The tasks incorporated the creation of requirements and different types of

Task	Average Time	Baseline Time
1	00:00:40	00:00:27
2	00:00:47	00:00:26
3	00:02:24	00:00:53
4	00:00:32	00:00:24
5	00:01:31	00:00:46
6	00:00:37	00:00:19
7	00:01:38	00:00:53
8	00:00:28	00:00:24
9	00:01:22	00:00:50
10	00:00:29	00:00:26
11	00:01:28	00:00:47
12	00:00:53	00:00:12
13	00:00:57	00:00:13
14	00:00:44	00:00:13
15	00:00:23	00:00:08
16	00:00:08	00:00:08
Total	0:15:05	0:07:26

Table 5: User tests time results

transformations, as well as exploring the interface using the supported information visualization features.

The author evaluated the users using three metrics: time, number of clicks to perform a task, and number of mistakes made. Time acts as a quantitative measure to analyze the users efficiency accomplishing each task. The number of clicks allows to infer if the users struggled to accomplish the task, deducing if the task's goal was achieved by trial and error or not. A mistake considers any step the user made that was not expected beforehand, diverging from the expected path to complete the task or any misinterpretation. Input errors do not count as mistakes, as per example misspelled words.

7.2.2 Results and discussion

The average time to perform all tasks was 15 minutes and 5 seconds, the lowest time being 11 minutes and 37 seconds and the highest 17 minutes and 43 seconds. The results from the user tests are presented in Appendix A with further detail. In comparison to the baseline time, performed by an user with previous knowledge on the interface, the new users took double the time to perform the same tasks. Although, similar task's time reduced after the completion of similar initial tasks, showing an exponential learning curve. No user failed to perform any task. Errors and/or failures were mitigated by field validations, guiding the user steps while performing the tasks. The results with each task's average time is illustrated in Table 5

When analyzing the clicks per task, the results were satisfactory not diverging from what was expected. The results with the average number of clicks by task, rounded to the units, is presented in Table 6.

Task	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Average Clicks	31	5	15	4	13	4	15	4	15	4	16	6	5	5	3	1
Baseline Clicks	30	4	12	4	12	4	15	4	14	4	16	6	2	3	3	1

Table 6: User tests clicks results

Finally, an analysis on the most common errors during the tasks is described. The most common error identified was in the transformation window when adding a transformation. The name of the transformation was confused with the input value for the operation. Users discovered the error afterwards, when processing the remaining required transformation fields, and changed the values accordingly.

From these results we can assess the utility of the artifact, regarding the consistency with people in the environment.

7.3 Usability Questionnaire

After the tests, the users answered a small questionnaire to analyze the interface's usability. The questionnaire consisted of five questions, using a scale of 1 to 5, with 1 meaning that the user strongly disagrees and 5 meaning he/she strongly agrees with the statement.

Figure 39 in Appendix B shows the questionnaire performed to the users after the realization of the user tests, as well as a graphical representation of the results.

The users understood the interface's goal (45% agree and 50% strongly agreed), found the interface easy to use (65% agreed and 30% strongly agreed), acknowledged that the features are well integrated in the interface (65% agreed and 30% strongly agreed) and disagreed that support was required from someone specialized to use the interface (25% strongly disagreed and 45% disagreed). From the feedback provided it is possible to validate the ease of use relative to the consistency with people in the environment.

In regard to the artifact consistency with the organization, the artifact introduces a missing feature to the organization's EA Tool, which improves the organization's EAM capabilities, verifying the artifact utility. The users agreed with that the interface would provide value for organizations (50% agreed and 45% strongly agreed), corroborating the statement.

The questionnaire asked for some comments and/or suggestions regarding the interface's usability. Users pointed that the remove and add project buttons were different from the others, not being so explicit. A single user commented that the transformation fields could be in white, or with more contrast, to depict better what fields are required for a specific transformation operation.

Most of the comments from the users to the interface's usability were already thought through during the interface design process. The interface was designed as a complement to an exist EA vendor Tool (EAMS), trying to accomplish the same look and feel. The buttons disposition intends to take better advantage of the available space, taking into consideration use case scenarios with larger amounts of information, as exemplified in the demonstration in section 6.

