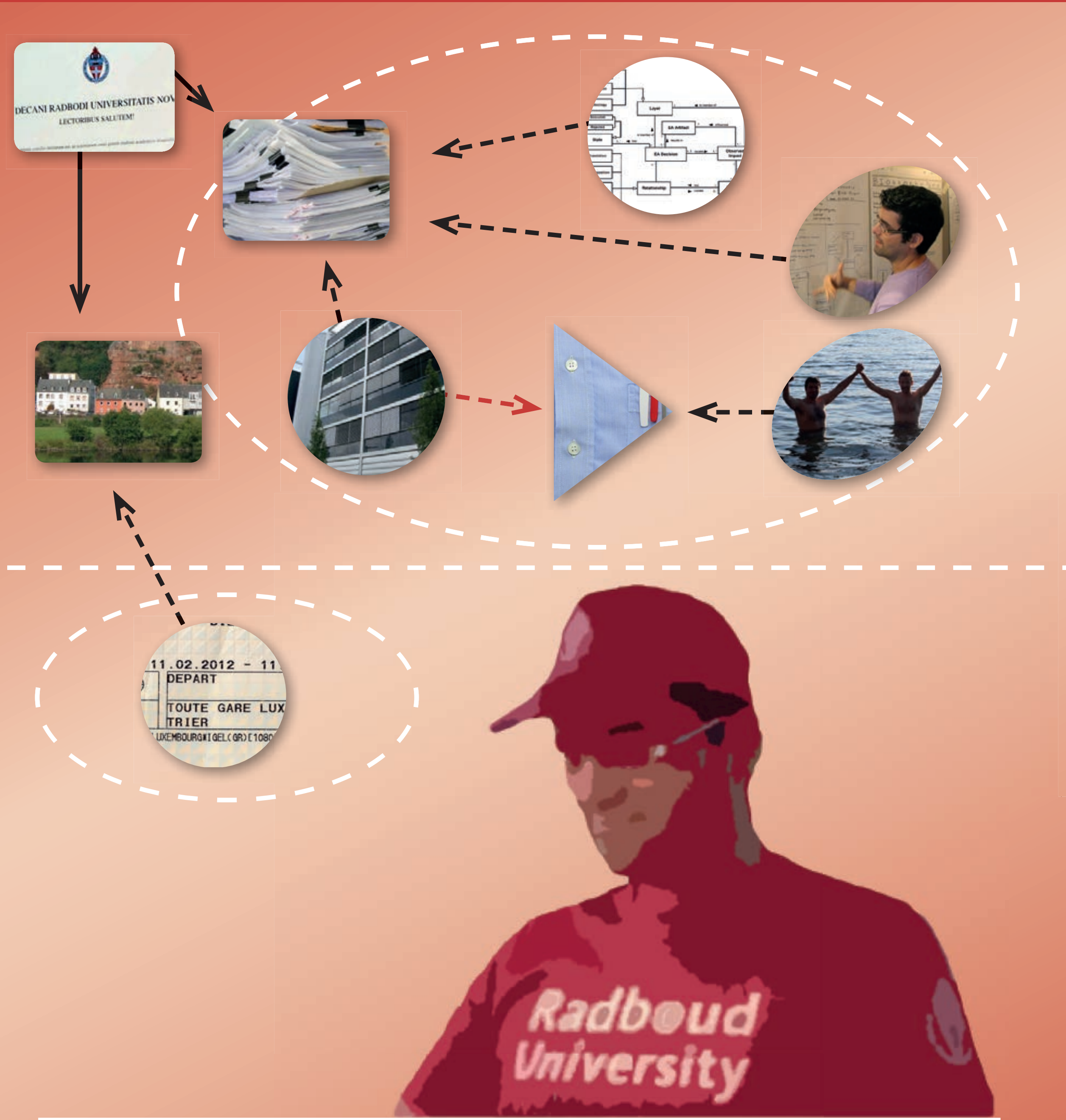
# *EA Anamnesis* - A Conceptual Framework for Enterprise Architecture Rationalization



**Georgios Plataniotis**

# EA Anamnesis – A Conceptual Framework for Enterprise Architecture Rationalization

by

Georgios Plataniotis

Printed and published by Ridderprint BV, Ridderkerk, the Netherlands
Layout by Georgios Plataniotis
Cover design by Georgios Plataniotis and Maria Fragkouli
Back cover drawing by Angelos-Theologos Plataniotis and Zoi Plataniotí
ISBN 978-94-6299-566-6

EE-Network dissertation series: 2017-1

The research reported in this thesis was conducted at Luxembourg Institute of Science and Technology, under the auspices of the Institute for Computing and Information Sciences of Radboud University Nijmegen and the Enterprise Engineering Network. The research was partially supported by the Fonds National de la Recherche Luxembourg.

# EA Anamnesis – A Conceptual Framework for Enterprise Architecture Rationalization

Proefschrift

ter verkrijging van de graad van doctor
aan de Radboud Universiteit Nijmegen
op gezag van de rector magnificus prof. dr. J.H.J.M van Krieken,
volgens besluit van het college van decanen
in het openbaar te verdedigen op dinsdag 4 april 2017
om 10.30 uur precies

door

Georgios Plataniotis

geboren op 24 april 1980
te Ilion, Griekenland

**Promotor:**
Prof. dr. H.A. Proper

**Copromotoren:**

| | |
|---|---|
| Dr. S. de Kinderen | University of Duisburg-Essen, Duitsland |
| Dr. Q. Ma | University of Luxembourg, Luxemburg |

**Manuscriptcommissie:**

| | |
|---|---|
| Prof. dr. Th.P. van der Weide | |
| Prof. dr. R. Winter | University of St. Gallen, Zwitserland |
| Prof. dr. D. Vergados | University of Piraeus, Griekenland |

**Paranimfen:**
M. Fragkouli, MSc.
Dr. D. van der Linden

*What is Success?*
*To laugh often and much;*
*To win the respect of intelligent people*
*and the affection of children;*
*To earn the appreciation of honest critics*
*and endure the betrayal of false friends;*
*To appreciate beauty;*
*To find the best in others;*
*To leave the world a bit better, whether by*
*a healthy child, a garden patch*
*or a redeemed social condition;*
*To know even one life has breathed*
*easier because you have lived;*
*This is to have succeeded.*

Ralph Waldo Emerson

# Contents

# Acknowledgments

The book that you are currently holding in your hands is the outcome of a long, strenuous, full of challenges and emotional changes journey that lasted for whole five years. It all started a snowy morning in St. Gallen, Switzerland, at the beginning of 2012. I was there to meet my future colleagues in CRP Henri Tudor in Luxembourg and my partners from the University of St Gallen. I woke up really excited and curious to meet new people and understand what our research project was about.

One of the key people that I met there, who supported and helped me throughout this journey, was my supervisor Erik Proper. Erik, I will begin by thanking you for the opportunity you gave me when you selected me to join your research team. I want you to know that I've been trying until this very moment of this project to prove that you made the right decision when you selected me as a member of the team. I also thank you for providing me with the necessary academic freedom, trust and guidance. This helped me a lot during the early stages of my research, in order to finally come up with an interesting topic to work on for the next few years and in the long term to grow as an independent researcher. Finally, thank you Erik for our long bike rides across the Mosel, and for the nice times we had in Monte Petris and other places as well.

Another person that played a tremendous role during the execution of this study was my first co-supervisor Sybren de Kinderen. Sybren, this work would not have even started without your guidance and support. Thank you for your exceptional dedication and continuous interest on my research work. I sincerely wish other junior researchers would have the same type of supervision that I had during this project. Moreover, thank you Sybren for being a good friend and for all the fun we had during cycling, playing pool, water-skiing and many other activities. It really helped me to overcome some of my PhD

related stress.

At this part, I also feel the need to thank my second co-supervisor Qin Ma. Qin, you started being involved in the co-supervision of this research work at a later stage, however that was not at all a problem for you. I am still amazed on how quickly you caught up with the project and on the contribution and impact you had on my work. I really admire your scientific qualities and your detailed oriented way of doing scientific research. It helped me a lot to improve my research work. Thank you for your effort and for your sincere interest on my research work.

Scientific research, especially at a PhD level, is not an easy thing at all. Junior researchers have to establish themselves as independent researchers and deal with various tasks such as publishing, presenting frequently their work, etcetera. Negative feelings such as frustration and stress are quite often. However, throughout these years, I found a way to keep myself strong, optimistic and persistent on my goal. Keeping only the positive things, having fun and laugh as much as possible was one of the best things I decided to adopt as an attitude towards difficulties in life. At this stage, I feel the need to thank my paranymph, officemate in Luxembourg, and good friend Dirk van der Linden. Dude, thank you for joining me in this 'hahaha' way of approaching things. I will forever remember the funny and sometime awkward moments in K2 office. I would also like to thank you for the nice academic discussions we had, but also for making me improve my swimming technique, and for the nice espressos we were enjoying together with Sybren in front of the JFK building reception. Additionally, I would like to thank my colleagues in the office in Luxembourg. Each colleague with their own unique character and culture contributed on the creation of a really interesting, fascinating environment.

The last part of this work, second case study and writing, was conducted when I returned to the e-government center for social security in Athens, where I am currently professionally employed. As such, I would like to thank the colleagues in the Athens office for their support. Especially, I would like to thank Charoula Laina and Georgios Tzortzis, who supported my initial decision to move abroad for this research project and helped me finish my thesis, when I returned back from Luxembourg. I would like also to thank my officemates Stavros Matsoukas and Elias Georgiou for the nice and friendly atmosphere in the office and for all those 'undisciplined' coffee breaks, which greatly contributed the most to keeping myself down to earth and enjoy the small pleasures of life.

The last part of this section is devoted to my parents and family. I would like to start by thanking my parents Despoina and Theologos. Thank you for raising me with good values and supporting me in every stage of my life. Unfortunately, my mother passed away a few months after I had moved abroad and started working on my PhD. Mom, not a single day passed without me thinking of you. I miss you until this very moment and this thesis is devoted to your Anamnesis.

I would like also to thank my family, my two kids Angelos-Theologos and Zoi, who came to my life while I was doing my PhD, only to make my life a little more difficult, but made me realize the real meaning of life as well. I hope that daddy made you proud with this piece of work, and will set a good example to follow in your life. Finally, I would like to conclude this section by thanking my beautiful wife Maria. Maria thank you for your love, devotion and patience throughout this study period. I know how hard was for you when I had to spend time away from you and our kids because of the research work. Just thank you for existing in my life.