7.4 Summary

After evaluating the proposed IS artifact, the author can conclude that the research objectives, defined in the ”Proposal” chapter, were met. The ”Demonstration” chapter supports this statement by representing a real case scenario of EA evolution, with two iterations, where EA architects are able to represent and manage the EA meta-model successfully. Also, additional tests, verifying the interface’s utility and ease of use, evaluated the artifact’s quality. These tests were complemented by an usability questionnaire, thus, also proving the proposed IS artifact achieves the desired goal, supporting EA architects tasks.

Chapter 8

Conclusion

In this chapter, a final conclusion of the research developed is presented. Initially an overall description of the the thesis research is described, followed by an explanation of how the research was communicated. Afterwards, the author analyzes the limitations found in the proposed solution. Finally, the research future work plan is delineated. This section comprehends the ”Communication” step in the DSRM process.

The author proposed an EA meta-model evolution interface to address the lack of support from EA tools in regard to EA evolution as an iterative process, that requires co-evolving not only the EA meta-model, but also the EA models. The proposed interface provides a link between the requirements motivating structural changes, the EA projects that implement those changes, and the organizational assets defining the organization's structure.

Afterwards, a demonstration was performed with a private organization under the scope of an EA project, followed by users validation ranging from EA practitioners to students. The interface was successful in providing a consistent EA evolution visualization environment to manage the organization's EA meta-model, hence supporting the organization's EAM tasks and achieving the research objective.

Besides the thesis itself and the feedback gathered through out the user tests, the research was also communicated to IJISMD, an IS journal, with the article:

Silva, N., Ferreira, F., Sousa, P., and da Silva, M. M. (2016). Automating the migration of enterprise architecture models. International Journal of Information System Modeling and Design (IJISMD)*, 7(2):72–90*

Another article presenting this thesis research, contributing to the IS community with a complete description of the proposed interface for EA meta-model evolution, was submitted to the journal Enterprise Modeling and Information Systems Architectures (EMISA) but has not been accepted yet.

The author identified some limitations of the interface during the demonstration and evaluation phases. Due to the organization's meta-model recent development, the number of classes was limited; therefore, the demonstrated evolution scenario did not impact many classes with respect to creation and removal. Furthermore, the absence of indicators reflecting the impact of meta-model changes in its instances (enterprise models) makes it difficult to assess the impact's degree of those changes. Also, not having the possibility of changing the meta-model to a previous state makes the effort of recovering previous meta-model versions higher. Finally, the application of the EA Transformations, defined using the in-

terface during an EA development project, to the EA instances in the repository has to be done with another module module in the EA tool.

For future work, despite minor improvements identified during the user validation, more features will be considered to address the above-mentioned limitations, namely impact indicators of EA transformations with respect to enterprise models during the migration phase, as well as a roll-back feature concerning both the enterprise meta-model and enterprise models, and to incorporate the ability to migrate the EA instances in the interface. Finally, more EA development project scenarios will be conducted to further validate the interface.

Bibliography

Abdallah, S. and Galal-Edeen, G. H. (2006). Towards a framework for enterprise architecture frameworks comparison and selection. In *The Fourth International Conference on Informatics and Systems (INFOS2006)*.

Ahlemann, F., Stettiner, E., Messerschmidt, M., and Legner, C. (2012). *Strategic enterprise architecture management: challenges, best practices, and future developments*. Springer Science & Business Media.

Buckl, S., Ernst, A. M., Lankes, J., Matthes, F., and Schweda, C. M. (2008). Enterprise architecture management patterns–exemplifying the approach. In *Enterprise Distributed Object Computing Conference, 2008. EDOC'08. 12th International IEEE*, pages 393–402. IEEE.

Buckl, S., Ernst, A. M., Lankes, J., Schneider, K., and Schweda, C. M. (2007a). A pattern based approach for constructing enterprise architecture management information models. *Wirtschaftinformatik Proceedings 2007*, page 65.

Buckl, S., Ernst, A. M., Lankes, J., Schweda, C. M., and Wittenburg, A. (2007b). Generating visualizations of enterprise architectures using model transformations. In *EMISA*, volume 2007, pages 33–46.

Buckl, S., Ernst, A. M., Matthes, F., and Schweda, C. M. (2009). Visual roadmaps for managed enterprise architecture evolution. In *Software Engineering, Artificial Intelligences, Networking and Parallel/Distributed Computing, 2009. SNPD'09. 10th ACIS International Conference on*, pages 352–357. IEEE.

Buschle, M., Ekstedt, M., Grunow, S., Hauder, M., Matthes, F., and Roth, S. (2012). Automating enterprise architecture documentation using an enterprise service bus.