# List of Publications

Georgios Plataniotis, Sybren de Kinderen and Henderik A. Proper: EA Anamnesis: Towards an Approach for Enterprise Architecture Rationalization. In: *Proceedings of the 2012 Workshop on Domain-specific Modeling*, pp. 27-32, ACM, 2012

Georgios Plataniotis, Sybren de Kinderen and Henderik A. Proper: Capturing Decision Making Strategies in Enterprise Architecture - A Viewpoint. In *Proceedings of EMMSAD 2013 (Exploring Modelling Methods for Systems Analysis and Design)*, pp. 339-353, LNBIP, 2013

Danny Greefhorst, Henderik A. Proper and Georgios Plataniotis: The dutch state of the practice of architecture principles. In: *Journal of Enterprise Architecture, (JEA)*, vol. 4, pp. 20-25, 2013

Georgios Plataniotis, Sybren de Kinderen and Henderik A. Proper: Relating Decisions in Enterprise Architecture Using Decision Design Graphs. In *Proceedings of EDOC 2013 (Enterprise Distributed Object Computing)*, pp. 139-146, IEEE, 2013

Georgios Plataniotis, Sybren de Kinderen, Dirk van der Linden, Danny Greefhorst and Henderik A. Proper: An Empirical Evaluation of Design Decision Concepts in Enterprise Architecture. In: *Proceedings of 6th IFIP WG 8.1 working conference on the Practice of Enterprise Modelling (PoEM)*, Springer, 2013

Georgios Plataniotis, Sybren de Kinderen and Henderik A. Proper: Challenges of Capturing Design Rationales in Enterprise Architecture: A case study. In *Proceedings of the 8th Workshop on Transformation and Engineering of Enterprises (TEE)*, vol. 1182, 2014

Georgios Plataniotis, Sybren de Kinderen and Henderik A. Proper: A computational approach for design rationalization in Enterprise Architecture. In: *Proceedings of 8th International Conference on Research Challenges in Information Science (RCIS)*, pp. 1-2, IEEE, 2014

Georgios Plataniotis, Sybren de Kinderen and Henderik A. Proper: EA Anamnesis: An Approach for Decision Making Analysis in Enterprise Architecture. In: *International Journal of Information Systems Modeling and Design (IJISMD)*, vol. 5, no. 3, pp. 75-95, IGI Global, 2014

Marc van Zee, Georgios Plataniotis and Dirk van der Linden and Diana Marosin: Formalizing enterprise architecture decision models using integrity constraints. In: *Proceedings of 16th Conference on Business Informatics (CBI)*, vol. 1, pp. 143-150, IEEE, 2014

Georgios Plataniotis, Sybren de Kinderen and Henderik A. Proper: Implementing a Software Prototype for Enterprise Architecture Rationalization: Lessons Learned. In *Proceedings of the 18th International Enterprise Distributed Object Computing Conference Workshops and Demonstrations (EDOCW)*, pp. 41-46, IEEE, 2014

Georgios Plataniotis, Sybren de Kinderen and Henderik A. Proper: Capturing Design Rationales in Enterprise Architecture: A Case Study. In *Proceedings of the 7th IFIP WG 8.1 Working Conference on the Practice of Enterprise Modeling (PoEM)*, vol. 197, pp. 133-147, Springer, 2014

Georgios Plataniotis, Sybren de Kinderen, Qin Ma and Henderik A. Proper: Traceability and modeling of requirements in enterprise architecture from a design rationale perspective. In: *Proceedings of 9th International Conference on Research Challenges in Information Science (RCIS)*, pp. 518-519, IEEE, 2015

Georgios Plataniotis, Sybren de Kinderen, Qin Ma and Henderik A. Proper: A Conceptual Model for Compliance Checking Support of Enterprise Architecture Decisions. In: *Proceedings of 17th Conference on Business Informatics (CBI)*, vol. 1, no. 3, pp. 191-198, IEEE, 2015

# List of Figures

# List of Tables

# Part I

# Foundations

# CHAPTER 1

# Introduction

## 1.1 Motivation

Modern day enterprises have to cope with different challenges such as new business models and the incorporation of new technologies. These challenges require organizations to be flexible and adaptable to this constantly changing environment. To ensure that enterprises have the required transformation capabilities, senior management has to make informed decisions on the design of the core organizational structure as well as the supporting IT (Lankhorst 2013). Furthermore, modern enterprises have to conform to different types of requirements. For example, legal requirements impose transparency on their operations (Ghanavati et al. 2009).

These needs can be addressed by the domain of Enterprise Architecture (EA). EA is considered as an instrument for the steering of enterprise transformations (Op 't Land et al. 2008, Hoogervorst 2004) and provides a holistic overview of the enterprise (Lankhorst 2013). EA consists of roadmaps that guide the actions of enterprise architects during an enterprise transformation. It also consists of models that help architects and relevant stakeholders to realize the cross-domain dependencies between business and IT, e.g., how a software application supports a business process.

An important step, before the initiation of the actual enterprise transformation, is the analysis of the current (as-is) state of an enterprise. EA models provide this information by representing the EA design in terms of its EA elements and their relationships. Enterprise architects inspect these models in order to understand *what* has already been done in the architecture, something that will subsequently help them with the design of the future (to-be) state of the enterprise. However, an important aspect which deals with the provision of the justification, namely the *why* behind the design, is not captured by EA models. Design issues, alternatives and decisions behind the resulting models, are often left implicit. Although we should be careful with the analogy, experience from the field of software architecture shows that leaving design rationales implicit leads to 'Architectural Knowledge vaporization' (cf. Jansen and Bosch (2005)).

Among others, such a lack of design rationale can cause design integrity issues when architects want to maintain or change the current design (Tang et al. 2007). This means that due to a lacking insight into the rationale, new designs are constructed in an ad-hoc manner, without taking into consideration constraints implied by past design decisions.

Moreover, according to a survey on software architecture design rationale (Tang et al. 2006), a large majority of architects (85,1%) admitted the importance of design rationalization in order to justify designs. Another interesting finding of this survey is that architects themselves declared that they frequently forget the reasons for their own decisions after some time.

Despite the fact that these findings originate from other domains, they do provide an indication of the more general consequences of the lack of design rationale. Indeed, anecdotal evidence out of six exploratory interviews that we conducted with senior enterprise architects, prior to the start of this PhD work, already suggested this. For example, enterprise architects often work as external consultants. This also means that clients employ different enterprise architects over time. Successor enterprise architects are then required to try to understand and analyze the architecture by searching through EA models and unstructured documentation without having a detailed insight regarding the justification behind the design. This situation worsens the architecture knowledge vaporization problem. These indications of ours, were confirmed during our interviews with the involved stakeholders of our case studies (Chapters 6, 7). Stakeholders, admitted that it would be extremely useful to have this kind of support during the execution of the enterprise transformation. Such a mechanism would inform them about the negative implications of specific architectural choices in the enterprise and would steer the decision making towards a different direction. Moreover, as we will see in Chapter 7, design rationale support raised also the awareness of problematic situations across the enterprise. Finally, as we will see in Section 2.3, the need for design rationale support for EA has also been confirmed by the means of a survey study we conducted among EA practitioners.

For all reasons stated above, we argue that EA models should be complemented with design rationale information. Design rationale is concerned with making the underlying decision making and rationale of designs (Lee and Lai 1991) explicit. Design rationale provides the underlying justification knowledge behind designs and it can be captured and/or used during the design process.

Analogous to medicine, there is a parallel between capturing and maintaining design rationales and keeping the medical history of a patient. Regardless of the critical ability of the doctor, the medical history can provide valuable information which facilitates the diagnosis and consequently the treatment of the patient. Medical history is as valuable as diagnostic tests and examinations. In an EA context, architects can use rationalization information during the analysis of existing designs/architectures to have more insights about the existing (as-is) architecture design. By doing so, they are able to make a better assessment of the as-is situation and in turn design better future enterprise transformations (Lankhorst 2013).

This thesis focuses on the problem of lack of design rationale in an EA context and proposes a conceptual framework which can be used as a basis for the creation of repositories of architectural knowledge in organizations. As we will see in Section 2.3, despite of the usefulness of capturing design rationale, the current practice of doing so is not yet prevalent. EA architects have not yet developed the discipline to capture rationale and even when they do so, they capture rationale in an unorganized way (free text format). The current status of capturing rationale does not allow them to efficiently search for rationale information or reuse some of the rationale for similar cases. Our main goal is to introduce such an organized framework for the capturing of design rationales and subsequently establish the discipline of capturing design rationales in an EA context.

## 1.2 Research questions

The goal of this thesis is the development of a conceptual framework (called EA Anamnesis, also in reference to the aforementioned analogy to a patient's medical history) that makes the design rationale of EA designs explicit, by complementing existing EA modeling languages. During the development of the framework, we identified that parts of our conceptual framework were addressing as well generic design rationale issues. Therefore, the framework will be reported on at two levels: (1) the level of a generic conceptual framework, and (2) the level of an EA specific conceptual framework that builds on the generic one. As such, the following research questions refer to both the generic and EA specific framework for design rationalization. In the discussion below, we will also highlight how the questions relate to these two levels.

- *RQ1: Which design rationale concepts can be used for the rationalization of EA designs?*

  By answering this research question we aim to identify a set of key design rationale concepts for the domain of EA. For our concepts identification we take into account existing design rationale approaches from various domains such as civil engineering, software architecture etcetera. We examine which concepts of these frameworks can be used for the domain of EA. Our main goal is to define a set of concepts which can be used as a basis for the development of our conceptual framework.

  Some of the identified concepts will be addressing design rationale issues in general. Subsequently, part of the answer of this research question will be covered by the general design rationale framework.

To span the actual framework in terms of relations, this generic research question is refined to two more specific research questions. Together they then cover the conceptual framework.

- *RQ2: How to make the underlying reasoning behind design decisions explicit?*

The decision making environment in EA is challenging due to the fact that architects have to take into account and balance among requirements of various stakeholders of the enterprise. We argue that capturing and representing the underlying reasoning behind design decisions can assist architects to inspect the as-is architecture, to analyze the evaluation process for specific decisions and to recognize which factors actually influenced their decision making process. By doing so, they can improve their future evaluations by following good practices or by avoiding bad evaluations of past decision making processes.

Making explicit the underlying reasoning of design decisions can be beneficial for other domains of high complexity where prioritization of requirements is required. As such this question concerns both the generic and the EA conceptual framework.

- *RQ3: How to capture and represent the design problem and its role in the decision making process?*

  By answering this question we provide an additional dimension to design rationalization which deals with the formulation of the EA design problem based on the given goals, principles and requirements. By doing so, we are able to analyze how the design problem is refined from high level goals to concrete requirements. Moreover, since requirements play a role in the decision making process as well, we are able to provide traceability between the problem and solution space and to check how the design decisions of the solution comply with the given requirements.