Card, S. K., Mackinlay, J. D., and Shneiderman, B. (1999). *Readings in information visualization: using vision to think*. Morgan Kaufmann.

Chen, M., Ebert, D., Hagen, H., Laramee, R. S., Van Liere, R., Ma, K.-L., Ribarsky, W., Scheuermann, G., and Silver, D. (2009). Data, information, and knowledge in visualization. *IEEE Computer Graphics and Applications*, 29(1):12–19.

da Silva, A. R. (2015). Model-driven engineering: A survey supported by the unified conceptual model. *Computer Languages, Systems & Structures*, 43:139–155.

Denert-Stiftungslehrstuhl, E. (2008). Enterprise architecture management pattern catalog.

Fischer, R., Aier, S., and Winter, R. (2007). A federated approach to enterprise architecture model maintenance. *Enterprise Modelling and Information Systems Architectures*, 2(2):14–22.

Frank, U., Heise, D., Kattenstroth, H., Ferguson, D., Hadar, E., and Waschke, M. (2009). Itml: a domain-specific modeling language for supporting business driven it management. In *Proceedings of the 9th Workshop on Domain-Specific Modeling (DSM) at the International Conference on Object Oriented Programming, Systems, Languages and Applications (OOPSLA), Orlando, Florida, USA*.

Godinez, M., Hechler, E., Koenig, K., Lockwood, S., Oberhofer, M., and Schroeck, M. (2010). *The Art of Enterprise Information Architecture: A Systems-Based Approach for Unlocking Business Insight, Portable Documents*. Pearson Education.

Herrmannsdörfer, M. et al. (2011). *Evolutionary metamodeling*. PhD thesis, Technische Universität München.

Hilliard, R. (2000). Ieee-std-1471-2000 recommended practice for architectural description of software-intensive systems.

Jonkers, H., Quartel, D., van Gils, B., Franken, H., and Lankhorst, M. (2014). Arquitetura empresarial com togaf® 9.1 e archimate® 2.1.

Josey, A. (2011). *TOGAF® Version 9.1-A Pocket Guide*. Van Haren.

Josey, A. et al. (2013). *ArchiMate® 2.1-A Pocket Guide*. Van Haren.

Kaisler, S. H., Armour, F., and Valivullah, M. (2005). Enterprise architecting: Critical problems. In *Proceedings of the 38th Annual Hawaii International Conference on System Sciences*, pages 224b–224b. IEEE.

Lankhorst, M. (2009). {Enterprise Architecture at Work: Modelling, Communication and Analysis (The Enterprise Engineering Series)}.

Mantz, F., Taentzer, G., Lamo, Y., and Wolter, U. (2015). Co-evolving meta-models and their instance models: a formal approach based on graph transformation. *Science of Computer Programming*, 104:2–43.

Matthes, F., Buckl, S., Leitel, J., and Schweda, C. M. (2008). *Enterprise architecture management tool survey 2008*. Techn. Univ. München.

Matthes, F., Nikolaus, K., and Matheus, H. (2014). *Enterprise architecture management tool survey 2014 update*. Techn. Univ. München.

Meyer, R. (2010). Knowledge visualization. *Trends in Information Visualization*, 23.

Munzner, T. (2014). *Visualization Analysis and Design*. CRC Press.

omg (2006). *Meta Object Facility (MOF) Core Specification Version 2.0*.

Peffers, K., Tuunanen, T., Rothenberger, M. A., and Chatterjee, S. (2007). A design science research methodology for information systems research. *Journal of management information systems*, 24(3):45–77.

Poole, J., Chang, D., Tolbert, D., and Mellor, D. (2002). *Common warehouse metamodel*, volume 20. John Wiley & Sons.

Prat, N., Comyn-Wattiau, I., and Akoka, J. (2014). Artifact evaluation in information systems design-science research-a holistic view. In *PACIS*, page 23. Citeseer.

Roth, S. and Matthes, F. (2014). Visualizing differences of enterprise architecture models. In *Proc. International Workshop on Comparison and Versioning of Software Models (CVSM) at Software Engineering (SE)*.

Rumbaugh, J., Jacobson, I., and Booch, G. (2004). *Unified Modeling Language Reference Manual, The*. Pearson Higher Education.

Santos, A. (2014). Visualizing business process based costs.

Shneiderman, B. (1996). The eyes have it: A task by data type taxonomy for information visualizations. In *Visual Languages, 1996. Proceedings., IEEE Symposium on*, pages 336–343. IEEE.

Silva, N., Ferreira, F., Sousa, P., and da Silva, M. M. (2016). Automating the migration of enterprise architecture models. *International Journal of Information System Modeling and Design (IJISMD)*, 7(2):72–90.

Sousa, P., Lima, J., Sampaio, A., and Pereira, C. (2009). An approach for creating and managing enterprise blueprints: A case for it blueprints. In *Advances in enterprise engineering III*, pages 70–84. Springer.

Tang, A., Han, J., and Chen, P. (2004). A comparative analysis of architecture frameworks. In *Software Engineering Conference, 2004. 11th Asia-Pacific*, pages 640–647. IEEE.

Tribolet, J., Sousa, P. M. A., and Caetano, A. (2014). The role of enterprise governance and cartography in enterprise engineering. *Enterprise Modelling and Information Systems Architectures*, 9(1):38–49.

Truyen, F. (2006). The fast guide to model driven architecture the basics of model driven architecture. *Cephas Consulting Corp*.

Urbaczewski, L. and Mrdalj, S. (2006). A comparison of enterprise architecture frameworks. *Issues in Information Systems*, 7(2):18–23.

Van Wijk, J. J. (2005). The value of visualization. In *VIS 05. IEEE Visualization, 2005.*, pages 79–86. IEEE.

Von Alan, R. H., March, S. T., Park, J., and Ram, S. (2004). Design science in information systems research. *MIS quarterly*, 28(1):75–105.

Wachsmuth, G. (2007). Metamodel adaptation and model co-adaptation. In *European Conference on Object-Oriented Programming*, pages 600–624. Springer.

Ward, M. O., Grinstein, G., and Keim, D. (2010). *Interactive data visualization: foundations, techniques, and applications*. CRC Press.

Winter, K., Buckl, S., Matthes, F., and Schweda, C. M. (2010). Investigating the state-of-the-art in enterprise architecture management methods in literature and practice. *MCIS*, 90.

Winter, R. and Fischer, R. (2006). Essential layers, artifacts, and dependencies of enterprise architecture. In *2006 10th IEEE International Enterprise Distributed Object Computing Conference Workshops (EDOCW'06)*, pages 30–30. IEEE.

Zachman, J. A. (1987). A framework for information systems architecture. *IBM systems journal*, 26(3):276–292.

Appendix A

User Tests

Visualização da Evolução de Meta-modelos em Arquiteturas Empresariais

Testes com Utilizadores

Introdução

A arquitetura empresarial permite analisar de um forma holistica e coerente, uma organização utilizando modelos e/ou representações para a descrever. Esta descrição incorpora tanto a parte tecnológica como a parte de negócio de uma organização, utilizando frameworks ou notações de arquitetura empresarial como guia para desenhar os modelos, definindo o meta-modelo da organização. Embora estas iniciativas sejam completas, as organizações inicialmente têm dificuldades em incorporar todos os seus conceitos na sua estrutura, sendo um processo iterativo. Para além disso, a arquitetura empresarial de uma empresa está sempre em constante evolução e é necessário planear e gerir as mudanças necessárias. A interface desenvolvida permite resolver alguns destes problemas, permitindo aplicar transformações a um meta-modelo e visualizar as suas consequências. Estas transformações passam por criar, eliminar ou atualizar elementos, classes ou propriedades do meta-modelo.

Conceitos Chave:
Meta-modelo – Representação que define modelos/representações de uma organização.
Transformação – Mudança efetuada ao meta-modelo.
Requisito – Motivação que levou a necessidade de uma transformação.

Tarefas

1. Criar um novo projeto com o nome “Demo”, entre o período de 01/01/2016 e 01/01/ 2018.
2. Adicionar um requisito “New class type required”.
3. Adicionar uma transformação, criando uma nova classe com o nome “Application Behaviour com a classificação “Application Architecture””, associando ao requisito criado no passo anterior.
4. Adicionar um novo requisito “Class no longer required”.
5. Adicionar uma nova transformação, removendo a classe “Application Component”, associando ao requisito definido no passo anterior.
6. Adicionar um novo requisito “New property required”.
7. Adicionar uma transformação, criando uma nova propriedade “Description” para a classe “Application Behaviour”, do tipo texto, associando ao requisito definido no passo anterior.
8. Adicionar um novo requisito “Property no longer used”.
9. Adicionar uma transformação, removendo a propriedade “Aggregates” da classe “Business Process”, associando ao requisito definido no passo anterior.
10. Adicionar um novo requisito “Property update required”.
11. Adicionar uma transformação, atualizando o tipo da propriedade “External Link” da classe “Business Process” de URL para texto, associando ao requisito do passo 10.
12. Visualizar quais as transformações e operações associadas à classe "Business Process".
13. Visualizar que alterações foram efetuadas as propriedas da classe “Business Process”.
14. Remover a transformação que elimina a classe "Application Component".
15. Remover o requisito “Class no longer required”.
16. Guardar as alterações.