  Part of the answer of this research question addresses the generic design rationale domain and as such it is provided through our generic conceptual framework. The rest of the answer which deals with the EA domain specificities is provided by our EA conceptual framework.

Next to the above, we need to incorporate EA specific considerations in terms of the typical EA domains. This leads to the following EA specific research question:

- *RQ4: How to make cross-domain relationships of design rationale explicit?*

  EA involves multiple domains (e.g., Business, Applications and IT). During an enterprise transformation, the decision making in a specific architecture domain may trigger the execution of new decisions in the same or in a different domain of the enterprise. For example, a business related design decision can trigger the execution of an IT decision. Moreover, a design decision on a certain domain can cause an unanticipated consequence for another domain. The confrontation of this research question makes cross-domain relations of design rationale more explicit.

This research question deals with the EA specific considerations and it will be answered by our EA specialized conceptual framework.

## 1.3 EA Anamnesis: framework or language?

Before we provide the details regarding our research methodology, we consider it important to reflect on some key terms that will help us position our research work. As such, we consider EA Anamnesis as a conceptual framework that:

- restricts (in a controlled language sense (Fuchs et al. 1999, Schwitter 2004)), the dialog that architects and stakeholders may have when rationalizing and capturing design decisions, and
- serves as a base to define a modeling language (as will actually be done to enable case study research). In doing so, it can actually be said that the conceptual framework defines the abstract syntax of the language, where one 'only' needs to add the concrete syntax.

To further illustrate the focus of our research work, we use the framework for 'IS development methods', as originally developed by Seligmann et al. (1989). They make a distinction between *'Way of Thinking'*, *'Way of Modeling'*, *'Way of Working'*, *'Way of Control'* and *'Way of Support'*. The core concepts of our conceptual framework define the *'Way of Thinking'*. The metamodel of EA Anamnesis defines the bridge between *'Way of Thinking'* and *'Way of Modeling'*, in the sense that it makes the *'Way of Thinking'* concrete in terms of the conceptual framework, which essentially defines the abstract syntax of the modeling language that corresponds to the *'Way of Modeling'*. This is in line with the characteristic 'Principles of implementation' (from Gregor and Jones (2007)) as discussed in Section 1.4.2.

EA Anamnesis, as a conceptual framework for the rationalization of EA designs, defines a set of concepts and their relations. When it is a conceptual framework underlying a modeling/controlled language, one could use the words 'ontology' such as e.g. the e3Value community does (Gordijn and Akkermans 2003), or 'metamodel', such as e.g. the MDA community does (OMG 2007). Furthermore, when developing a programming / modeling language, one can typically distinguish between an abstract syntax (not taking notation specific considerations into account), and a concrete syntax which defines the actual notation and associated syntactic conventions (ter Hofstede and Proper 1998).

The ontology/metamodel, or conceptual framework in our case, can be seen to define the abstract syntax of the modeling/controlled language. The conceptual framework for

(EA) Anamnesis can also be regarded as defining (in terms of its metamodel) an abstract syntax for a language to express the rationalization of EA designs.

On top of this abstract syntax, a concrete syntax (notation) may be defined, leading to a full-fledged modeling language. The concrete syntax that we used in our cases, as well as the demonstrator, are considered as 'byproducts' of the research effort. The (EA) Anamnesis framework is, as such, not intended as a full-fledged modeling language with a *concrete* syntax.

Based on this discussion about the position of our work, we admit that we can talk about language as a result, but only have quality claims about the abstract syntax. As we will see in Section 1.4.3, Krogstie (2002) has proposed an approach for the evaluation of the qualities of a language. We use this framework for the evaluation of the *abstract* syntax of EA Anamnesis. As we do not focus on the *concrete* syntax, we do not take issues such as the understandability of notations to end users (Moody 2009), into account.

## 1.4 Research design

Our main goal is the development of a design artifact for the rationalization of EA designs. Such a process requires the execution of certain research steps. As such, we follow the design science research paradigm as indicated by Hevner et al. (2004) and reference process models for design science research as e.g., proposed by Peffers et al. (2007). In Section 1.4.1 we provide the details for it. However, the design research paradigm itself does not provide guidelines regarding the required structural elements of the artifact and the identification of objectives for its creation and evaluation. As such, our research design incorporates the work by Gregor and Jones (2007), more explicitly by identifying the anatomy of a design artifact. By doing so we can define some structural elements of our design artifact. These are discussed in Section 1.4.1. As we will see below, one of the characteristics of these frameworks is the definition of testable propositions of the design artifact. Testable propositions are important for the identification of the objectives of the design artifact and subsequently for execution of the evaluation. Since our main focus is the development of a conceptual framework (abstract syntax of a language), we use the work by (Krogstie 2002) as a basis for the identification of the testable propositions. We will discuss these guidelines in more detail in Section 1.4.3.

### 1.4.1 Research paradigm used in developing EA Anamnesis

In this thesis we aim to develop a conceptual framework for the rationalization of EA designs, which answers the aforementioned research questions. As such, we follow the design science research paradigm, as indicated by Hevner et al. (2004) and Peffers et al. (2007), and as illustrated in Figure 1.1.

The *problem identification and motivation* involves the justification for the development of the design artifact, and the *identification of the objectives* of the design artifact. In our case, Section 1.1 already provided an initial report on the motivation. In Chapter 2, we report on the problem identification, motivation, and objectives, in more detail.

The next step involves the *design and development* of the actual design artifact. In our case, the artifact is a conceptual framework for design rationalization of EAs. Chapters 3 and 4 report on this framework in two steps. First, we report on the generic level for design rationale, and then on the EA specific specialization.

The development of our artifact has been done in an iterative way, as also suggested in Figure 1.1. As such, we have used *demonstration, evaluation and communication* steps in order to gather feedback and further extend and improve our design artifact. For the *demonstration* step, we used a fictitious case study from the insurance sector, taken from the specification of the ArchiMate modeling language (The Open Group 2012).

Concerning the *evaluation* step, we developed a software prototype tool in order to demonstrate to practitioners the potential usefulness of the design artifact, to provide evidence that it can be developed in a software tool and to conduct a computational assessment for it (Chapter 5).

Thereafter, we proceed with real world case study validation (Chapters 6, 7). Our artifact has been applied on two case studies to assess its practical validity. The first case study took place at a Luxembourgish research and technology organization and the second one took place in a Greek e-government organization.

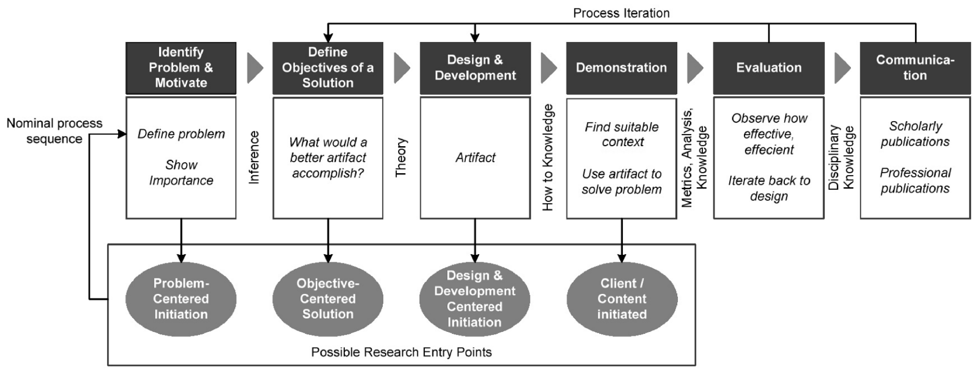

Process Iteration
Identify Problem & Motivate
Define Objectives of a Solution
Design & Development
Demonstration
Evaluation
Communica-tion
Nominal process sequence
Define problem
Show Importance
Inference
What would a better artifact accomplish?
Theory
Artifact
How to Knowledge
Find suitable context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

Figure 1.1: Design science research paradigm (Peffers et al. 2007)

Finally, the *communication* step involves publications and presentations of our design artifact at various scientific and industrial events. This has resulted, a.o. in the publications as listed on Page xvii in the preamble.

### 1.4.2 Anatomy of EA Anamnesis

According to Winter (2008), the concrete outcomes of the design science research process are design research artifacts. In our case, the design research artifact is a conceptual framework to capture design rationale.

Below we provide some structural characteristics that this artifact should exhibit, based on the work of Gregor and Jones (2007) on the anatomy of a design theory:

- *Purpose and scope:* As mentioned above, the development of our design artifact is done in an iterative way. While still being in the first iterations of the development we realized that parts of our artifact address some generic design rationale issues as well. As such, we provide two conceptual frameworks. The first (Anamnesis) addresses those generic design rationale issues and the second (EA Anamnesis) is a specialization of the first one, and addresses the identified specificities of EA.

  However, Anamnesis (the generic part) acts as a basis for EA Anamnesis, while EA Anamnesis will be validated (in terms of the testable propositions as discussed below).

- *Constructs:* The constructs of our design artifact are the core concepts which rationalize EA designs. We identified these concepts by exploring the literature in design rationale, decision analysis and EA and by conducting a survey analysis among EA practitioners that enabled us to testify their perception on the proposed concepts. The identification of these concepts allows us to define a set of key design rationale concepts and to subsequently answer *RQ1.*

- *Principles of form and function:* Our framework will be represented as a metamodel (concepts and their relationships) with associated definitions and explanations. As such we provide operational formulation of what could be done in the real world with the framework.

- *Design mutability:* The generic Anamnesis framework is specialized towards EA. Depending on the specific EA framework selected, this could be further specialized, if so desired. Moreover, depending on the needs, different language notions can be used as materializations of the framework.

Next to the notation used in the cases, as reported in Chapters 6 and 7, one could also think to create, and use, a UML stereotype, or similarly use the specialization/stereotyping mechanism of the ArchiMate language (The Open Group 2012).

- *Testable propositions:* Testable propositions define the criteria that will be used for the evaluation of our design artifact. Moreover, testable propositions are positioned as objectives while being on the 'Identify Problem and Motivate' step of Peffer's research paradigm. As we mentioned before, we base ourselves on the work of Krogstie (2002) to structure the set of testable propositions. We will discuss this in more detail in Section 1.4.3.
- *Justificatory knowledge:* This involves e.g. the (kernel) theories used in designing/creating the artifact (Gregor and Jones 2007). In our case, we base our conceptual framework on theories and techniques from the domains of operations research (decision analysis) for the provision of the reasoning of design decisions in the solution space. Moreover, we use techniques from the domain of goal modeling for the formulation of the design problem in the problem space.
- *Principles of implementation:* The main goal of our framework is the *ex-post* rationalization of EA designs. We argue that our conceptual framework guides architects to structure their decision making and consider more carefully the given goal, principles and requirements during the design process. As such, the framework itself provides guidelines on architecting the enterprise, by taking into account the aspect of rationalization. Furthermore, our software demonstrator enables us to guide practitioners during the capturing of rationale information. For example, practitioners can relate rationale by using specific relationship types, etcetera.
- *Expository instantiation:* To be able to evaluate our conceptual framework in concrete cases, a concrete notation is needed. Therefore, we also provide a concrete syntax. By doing so, we are able to illustrate our framework and to apply it as a language to two real world cases. As such, the cases also provide an expository instantiation of both the framework and the suggested notation. Moreover, we provide a software demonstrator which helped us to create instantiations of the language.

### 1.4.3 Quality characteristics of EA Anamnesis

Krogstie et al. (1995) identifies several quality criteria on models. Some of these have a pendant when talking about the quality of modeling languages and the abstract syntax in particular, since the language should enable/support/restrict the formulation of the models. Krogstie (2002) therefore also translated model qualities to qualities of modeling languages. As such, we make use of the latter framework of qualities for the evaluation and identification of objectives of our design artifact.

The qualities defined by Krogstie, as used for evaluating EA Anamnesis, are as follows:

- *Domain appropriateness:* This quality deals with relationship between the language and the domain. A language should cover the statements of a specific domain as much as possible. For the abstract syntax this leads to the question of to what extent rationalizations can indeed (in 'theory') be expressed (and reasoned about) easily in terms of the concepts defined by the (EA) Anamnesis framework.
- *Participant language knowledge appropriateness:* This quality deals with the level that the domain terminology of a language is well understood by the participants of this domain. This leads to the question of to what extent the users of EA Anamnesis understand the concepts appropriately.
- *Knowledge externalizability appropriateness:* This quality deals with relationship between the domain participant knowledge and the language. The goal is that the knowledge of the domain participants should be reflected as much as possible by the language. This leads to the question to what extent the 'users' of EA Anamnesis recognize their decision making as captured by EA Anamnesis.

Moreover, Krogstie has defined the following two qualities which were not used for the evaluation of EA Anamnesis because, as we will explain below, they cover factors outside the scope of our research:

- *Participant comprehensibility appropriateness:* This quality deals with the relationship between language and the social actor interpretation. The goal is that produced models should be as much as possible understandable by participants. In order to support the understandability of our conceptual framework, we provide a concrete syntax (as used in Chapters 6 and 7) and a software demonstrator (as used in Chapter 5). However, the concrete syntax is a byproduct of our research work and as such, this quality characteristic will not be taken into account for the evaluation of EA Anamnesis.
- *Technical actor interpretation appropriateness:* This quality deals with the technical implementation of a language and more specifically with the provision of automatic reasoning. Concerning EA Anamnesis, we would like to restate, that our main goal is to make the design rationale of EA design explicit. Therefore, we first focus on the conceptualization part of design rationale. As such, we do not evaluate EA Anamnesis in terms of this quality.

## 1.5 Research contributions

The main contribution of this thesis is a conceptual framework for the rationalization of EA designs. Our framework consists of an abstract syntax and as a side product a concrete syntax. The abstract syntax is provided by means of a metamodel and the concrete syntax by means of a notation. Note again, that we focus on the development of the abstract syntax, as per our primary concern of uncovering the key rationalization concepts for EA *RQ1*. The development of a concrete syntax, while important for our own use of the language, is a secondary concern. Therefore, we do not take into account issues like the understandability of notations to end users (Moody 2009).

Our design artifact is a conceptual framework which formalizes design rationalization information. For the development of the artifact we base our work on theories and techniques from the domains of operations research (decision analysis) and goal modeling (for the analysis of the problem formulation). Moreover, we take into account existing design rationale approaches from other domains (e.g. software architecture), and the special characteristics of EA. One of the findings during this development, is that parts of our framework address generic knowledge gaps in the domain of design rationale. Such an example is the formalization of decision making processes. In order to manifest this, our design artifact is presented in two steps: The first step shows a generic conceptual framework for design rationalization. The second step presents an extension of the generic framework, which takes into account the specificities of EA.

Below we provide the description of our contributions:

- *A generic conceptual framework for design rationalization:* We contribute a conceptual framework for design rationalization which is composed of two parts: The first part deals with the problem formulation before the initiation of the actual design decision making process. Here we capture the requirements which have to be taken into account during the design decision making process. The second part deals with the capturing of the design decisions and their rationale.

  More specifically:

  - The provision of rationalization is achieved by conceptualizing the reasoning behind design decisions based on operation research techniques and more specifically Multi-Criteria Decision Analysis (MCDA). Capturing rationale in such a way allows us to 1) structure the decision making process in an analytic way where criteria, their relative importance and constraints are taken into account, 2) to create a repository of structured rationalization information, with which we can retrospectively check the compliance of decisions with the given requirements, and 3) to ex-post compare the captured decision making process with the observed outcome of a decision.

- *A conceptual framework for EA design rationalization:* We contribute a conceptual framework which is an extension of our generic conceptual framework that incorporates the identified specificities of EA.

  More specifically:

  - We incorporate in our conceptual framework the notion of EA perspective. As we will see in Section 2.1.2, EA perspectives define the architectural boundaries within the enterprise and as such they enable us to have a structured way of viewing and defining the enterprise. In our conceptual framework, EA perspectives are used to categorize design rationale and to make explicit its cross-domain relationships, for example how a business decision triggers an IT requirement.
  - We incorporate the notions of goals and EA principles. We use these concepts, which are widely used in the domain of EA for the formulation of the design problem and to reveal as well the reasons behind the elicitation of requirements.

## 1.6 Thesis structure

Our discussion of the research design in Section 1.4, already alludes to the structure of the thesis. In finishing this introductory chapter, we will now briefly describe the overall structure of the thesis.

Chapter 2 provides the justification for the development of a design rationale approach for the domain of EA. We briefly present the domains of EA and design rationale, and their characteristics. Thereafter, through a survey we conducted among EA practitioners (Section 2.3), we investigate the usefulness of some core design rationale concepts and the current state of their usage. Finally, we present the main objectives of a design rationale framework for EA and we show that current approaches lack the characteristics that such an approach requires.

The second part of the thesis presents our main contribution. Chapter 3 presents our generic conceptual framework which deals with the capturing of reasoning behind design decisions and the problem formulation. Subsequently, in Chapter 4 we present EA Anamnesis, a conceptual framework which extends our generic one and deals with the identified specificities of EA. We then use a fictitious case study from the insurance sector to illustrate EA Anamnesis.

The thesis continues with the evaluation and reflection part of our design artifact. Chapter 5 presents a software demonstrator tool which was implemented during the initial phases of the development our design artifact. The implementation of this software demonstrator allowed us to enforce a certain degree of specificity and to further improve our design artifact in the next steps of its development process. As such, the software tool does not reflect the final version of the design artifact but it was used for further improvements of it. The next two chapters present our real world case study evaluations. In Chapter 6 we present the application of EA Anamnesis in a Luxembourgish research and technology organization, whereas Chapter 7 presents the application of our conceptual framework in a Greek e-government organization.

Finally, Chapter 8 summarizes the contributions of this thesis and provides directions for further research.

CHAPTER 2

# Design Rationalization in Enterprise Architecture

This chapter provides the justification for the development of a design rationale approach for the domain of EA. We start by presenting EA and design rationale and their main characteristics. Thereafter, based on a survey we conducted among EA practitioners, we investigate the perceived usefulness of key design rationale concepts as typically are found in literature. The results indicate that practitioners perceive the design rationale concepts as useful, but they do not capture them in an organized way. The chapter continues by presenting the main objectives for the development of a design rationale approach for the domain of EA and showing that current approaches lack the characteristics required by such an approach.

The content of this chapter is based on work published as:

Georgios Plataniotis, Sybren de Kinderen and Henderik A. Proper: EA Anamnesis: Towards an Approach for EA Rationalization. In: *Proceedings of the 2012 Workshop on Domain-specific Modeling*, pp. 27-32, ACM, 2012

Georgios Plataniotis, Sybren de Kinderen, Dirk van der Linden, Danny Greefhorst and Henderik A. Proper: An Empirical Evaluation of Design Decision Concepts in EA. In: *Proceedings of 6th IFIP WG 8.1 working conference on the Practice of Enterprise Modelling (PoEM).* Springer, 2013

## 2.1 What is Enterprise Architecture?

According to the TOGAF specification (The Open Group 2011), *'Enterprise Architecture (EA) is a formal description of a system or a detailed plan of the system at component level, to guide its implementation and the structure of components, their interrelationships and the principles and guidelines governing their design and evolution over time'.*

The definition provided by TOGAF showcases the various aspects which are covered by the domain of EA. As Faller (2016) states, EA can be considered as the structural state of the enterprise, where itself comprises the fundamental concepts or properties of the enterprise (IEEE 2011). It can be also understood as a set of descriptive products that describe the characteristics of existing or desired states of the enterprise (Hoogervorst 2004). Furthermore, EA can be seen as a set of prescriptive products, where the implementation of EA is based on guidelines and principles (Hoogervorst 2004). According to van Steenbergen and Brinkkemper (2009), EA can be considered as the practice of applying a consistent set of rules and models which guide the implementation of EA.

Despite the variations in the definitions of EA, most of the researchers and practitioners (Op 't Land et al. 2008, Lankhorst 2013) agree that EA provides a holistic overview of the enterprise and captures the essentials of the business, the IT and how the enterprise evolves over time. EA facilitates enterprise architects to have an insight into the requirements that originate from different domains of the enterprise and helps them to design solutions that satisfy the given business goals.

Without an established EA capability, stakeholders of the individual domains of the enterprise try to achieve optimization in their own domain of responsibility without considering the 'big picture' of the enterprise (Lankhorst 2013). For example, consider a well established technical infrastructure which is not flexible enough when it comes to supporting a highly changing and agile business environment. IT, instead of supporting the business, will be one of the basic obstacles for the transformation of the business model of the enterprise.

In our work, we follow the descriptive definition of EA (Faller 2016). We consider that EA provides a common language that is understandable by the stakeholders of the different domains of the enterprise and brings together information from these formerly independent domains. This gives the ability to stakeholders to speak the same language in terms of models and tools and their decision making can be improved (Lankhorst 2013).

In the next subsections we discuss some important aspects of EA that we identified in literature.