Nota: *Durante a realização das tarefas expõe os teus pensamentos ou comentários em voz alta.*

Figure 35: User tests guidelines

User	Tarefa 1	Cliques 1	Tarefa 2	Cliques 2	Tarefa 3	Cliques 3	Tarefa 4	Cliques 4	Tarefa 5	Cliques 5	Tarefa 6	Cliques 6	Tarefa 7
Baseline	00:00:27	30	00:00:26	4	00:00:53	12	00:00:24	4	00:00:46	12	00:00:19	4	00:00:53
1	00:00:34	30	00:00:29	4	00:02:14	17	00:00:19	4	00:01:23	12	00:00:23	4	00:01:57
2	00:00:43	30	00:00:22	4	00:03:40	22	00:00:15	4	00:00:43	13	00:00:27	6	00:02:38
3	00:00:33	30	00:00:41	4	00:03:30	16	00:00:25	4	00:01:11	12	00:00:50	4	00:01:34
4	00:00:45	30	00:00:31	4	00:02:30	15	00:00:35	4	00:02:46	12	00:00:19	4	00:00:53
5	00:00:33	30	00:00:54	4	00:01:40	15	00:00:30	4	00:01:59	14	00:00:53	4	00:01:36
6	00:00:55	32	00:00:58	6	00:01:40	15	00:00:29	4	00:00:47	12	00:00:41	4	00:01:04
7	00:00:44	30	00:00:53	4	00:01:27	15	00:00:29	6	00:00:48	12	00:00:27	4	00:01:50
8	00:00:20	30	00:00:51	4	00:03:18	20	00:00:22	4	00:01:26	18	00:01:17	4	00:01:04
9	00:00:43	30	00:00:43	4	00:03:09	14	00:00:40	4	00:00:01	12	00:00:33	4	00:01:24
10	00:01:00	32	00:01:23	6	00:02:13	14	00:00:40	4	00:02:54	15	00:00:40	4	00:02:45
11	00:00:47	35	00:00:50	6	00:02:14	12	00:00:32	4	00:02:34	14	00:00:41	4	00:02:26
12	00:00:57	30	00:00:44	4	00:03:00	15	00:00:43	5	00:01:49	12	00:00:33	4	00:01:33
13	00:00:40	30	00:00:37	6	00:02:30	14	00:00:33	4	00:02:09	14	00:00:40	4	00:00:31
14	00:00:40	29	00:00:48	4	00:02:40	15	00:00:32	4	00:01:39	13	00:01:08	4	00:02:07
15	00:00:34	30	00:00:42	4	00:02:30	14	00:00:29	5	00:01:22	12	00:00:20	4	00:01:12
16	00:00:44	30	00:00:45	4	00:02:08	14	00:00	4	00:00:47	12	00:00:18	4	00:00:50
17	00:00:32	33	00:01:32	6	00:01:53	14	00:00:30	4	00:01:21	14	00:00:18	4	00:02:33
18	00:00:37	30	00:0050	5	00:01:28	13	00:00:22	4	00:00:55	12	00:00:25	4	00:01:37
19	00:00:40	30	00:00:55	4	00:01:51	16	00:00:32	4	00:01:10	12	00:00:38	4	00:01:08
20	00:00:43	30	00:00:40	4	00:01:39	14	00:00:38	4	00:01:55	14	00:00:33	4	00:01:10
**Media de (	**00:00:43**	**30,55**	**00:00:47**	**4,55**	**00:02:24**	**15,2**	**00:00:32**	**4,2**	**00:01:31**	**13,05**	**00:00:37**	**4,1**	**00:01:38**

Figure 36: User tests results (part 1)

Cliques 7	Tarefa 8	Cliques 8	Tarefa 9	Cliques 9	Tarefa 10	Cliques 10	Tarefa 11	Cliques 11	Tarefa 12	Cliques 12	Tarefa 13	Cliques 13	Tarefa 14
15	00:00:24	4	00:00:50	14	00:00:26	4	00:00:47	16	00:00:12	6	00:00:13	2	00:00:13
15	00:00:17	4	00:01:00	14	00:00:22	4	00:01:22	16	00:00:56	3	00:00:58	3	00:00:42
15	00:00	4	00:01:01	14	00:00:19	4	00:02:08	16	00:00:25	2	00:00:32	2	00:00:34
15	00:00:15	4	00:01:00	14	00:00:23	4	00:00:51	16	00:00:28	5	00:00:46	4	00:00:20
15	00:00:24	4	00:02:44	14	00:00:12	4	00:01:01	19	00:01:10	7	00:01:21	2	00:00:26
15	00:00:27	4	00:01:50	14	00:00:31	4	00:01:07	16	00:00:44	9	00:01:08	5	00:00:24
15	00:00:40	4	00:01:20	14	00:00:29	4	00:01:09	16	00:00:19	6	00:00:52	8	00:00:26
15	00:00:18	4	00:01:08	14	00:00:19	4	00:00:57	16	00:00:42	5	00:01:04	6	00:01:08
15	00:00:14	4	00:01:27	14	00:00:17	4	00:01:14	16	00:00:49	6	00:02:04	10	00:00:42
15	00:00:30	4	00:01:08	14	00:00:32	4	00:00:55	16	00:02:04	11	00:02:04	13	00:01:29
17	00:00:17	4	00:01:46	19	00:00:40	4	00:01:44	16	00:01:24	5	00:00:37	4	00:00:40
15	00:00:42	4	00:01:27	14	00:00:37	4	00:01:24	16	00:00:50	8	00:00:20	3	00:00:53
16	00:00:19	4	00:01:26	18	00:00:30	4	00:02:35	15	00:00:20	4	00:01:30	12	00:01:20
15	00:00:40	4	00:01:22	14	00:00:40	4	00:01:47	16	00:01:04	8	00:00:20	3	00:00:46
15	00:00:43	4	00:01:23	14	00:00:45	4	00:01:49	16	00:00:40	8	00:00:23	4	00:00:41
15	00:00:32	4	00:01:06	16	00:00:29	4	00:01:26	18	00:00:47	7	00:00:41	5	00:00:45
15	00:00:20	4	00:00:56	14	00:00:30	4	00:01:06	16	00:01:10	5	00:00:29	3	00:00:34
15	00:00:34	4	00:01:14	14	00:00:25	4	00:01:38	18	00:00:41	7	00:00:34	4	00:00:32
17	00:00:22	4	00:00:36	14	00:00:19	4	00:01:16	16	00:01:00	8	00:01:24	8	00:01:07
15	00:00:30	4	00:01:13	16	00:00:29	4	00:01:22	16	00:00	6	00:00:40	4	00:00:38
15	00:00:32	4	00:01:24	16	00:00:30	4	00:01:41	16	00:00:50	8	00:01:02	5	00:00:29
15,25	**00:00:28**	**4**	**00:01:22**	**14,75**	**00:00:29**	**4**	**00:01:28**	**16,3**	**00:00:53**	**6,4**	**00:00:57**	**5,4**	**00:00:44**

Figure 37: User tests results (part 2)

Cliques 14	Tarefa 15	Cliques 15	Tarefa 16	Cliques 16	Tempo Total
3	00:00:08	3	00:00:05	1	**0:07:26**
5	00:00:21	5	00:00:06	1	**0:13:23**
5	00:00:21	4	00:00:04	1	**0:14:30**
3	00:00:09	3	00:00:04	1	**0:13:00**
6	00:00:15	3	00:00:08	1	**0:16:00**
3	00:00:13	3	00:00:08	1	**0:14:37**
3	00:00:16	3	00:00:04	1	**0:12:09**
6	00:00:13	3	00:00:06	1	**0:12:33**
8	00:00:21	3	00:00:10	2	**0:15:56**
5	00:01:05	3	00:00:11	1	**0:17:11**
3	00:00:11	3	00:00:06	1	**0:19:00**
7	00:00:38	6	00:00:12	1	**0:17:07**
3	00:00:16	3	00:00:08	1	**0:17:43**
3	00:00:17	3	00:00:07	1	**0:14:43**
3	00:00:12	3	00:00:09	1	**0:16:19**
10	00:00:21	3	00:00:07	2	**0:13:23**
5	00:00:20	3	00:00:06	1	**0:11:37**
3	00:00:13	3	00:00:03	1	**0:14:33**
5	00:01:07	5	00:00:11	1	**0:12:46**
3	00:00:27	3	00:00:07	1	**0:13:19**
4	00:00:21	3	00:00:09	1	**0:14:16**
4,65	**00:00:23**	**3,4**	**00:00:08**	**1,1**	**0:15:05**