### 2.1.1 Business-IT alignment

EA is considered as an important instrument for the effectiveness of Business-IT alignment (Lankhorst 2013). *'Business-IT alignment is the state where information technology (IT) is applied in an appropriate and timely way, in harmony with the business strategies, goals and needs'* (Luftman 2004). Business-IT alignment is not obtained by local optimizations but is realized by well-orchestrated interaction of organizational components (Nadler et al. 1992). In other words it is driven by the relationships between components rather than by the detailed specification of each individual component.

The pioneers of the term 'alignment' are Parker and Benson (1989). In their work they emphasize the importance of architecture for the achievement of alignment. The aspects of business strategy and organizational infrastructure compared with IT strategy and IT infrastructure are also emphasized in the well known model of (Henderson and Venkatraman 1993) for strategic alignment. Figure 2.1 presents this model. The model gives various options for the achievement of alignment. For example, one can start with the business strategy and define the IT infrastructure directly. As a matter of choice, one can also consider the IT strategy or the organizational infrastructure. Alternatively, the IT infrastructure can be used as a basis for the definition of the business strategy.

Figure 2.2 illustrates how EA can be used as a supporting instrument for the achievement of alignment. EA intervenes between the strategy and the operations of the organization. After the definition of the mission, vision, strategy and high level goals, EA is used as an instrument to translate these high levels goals into concrete changes to the daily operations of the company. EA provides a holistic perspective on the current and future states of these operations and on the actions that should be taken for the realization of the business goals.

The role of EA as a strategic instrument which guides an organization through a planned development is also discussed by (Ross et al. 2006). Ross considers EA as the organizing

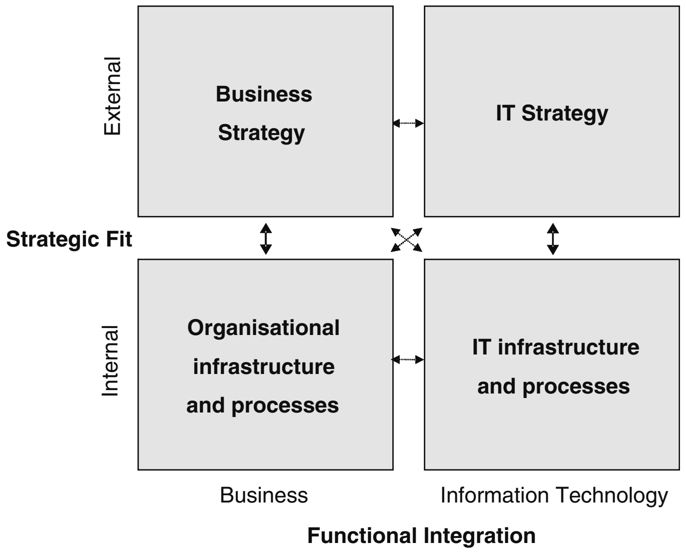


Figure 2.1: Strategic alignment model (Henderson and Venkatraman 1993)

logic for the operational part of the organization. Business processes and the underlying IT infrastructure should be standardized and integrated based on this logic.

### 2.1.2 The notion of EA perspectives

The practice of EA is supported by modeling languages and frameworks. One of the fundamental properties of EA frameworks and languages is that they provide a formal and structured way of viewing and defining enterprises. By formal and structured we mean that the architecture description is comprised of different perspectives and each perspective deals with different aspect of the organization. For example, one perspective deals with the description of business products of an organization and another one with the IT. This description enables stakeholders to focus on specific aspects of their own domain responsibility and at the same time to have a holistic overview of the architecture.

EA frameworks provide a two dimensional scheme which results in two dimension matrices. The intersections (cells) of rows and columns define different perspectives on the enterprise. Below we use two of the most well known EA frameworks to illustrate the notion of EA perspectives.

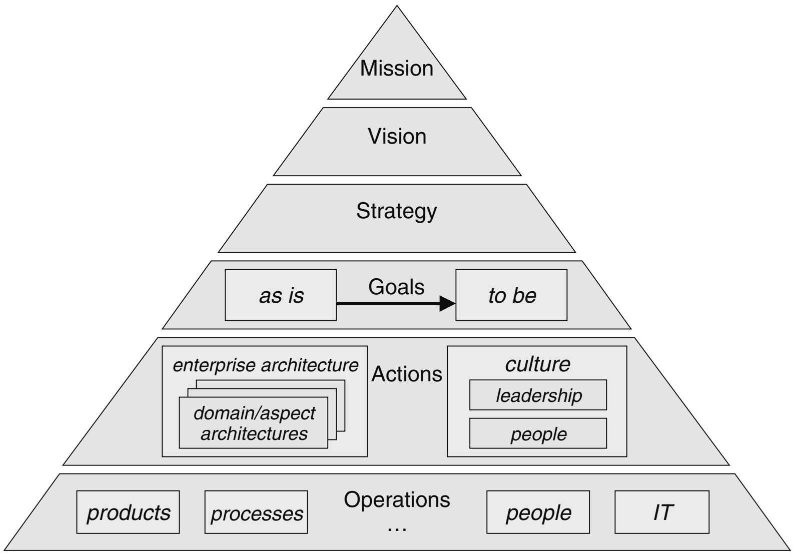


Figure 2.2: EA as a management instrument (Lankhorst 2013)

**Zachman framework:**
The Zachman framework (Zachman 1987) is an enterprise taxonomy which was developed by John Zachman in the 1980s. The framework provides a taxonomy of different representations of the enterprise which are considered important for the development and management of the EA.

Through the Zachman framework, an abstract idea concerning the architecture of the enterprise can be described in different perspectives. As we can see in Figure 2.3, the Zachman framework is comprised by 36 cells where each cell covers a different perspective. The *horizontal* dimension (rows) defines six increasingly detailed views or levels of abstraction. In each of these perspectives different stakeholders are involved. The version 3 of the Zachman framework defines the following perspectives: Contextual perspective (scope contents), Conceptual perspective (business concepts), Logical perspective (system logic), Physical perspective (technology), As Built perspective (tool components), Functioning perspective (operations instances). The deliverables of each perspective can be used as an input to the perspective of next row. For example, the decisions of business executives (Contextual perspective) can be translated to business process models in the Conceptual perspective.

The *vertical* dimension (columns) defines the different abstractions for each of the horizontal perspectives. Version 3 of the Zachman framework provides six interrogatives and

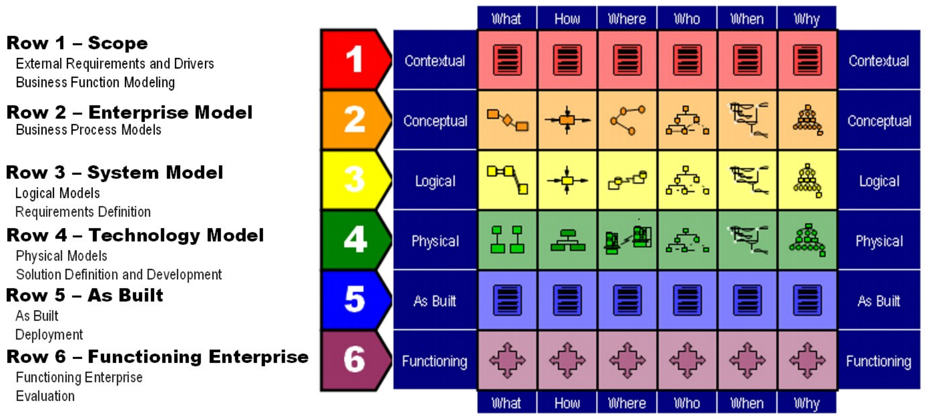


Figure 2.3: The Zachman Framework dimensions and cells (adopted from Zachman (1987))

these are the following: What (inventory sets), How (process flows), Where (distribution networks), Who (responsibility assignments), When (timing cycles), Why (motivation intentions). The idea is that each of this question depicts an independent variable that constitutes a comprehensive description of each of the horizontal perspectives. For example, while being in the contextual phase 'what' columns defines a list of important things for the enterprise, 'how' the business processes, 'where' the business locations, 'who' important organizations', 'when' the various events and 'why' the goals and strategies.

The Zachman framework does not not provide roadmaps or support mechanisms which guide stakeholders during the design process. Rather, it provides a taxonomy of perspectives that stakeholders should take into account while they architecting the enterprise. According to John Zachman, the framework is unique in the sense that perspectives provide a way to explicitly distinguish the architecture elements. However, there is also some critique which states that the framework is purely speculative and non-empirical and that the idea of creating descriptions of enterprise based on Zachman framework is unrealistic (Kim and Everest 1994).

**ArchiMate:**
ArchiMate (Lankhorst et al. 2010, The Open Group 2012) is an EA modeling language that offers an integrated way for describing and visualizing the different architecture domains and their underlying relations and dependencies. ArchiMate is a graphical language.

The ArchiMate language is organized as a two-dimensional framework and it is comprised of nine perspectives. This is illustrated in Figure 2.4. The first dimension (rows) deals with the architectural domains of the enterprise, called layers. These are the business, application and technology. The business layer provides concepts dealing with the products and services of the enterprise and the business processes and functions that realize these products and services. The application layer deals with the IT/application systems, the functions and data that are required to support the business needs. Finally the concepts of the technology layer are used for the modeling of the infrastructure in terms of hardware, networks and system software.

The second dimension (columns) further categorizes the modeling concepts of each layer into three subsets. The first subset has to do with elements of passive structure. Passive structure models the elements on which the behavior is executed. The second subset is about the behavior and models elements of behavior such as a business process. The third subset deals with active structure and models elements which perform the behavior such as an actor.

Such a two dimensional fine-grained classification allows us to model specific EA areas by using a common modeling language. In addition, ArchiMate provides relationships to interrelate elements from different layers of the enterprise.

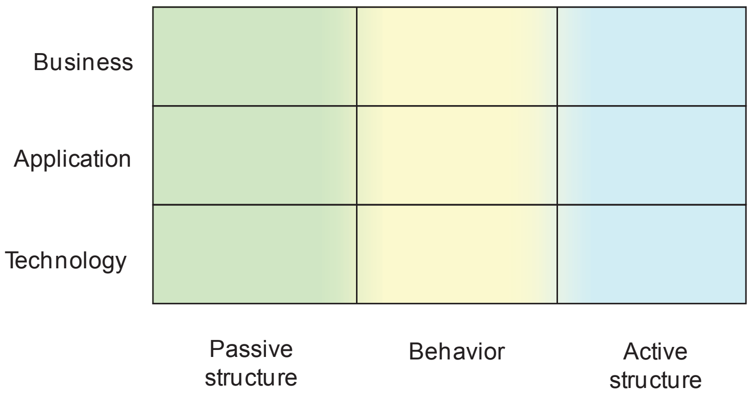


Figure 2.4: The ArchiMate language dimensions and cells (Lankhorst 2013)

### 2.1.3 EA Principles

EA principles are another important instrument for the effectiveness of EA (Greefhorst and Proper 2011). By effectiveness we mean to what extent the objectives are achieved and the problems are solved. TOGAF (The Open Group 2011) defines principles as *'general rules and guidelines that inform and support the way in which an organization sets about fulfilling its mission'*. Principles should be 'enduring and seldom amended'. Principles provide continuity and relative stability in the organization despite the continuously changing and uncertain business environment (Greefhorst and Proper 2011).