Figure 38: User tests results (part 3)

Appendix B

Usability Questionnaire

Visualização da Evolução de Meta-modelos em Arquiteturas Empresariais

*Obrigatório

Usabilidade do Sistema

Percebi o objetivo da interface. *

	1	2	3	4	5	
Discordo Totalmente	○	○	○	○	○	Concordo Totalmente

A interface foi fácil de utilizar. *

	1	2	3	4	5	
Discordo Totalmente	○	○	○	○	○	Concordo Totalmente

As funcionalidades estão bem integradas na interface. *

	1	2	3	4	5	
Discordo Totalmente	○	○	○	○	○	Concordo Totalmente

A interface será útil para as organizações. *

	1	2	3	4	5	
Discordo Totalmente	○	○	○	○	○	Concordo Totalmente

Preciso de suporte de alguém especializado para usar a interface. *

	1	2	3	4	5	
Discordo Totalmente	○	○	○	○	○	Concordo Totalmente

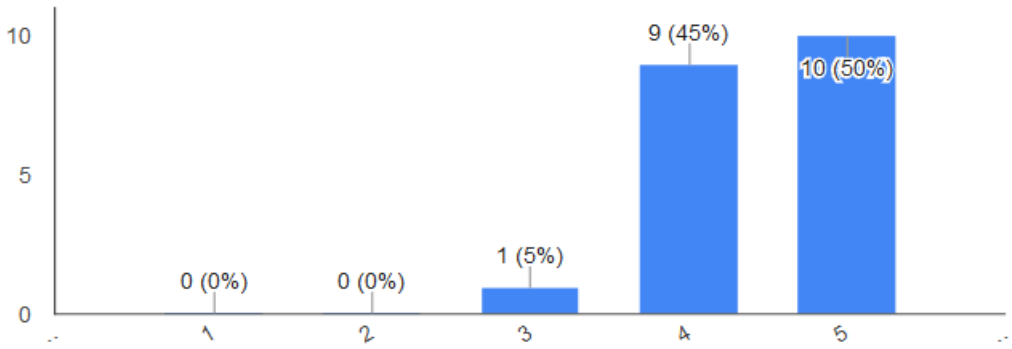
Comentários e/ou Sugestões

A sua resposta

Figure 39: Usability questionnaire

Usabilidade do Sistema

Percebi o objetivo da interface. (20 respostas)



A interface foi fácil de utilizar. (20 respostas)

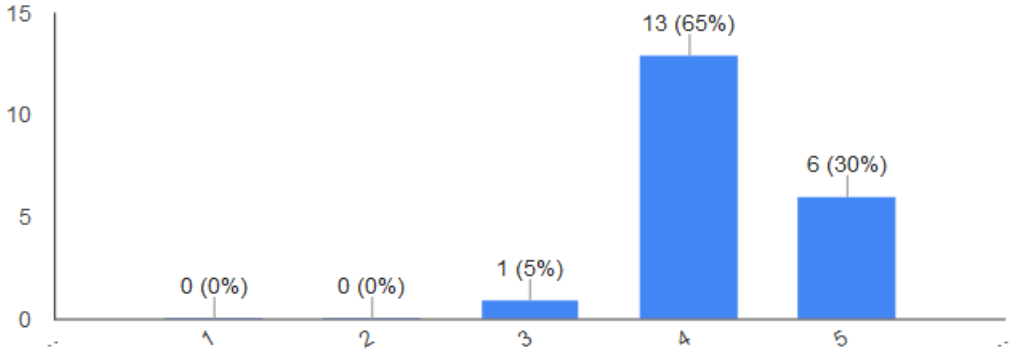
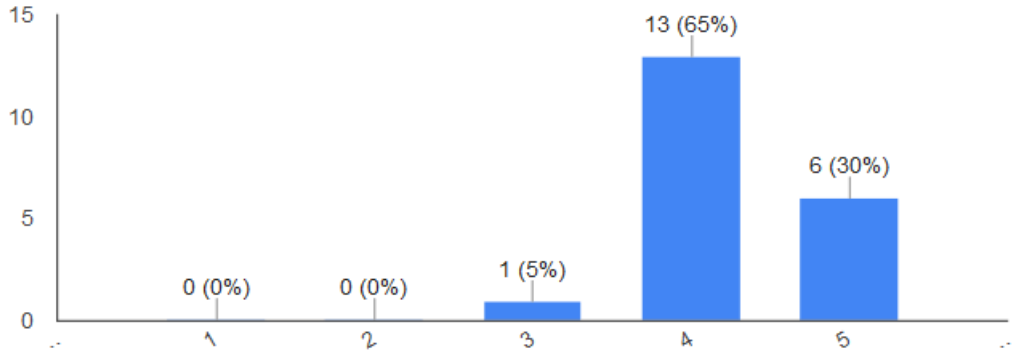


Figure 40: Usability questionnaire results (part 1)

As funcionalidades estão bem integradas na interface. (20 respostas)



A interface será útil para as organizações. (20 respostas)

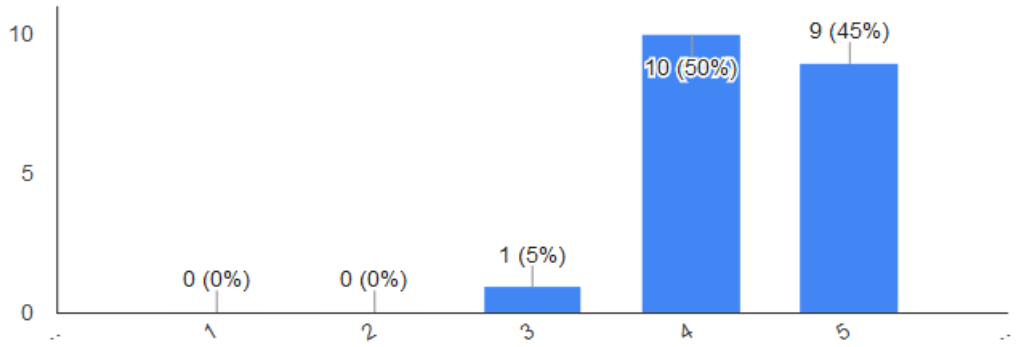
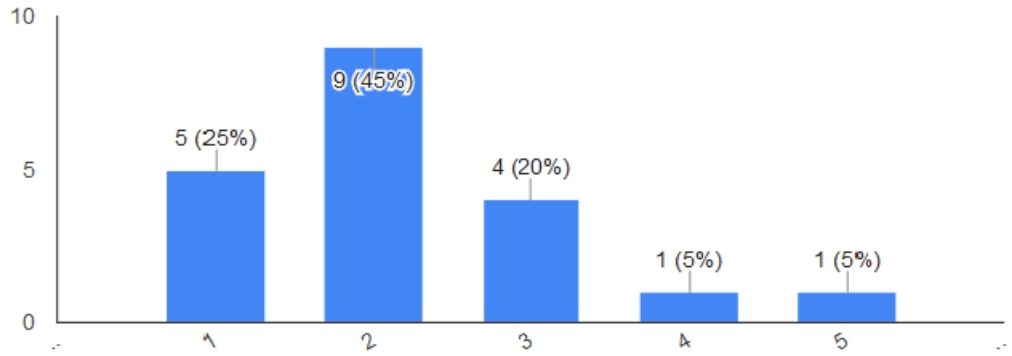


Figure 41: Usability questionnaire results (part 2)

Preciso de suporte de alguém especializado para usar a interface. (20 respostas)



Comentários e/ou Sugestões (5 respostas)

O create project não está tão explicito como as restantes operações.
Ter um botão de zoom.

Quando existe selecção de transformações ou propriedades, deveria de saltar para a vista mais apropriada a percepção das mesmas

Eram mais intuitivo os botões para adicionar/remover transformações/requisitos estarem ao pé das listras de transformações/requisitos.

Para apagar os requisitos pôr uma cruz mesmo em cada requisito.
Mudar o nome do campo "Value"

No formulário de criação de transformações eu sugeria ordenar os campos para relacionar melhor com a linguagem falada. Por exemplo "Pegar na [Class] Application Architecture, adicionar uma [Propriedade] do [Tipo] e com o [Valor] tal."

Figure 42: Usability questionnaire results (part 3)