EA principles define the future direction of the EA and they are used to guide the decision making processes behind design decisions. Through principles, architects can gradually translate the high level goals of the enterprise into refined requirements which subsequently influence the various decision making processes during the design process. Principles facilitate the decision making processes of architects and prevent situations where a decision is hard to take because of the complex environment and the numerous evaluation criteria (analysis paralysis). Moreover, they document essential choices in an accessible form and they facilitate the communication of stakeholders that are affected.

The number of principles should be low. This is because high number of principles reduce the flexibility of the architecture (The Open Group 2011). Moreover principles should be future oriented and they should be endorsed by senior management stakeholders. Principles provide a firm foundation for the decision making of the EA, they frame policies, procedures and standards and provide the means to resolve conflicting situations during the design process (Greefhorst and Proper 2011).

## 2.2 What is design rationale?

During the design process architects make decisions which have an impact on the design. Modeling languages capture the results of their actions. However, the decisions and rationale behind the design are not captured. Design rationale is concerned with making explicit the underlying decision making and rationale of designs (Lee and Lai 1991). Design rationale provides the underlying justification knowledge behind designs. It can be captured and used during the design process. Designers can use this information during the analysis of existing designs/architectures to better understand the existing (as-is) architecture/design. Additionally, by using this information they are able to explain past decisions to newcomers and therefore facilitate design communication and teaching (Burge and Brown 1998).

An important aspect for design rationale approaches is how the rationale information is captured and represented. Depending on the degree of formality, design rationale approaches can be divided in three main categories: informal, semi-formal and formal (Lee 1997):

- Informal approaches capture rationale by using traditional media such as word processors or even audio and video recordings. The main advantage is that the design rationale can be captured easily in a format that stakeholders are familiar with. No special tools are needed. However, the main drawback of informal approaches is that rationale information is not organized and it is hard to be processed and interpreted in a computer based system (Lee 1997). As such, in the case that stakeholders want to use the design rationale, they have to spend significant time in order to find the information they are interested in. Moreover they do not have insight of the various relations of information. For example, which requirements were used for a specific design decision.
- Formal approaches structure the rationale information through a strict format in order to be easier for computers to interpret and process that information. A common problem of this kind of approaches is that the contents are hard to understand by human beings and the process of capturing the information requires more effort (Burge and Brown 1998).
- Semi-formal approaches aim to combine the advantages of the aforementioned types. The rationale information is structured up to a certain degree in order to be processable by computers, but at the same time it can be understood by human beings as well. In most cases, a system suggests the chunks of information that should be captured and the user captures the information by following specific instructions of the system.

Various design rationale approaches have been proposed by the research community in various domains (civil engineering, mechanical design, artificial intelligence, software engineering, and human-computer interaction). For example, civil engineering design rationale is used to coordinate the activities of stakeholders in different areas of a construction project. Based on design rationale the stakeholders can understand and respect the ideas of others and resolve possible conflicts (Whelton et al. 2001). Another good example is software engineering. Design rationale is used there to support the process of requirements analysis and to capture the design decisions made during new designs (Dutoit et al. 2006). Design rationale can also be used by stakeholders who missed an important project meeting and want to have insight into a particular topic. Possibly unresolved issues captured by design rationale can be discussed in future project meetings (Dutoit et al. 2006). Another important application of design rationale is that it can be used by designers to avoid the same mistakes observed in a previous design iteration and to avoid duplication of work (Jarczyk et al. 1992).

## 2.3 Need and current practice of EA design rationalization

In this Section we present a survey study which investigates how EA practioners perceive some key design rationale concepts. As stated in EA literature, capturing design rationale is quite essential for EA descriptions (Ross et al. 2006, Lankhorst 2013). An EA description should not only refer to the relationships between business and IT, but also to the architectural decisions that lead to specific EA elements. According to Lankhorst (2013), the recording of rationale related to traceability, accountability etcetera and the documentation and revisiting of rejected alternatives are important actions. Another useful characteristic is the capability to capture the relationships of design decisions with the given business goals and requirements. More specifically how the architecture choices are contributing to the achievement of the business objectives.

Lankhorst (2013) states that the process of capturing design rationale during the design activity helps the architects to externalize some of their intuitive decision making. This can result in increased awareness of their actions and subsequently in more rationalized choices. Moreover, capturing design decisions and rationale during the design process enhances the collaboration among stakeholders of various domains (Business, IT, etcetera) because they are informed in advance about the upcoming changes in the architecture. As such, they have the opportunity to intervene in the process. Sharing such an information at a later stage has negative consequences for the commitment of stakeholders (Lankhorst 2013).

Furthermore, anecdotal interviews conducted with enterprise architects also suggest that it is important, for their role in the organization and the profession of enterprise architects in general, to capture and maintain design rationale in the organization. Architects are usually asked by the management for the justification of designs. Therefore, an architect should be always prepared to clarify better the goals of a particular architectural change and provide detailed information to the relevant stakeholders (Lankhorst 2013).

Motivated by the aforementioned indications, we conducted a survey (see Appendix Survey Study) with regard to the potential usefulness of a design rationale approach in the domain of EA. In that survey thirty five EA practitioners participated in total. Through the survey we tested how the stakeholders perceive in terms of usefulness some key design rationale concepts and compared the results to their current uptake in practice. It is worthwhile to mention that the identification of the challenges for the development of a design rationale approach was not amongst the objectives of this study, since it was conducted at a stage where we had already identified the key challenges and the initial set of concepts.

Our results go further and show that architects currently rationalize architectural decisions in an ad hoc manner, forgoing structured templates. Finally, we interpret the survey results by discussing for example possible reasons for the gap between perceived usefulness and uptake of EA rationale.

### 2.3.1 Study setup

*Participants:* Participants were gathered during a professional event on EA organized by the Netherlands Architecture Forum (NAF). NAF is a leading Dutch (digital) architecture organization, concerned with the professionalization of Enterprise and IT Architecture. A total of sixty five people started the survey, thirty five out of which actively finished the study. The majority of the participants were of Dutch nationality, had at least several years (more than ten) of professional experience in EA, and were fluent in English.

*Materials:* The questions and input used for this survey are derived from previous research and professional workshops on the use and creation of architecture principles in Dutch knowledge management and enterprise modeling organizations. The data analyzed and used for this study are derived from a subset of the embedding survey, which contained additional sections dealing with other, related, factors of architecture principles creation and use (Greefhorst et al. 2013). Both surveys dealt with factors that rationalize EAs such as principles, design decisions etcetera. All questions were presented in English because non-Dutch speakers were expected. Furthermore, the survey was planned to be extended to other European countries afterwards.

*Method:* The survey consists mostly of structured and closed questions. The participants were given the context that the questions dealt with a larger area of architecture principles. In particular, we explained to them the fact that principles provide a foundation for EA design decisions, and which factors are important for such decisions.

Thereafter, we presented to the practitioners a set of basic concepts that could potentially be used for design rationale in EA. More specifically, practitioners had to provide their feedback on some key concepts that would potentially enrich the provision of rationale information of EA design decisions. These are the following: ‘Rationale’, ‘Rejected alternatives’, ‘EA layer’, ‘Unanticipated observed impact’ and ‘Design decision traceability’. It is worthwhile to mention that we avoided using generic terms such as ‘EA perspective’ to indicate the categorization of design decisions and their rationale. We used ‘EA layer’, which represents one of the dimensions of ArchiMate language, because it is a term widely used by EA practitioners, especially of those who are familiar with the ArchiMate.

To investigate to what extent the aforementioned concepts are grounded in reality, we queried for them (a short explanation for each concept was provided) whether participants considered it to (1) help with the maintenance of an EA, (2) help them to justify an EA, and (3) be currently actively documented in the participant’s organization or professional experience.

For each of the three directions, participants could answer whether they disagreed, agreed, or strongly agreed with usefulness of the given concept. The format of the answers was adopted from the embedding survey, which was executed by an outside party. The outside party had already structured the answering format of this survey and as such we adopted the same answering format in order to avoid any potential confusion as much as possible.

To follow up on the current practical state of design decisions, we then enquired whether any standardized approaches or processes existed for the capturing of EA design decisions. Participants were given the choice of stating whether, for their organization, such approaches either exist, do not exist, or whether they were uncertain of their existence. In the case of nonexistence of documentation approaches, participants had the option to expose the reasons for this through a hybrid structure with predefined answers as well as free text comments.

*Data analysis:* The data resulting from the main questions (whether the rationalization concepts help in the maintenance and justification of an EA and whether they are documented) were quantified by assuming that ‘strongly agree’ implied ‘agree’, and that ‘strongly agree’ could be treated as ‘agree’. Based on this, we calculated the total amount of ‘agree’ and ‘disagree’ answers for each of our concepts. Of course, questions that were not filled in were disregarded in our calculation. While the size of groups of participants for each dimension (maintenance, justification, documentation) (resp. 33, 34 and 35 participants), and comparison between them should thus be a valid endeavor, care should still be taken not to assume they represent a breakdown of opinions in the exact same group. The data resulting from the question regarding the use of standardized templates for documenting design decisions were analyzed in a straightforward way, calculating the percentages of yes, no and uncertain answers for the group of (n=35) participants who answered this question.

### 2.3.2 Survey limitations

The main difficulty in executing this study was that our questions had to be integrated into a existing study, of which the structure and answering format were already determined. Unfortunately, the opportunity to conduct a dedicated survey regarding design rationale with such a number of participants was quite limited due to time constraints of practitioners. Therefore, we had a limitation regarding the number of questions we could incorporate into this wider study.

Thus, in order to ensure that participants would not feel confused by a different question and answering format, we had to deal with a suboptimal set of answers for our first question. Ideally, the questions of whether certain concept applies to a given dimension (i.e. maintenance, justification, documentation), would be done on a Likert scale, with equal amounts of negative and positive answers. However, as the goal of the wider survey was to elicit as much (strong) opinions as possible from practitioners, it was chosen to use an answering format which contained no neutral grounds and thus forced people to make a polarized choice.

We will take these issues into account during the analysis of our data, and attempt to account for the possible loss of nuance.

### 2.3.3 Results

Tables 2.1, 2.2, 2.3 show the survey results on to what extent design rationale concepts help the EA practitioners (1) to maintain the architecture, (2) to justify the architecture and (3) to document design rationales in current practice.

For each question, we provide a division into ‘positive’ and ‘negative’, and a subsequent division of ‘positive’ into ‘agree’ and ‘strongly agree’. We do this for the sake of transparency: on the one hand, we want to show aggregated results on positive reactions to a concept, but on the other hand we do not want to hide that the questions were posed in a possibly biased manner (as discussed in Section 2.3.2).

Furthermore, Table 2.4 shows us to what extent practitioners use standardized templates to capture design rationale. In case practitioners forgo the use of standardized templates, Table 2.5 shows why this is so, by means of closed answers (such as ‘no time/budget’) and open answers (whereby the architects could provide a plain text description such as ‘EA is not mature enough’).

Table 2.1: To what extent study participants (n=35) find that design rationale concepts help with the maintenance of the EA.

| | Helps with the maintenance of EA | | | |
|---|---|---|---|---|
| Concept | Negative | Positive | Positive-Agree | Positive-Strongly agree |
| Rationale | 9% | 91% | 42% | 49% |
| Rejected alternatives | 26% | 74% | 43% | 31% |
| EA layer | 9% | 91% | 46% | 45% |
| Unanticipated observed impact | 23% | 77% | 43% | 34% |
| Design decision trace-ability | 14% | 86% | 40% | 46% |

Table 2.2: To what extent study participants (n=35) find that design rationale concepts help with the justification of the EA.

| | Helps with the justification of EA | | | |
|---|---|---|---|---|
| Concept | Negative | Positive | Positive-Agree | Positive-Strongly agree |
| Rationale | 18% | 82% | 29% | 53% |
| Rejected alternatives, | 29% | 71% | 44% | 27% |
| EA layer | 38% | 62% | 38% | 24% |
| Unanticipated observed impact | 18% | 82% | 50% | 32% |
| Design decision trace-ability | 26% | 74% | 44% | 29% |

### 2.3.4 Discussion

Generally, the results from Tables 2.1, 2.2 indicate that practitioners perceive that the given design rationale concepts will help them with the maintenance and justification of EA designs. This can be concluded from the fact that, for each concept, the majority of architects agree with its usefulness for both maintenance and justification. Yet,

Table 2.3: To what extent study participants (n=33) currently document design rationale concepts.

| | Current documentation practice | | | |
|---|---|---|---|---|
| Concept | Negative | Positive | Positive-Agree | Positive-Strongly agree |
| Rationale | 30% | 70% | 55% | 15% |
| Rejected Alternatives | 73% | 27% | 27% | 0% |
| EA Layer | 21% | 79% | 40% | 39% |
| Unanticipated observed impact | 73% | 27% | 24% | 3% |
| Design decision traceability | 58% | 42% | 36% | 6% |

Table 2.4: To what extent study participants (n=35) use a standardized template for documenting EA design decisions.

| Question | Not aware | Yes | No |
|---|---|---|---|
| Does your organization use a standardized template for documenting EA design decisions? | 23% | 40% | 37% |

the results from Table 2.3 indicate that despite the fact that design rationale concepts are considered useful, the majority of them are not documented by practitioners. While practitioners indicate that they document the rationale for a decision (70%) and its architecture layer (79%), the majority of them do not document neither their unanticipated observed impacts, nor their traceability, nor rejected alternatives.

Moreover, in cases where practitioners document decisions, 40% of them use standardized templates for documentation, while 23% of them are not aware of the existence of such templates. The remaining 37% of practitioners, that do not document design rationale or that do not use standardized templates, think that standardized templates are not useful (30%), or that there are no available resources in terms of time/budget (3%), or that there no suitable tool for this (9%). Furthermore 58% of the practitioners do not use standardized templates, either because they feel covered by documenting design decisions

Table 2.5: The proportions of the reasons that practitioners (n=13) do not use standardized templates for documenting EA design decisions.

| | |
|---|---|
| Not useful | 30% |
| No time/budget | 3% |
| No suitable tool | 9% |
| Other comments: | 58% |
| Design decisions are documented inside PSA/PEA (Word or Powerpoint) | |
| Depends mostly on the client | |
| EA is not mature enough | |
| Our organization is not mature enough when it comes to EA | |
| General immaturity of EA departments | |
| We use several templates, but they are not exactly the same | |
| Company standard is the TOGAF template | |

inside MS Word/Powerpoint, or because they insist that EA is not a mature practice in the organization.

A possible reason for the currently limited practice of design rationale in EA is that practitioners are insufficiently *aware* of the potential usefulness of design rationale techniques. This may be caused by the relative immaturity of the EA field compared to areas in which decision rationale and tool support are well established, such as the field of software architecture.

We now discuss our findings per given concept:

**Rationale:** The 'Rationale' concept, which captures the reasoning behind a decision, is considered as an important concept by the majority of participants. More specifically 91% believe that this concept helps with the maintenance of the EA, and 82% believe that it helps to justify existing EAs. Interestingly though, as opposed to other concepts, the current practice of documenting rationale of decisions is quite high (70%). We argue that this happens, because architects usually have to justify their design decisions to other stakeholders and to the management of the organization.

**Rejected alternatives:** The majority of participants (74%) acknowledge that capturing the rejected alternatives information assists them with the maintenance of the EA and

71% of them think that they are helped with the justification of it. Practitioners seem to understand that this information provides a better insight into the rationalization process. We speculate that rejected alternatives, in combination with selection criteria, provide them with additional rationalization information by indicating the desired qualities which were not satisfied by these alternatives.

However Table 2.3 indicates that only 27% of the participants capture rejected alternatives. We reason that the effort of capturing rejected alternatives in combination with the ignorance of the potential usefulness of this information demotivate practitioners to document rejected alternatives. Even if this information is documented, the added value provided is not so high due to the lack of specialized tool support. This is because the rationale is captured in an unstructured manner and therefore it is difficult to search through it. Moreover, when rejected alternatives are combined with other rationale concepts (such as decision criteria), it does allow one to better trace the decision making process by gaining insight into the criteria that were considered during this process.

**EA layer:** 91% of the participants agree that the concept of the 'EA layer' helps them with the maintenance of an EA. The percentage of participants that agree that this concept helps them to justify EAs is 62%. Although the percentage itself is quite supportive, we can observe quite a big variation compared to that agreeing that 'helps with maintenance'. We argue that this is because the 'EA layer' concept is not a justification concept itself, but when it is combined with the other design rationale concepts, in the context of a specialized tool, it can actually contribute to the rationalization.

**Unanticipated observed impact:** The majority of participants (77%) recognize that the explicit information of unanticipated observed impacts helps them with the maintenance of the EA. We speculate that participants, while they maintain existing architectures, are expected to use information of the unanticipated outcomes of past decisions in the enterprise to avoid past mistakes. Furthermore, 82% of enterprise architects agree that the unanticipated observed impact concept helps them with the justification of the architecture.

An interesting finding is that participants recognize the usefulness of capturing the unanticipated observed impact, but only the 27% of them has a standard practice to document this concept. We believe that when an unanticipated outcome of a design decision is observed, practitioners are focused on immediately solving this issue. From a short term perspective, the documentation of this observed impact is a minor issue for them. However, in the long term, the documentation of observed impacts raises awareness of unanticipated outcomes. Another reason could be the lack of a structured environment for architectural rationalization, which would allow architects to interrelate unanticipated observed impacts to design decisions across the EA (for example unanticipated impact of an IT decision on a business process).

**Design decision traceability:** A majority of the participants (86%) find that capturing the traceability of design decisions can assist them with the maintenance of the EA. Moreover, 74% indicate that traceability helps them with the justification of the EA.

Regarding the documentation practice, some of the practitioners (42%) capture the traceability of design decisions, but still the majority of them (58%) do not. In our view, this indicates a tendency of practitioners to rationalize EA designs through traceability. However, we think that documenting decision traceability is still limited since architects lack structured ways to capture design rationales, as we can see in Table 2.5.

### 2.3.5 Survey conclusion

Through this survey, we reported on the perception of EA practitioners to a set of basic design rationale concepts. We found that the given concepts are largely perceived as useful to architectural practice. Yet, we also found that the uptake of rationalization in practice is currently limited to only a few concepts, prominently 'rationale'. Furthermore, these few concepts are captured in an ad hoc manner, forgoing structured rationalization approaches and tools.

Finally, we speculated on (1) the distinction between perceived usefulness of rationalization concepts on the one hand, and the uptake in practice on the other, and (2) the current limited use of a structured template for rationalization. A possible explanation is the relative immaturity in the field of EA, compared to fields where rationalization is well accepted, such as Software Architecture. Such immaturity manifests itself the lack of awareness of rationalization, including recognizing its potential usefulness for tracing design decisions, as well as in a lack of structured templates for documenting design decisions in EA.

## 2.4 Objectives of a design rationale approach for EA

As we have seen in Section 2.3 a possible reason that demotivates EA practitioners to capture rationalization information is the lack of a standardized way to document design decisions. Despite the fact that a plethora of design rationale approaches (Tang et al.

2007, Tyree and Akerman 2005) have been implemented in other domains such as software architecture, the domain of EA still remains unexplored.

Below we provide a set of identified objectives for the development of specialized design rationale approach for EA.

### 2.4.1 Relationships of design rationale across the enterprise

As we have seen in Section 2.1, EA provides a structured way (through various perspectives) to view and define the enterprise. Moreover, EA supports business-IT alignment by making the various dependencies among perspectives explicit. By doing so, stakeholders are helped to realize the business objectives into concrete architecture changes in a timely and appropriate way. Formerly independent domains (Business, IT) should be brought together so that stakeholders are able to satisfy by means of design decisions the given requirements. This type of traceability from design decisions to requirements is of high importance the domain of EA. The need for traceability is also revealed through our survey study. Therefore, we argue that a design rational approach for EA should be able to capture and make explicit the various relationships of design rationales across the architecture. Such kind of relationships are the satisfaction of requirements by design decisions and the translation of requirements across the various perspectives of the architecture (Lankhorst 2013, Proper and Op 't Land 2010). For example, consider the translation of a requirement of a specific perspective (business) to a requirement of another perspective (application) and the design decisions that were taken in order to address this requirement. By doing so, we can have a holistic design rationalization overview where we can analyze how the design problem was formulated across the enterprise and which design decisions were taken to address it.

### 2.4.2 The role of requirements in design decision making

Decision making in EA involves the consideration of requirements from various domains. This is not always an easy process, since enterprise architects have to balance between conflicting requirements coming from those domains (Greefhorst and Proper 2011). Moreover, situations like budget constraints or lack of time also influence the decision making process. Instead of evaluating the quality characteristics of alternatives, decision makers have to take into account these constraints and propose appropriate solutions. A design rationale approach for EA should capture which requirements and constraints were taken into account during the design decision making process as well as their importance and role during that process. By doing so, we can have an in-depth analysis of the reasoning behind individual design decisions and we can check to what extent they comply or not with the given requirements.

### 2.4.3 Traceability between the design and its rationalization

An EA design comprises of elements in different parts of the architecture (e.g. business-IT). These artifacts should support in a harmonic way the realization of the business objectives of the enterprise (Lankhorst 2013). When it comes to design rationalization, an important capability is the traceability between the various enterprise elements and the rationalization information (Tang et al. 2007). A design rationale approach should provide traceability from the EA elements of the design to their corresponding rationalization and vice versa. By doing so, enterprise architects can have bidirectional traceability. They can start their analysis by inspecting the design decisions and their underlying rationales and then have an overview of the enterprise elements that result from those decisions. Alternatively, they can start by inspecting specific EA elements and then have an overview of the design decisions that constitute these elements.

### 2.4.4 Unanticipated observed impacts across the EA

EA facilitates the structural impact analysis during an architecture change. Enterprise architects can check during an architectural change or modification of a given architecture element which other elements are possibly influenced. This analysis assists stakeholders with their design decision making process (Lankhorst 2013). However, it is not possible to anticipate all the outcomes/consequences of design decisions. During the execution or maintenance phases some of these decisions prove to be false and result in unanticipated consequences on the architecture. These situations should be recovered by enterprise architects by taking new design decisions (Proper and Op 't Land 2010).

As it is also indicated in our survey study, a design rationale approach for EA should capture these unanticipated observed impacts across the architecture. Moreover, it should capture the relationships between the unanticipated observed impact and the decisions that caused and resolved this. By doing so, enterprise architects (especially newcomers) can learn about previously problematic situations and the vulnerabilities of the EA and then can subsequently avoid repeating the same mistakes in future design process iterations.

## 2.5 Existing design rationale approaches

In this section, we present existing design rationale approaches and we show that current approaches lack the characteristics that a specialized approach for EA requires. We first present goal-oriented approaches and argumentation-based approaches . Then, we discuss about the state of the art of the domains of EA and software architecture. We use bold letters to emphasize on the aspects that are not covered by existing approaches.

Goal-oriented requirements engineering approaches propose mechanisms for the definition, documentation and maintenance of requirements (van Lamsweerde 2001, Yu 1997). However, EA specificities such as **the categorization of requirements in various perspectives and the relationships among them are not addressed**. Moreover, despite the fact that some concepts from goal-oriented modeling can be used to describe design rationales, **they do not cover entirely the rationalization of the solution space. For example, they do not take into account the role of requirements for the provision of rationale for design decisions**. Last but not least, there is no distinction between high level strategic goals (for example, 'make more profit') and architecture level goals (for example, 'application scalability').

Moreover, argumentation-based approaches, the Decision Representation Language (DRL) (Lee 1991) and Issue Based Information System (IBIS) (Kunz and Rittel 1970) are two well known approaches for capturing design rationale. Both DRL and IBIS are inspired by Toulmin's analysis of argumentation (Toulmin 2003) and argumentation maps. For DRL and IBIS, key rationalization concepts are the issue, the arguments and the resolution of design argumentation. Here, for example, resolutions are similar to Toulmin's conclusion of an argument.

However, argumentation-based approaches **do not make an explicit relation to the design artifact under consideration**, while for us it is important to focus our rationalization on particular EA elements (such as elements of architectural languages). Furthermore, in a more general context, argumentation based approaches are not suitable for capturing communicating design rationales in practice (Shipman and McCall 1997). This is mainly because argumentation-based approaches require extensive documentation (Shipman and McCall 1997). Last but not least, argumentation-based approaches lack formality which is not amenable to computer-based support.

With regard to the domain of EA, there are other specialized approaches such as TOGAF (The Open Group 2011) or IAF (van't Wout et al. 2010) which aim to rationalize the decision making in EA, but from an process oriented perspective. For example TOGAF's Architecture Development Method (ADM) cycle, provides a roadmap for the execution of the decision making during the enterprise transformation starting from the architecture vision, development of the various parts of the architecture business-IT and defining appropriate migration and change management plans. In other words these frameworks aim to rationalize **the way** that the architectural process is executed **but not the outcome of this process which are the individual design decisions**.

ArchiMate (The Open Group 2012) features, from its second version, a motivation extension. The motivation extension is used to model the reasons behind architectural changes, but lacks concepts common to existing rationalization approaches. For example, the motivation extension does not capture explicitly the rationalization behind the selection of specific EA elements. As such, **design alternatives, the role of requirements during the decision making process and unanticipated consequences of design decisions are not taken into account**

Additionally, there also exist design rationale approaches for Software Architecture (Jansen and Bosch 2005, Tang et al. 2007, Tyree and Akerman 2005, Kruchten 2004, Savolainen 1999). These approaches are template based or model based. Akin to argumentation based rationalization approaches, template based approaches (Tyree and Akerman 2005, Savolainen 1999) describe *in textual format* elements of architecture decisions such as 'Rationale', 'Issue', 'Implications' etcetera. Differently, model based approaches (Jansen and Bosch 2005, Tang et al. 2007, Avgeriou et al. 2011, Kruchten 2004) provide a *formal metamodel* of decision rationalization concepts, thus enabling computer-processable rationalization. However, software architecture is only a subset of the domains of EA (Lankhorst 2013). Issues, like **consideration of requirements coming from various domains of the enterprise (i.e. Business-IT) during the decision making process or the implication of design decisions to other domains of the enterprise are not covered by these approaches**.

At this point, we would like to state that certain parts of the aforementioned approaches fulfill some of the objectives that were described in Section 2.4. As such, as we will see in Chapters 3, 4, we have taken into account these specific aspects for the development of our conceptual framework. A characteristic example of such an approach is the Traceability metamodel (Avgeriou et al. 2011) that provides design rationale support in the domain of Software Architecture. The Traceability metamodel is depicted in Figure 2.5. The authors provide a distinction of rationale between the problem space and the solution space which enables us to focus on different aspects of design rationale. This is very useful for the domain of EA as well, since it allows us to analyze the requirements that formulate the design problem and how the problem is addressed be design decisions. As will see in Section 3.2.2, we used such a distinction for our work. However, the approach does not touch upon on other identified objectives like the role of requirements during the decision making process or the categorization of requirements per architectural domain.

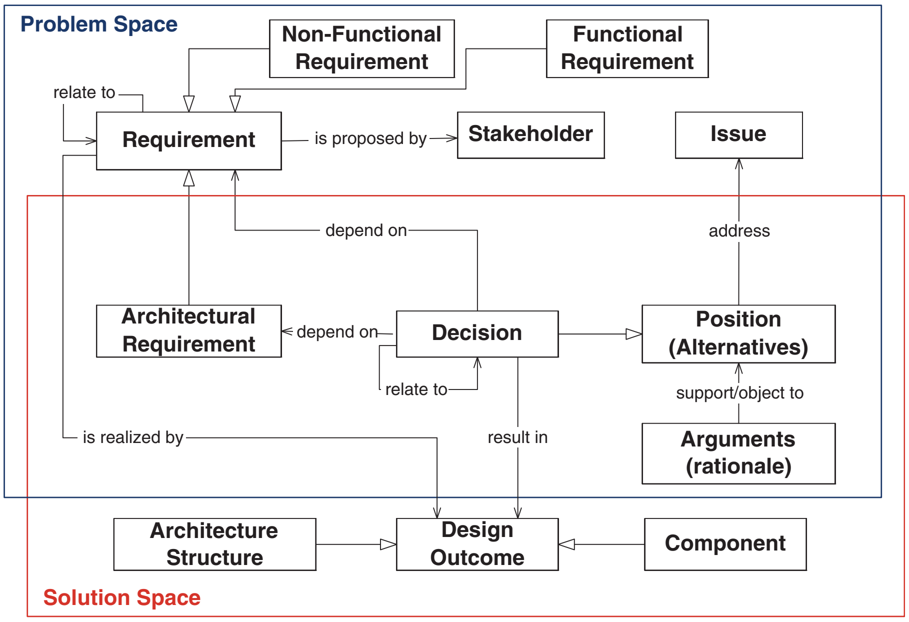


Figure 2.5: The Traceability metamodel (Avgeriou et al. 2011)

For the latter we used the structural approach that EA provides for the categorization of EA elements. By doing this kind of combination, we tried to reuse as much as possible of the existing state of the art and provide a better grounding to our work.

## 2.6 Summary

In this chapter we presented the main characteristics of EA and design rationale. Thereafter we presented the findings from a survey for the current practice and need of design rationale in the domain of EA. Results indicated that the lack of a specialized approach demotivates practitioners from capturing design rationales despite the fact that they found the proposed rationalization concepts as useful. Motivated by the characteristics of EA and the survey, we presented the main objectives for the development of a design rationale approach for EA. Finally, we enlisted relevant design rationale approaches and we explained what they miss in order to cover the domain of EA.

# Part II

# EA Anamnesis – Modeling EA Rationalization

# CHAPTER 3

# A generic conceptual framework for design rationalization

As discussed in Section 1.5, parts of our conceptual framework for the rationalization of EA designs also address generic aspects of design rationalization i.e., aspects independent of the EA domain. To account for this in this chapter we introduce Anamnesis, a generic conceptual framework for design rationalization.

Anamnesis provides insights both into the problem and the solution space of a design. In the problem space, it captures the role of requirements for the formulation of the design problem and in the solution space it captures the design decisions and their rationalization. Furthermore, it reveals the intertwining between the two spaces by making explicit how the requirements influence the design decision making processes and how the design decisions and their possible unanticipated consequences motivate the elicitation of new requirements.

The content of this chapter is based on work published as:

Georgios Plataniotis, Sybren de Kinderen and Henderik A. Proper: EA Anamnesis: Towards an Approach for EA Rationalization. In: *Proceedings of the 2012 Workshop on Domain-specific Modeling*, pp. 27-32, ACM, 2012

Georgios Plataniotis, Sybren de Kinderen and Henderik A. Proper: Capturing Decision Making Strategies in EA - A Viewpoint. In *Proceedings of EMMSAD 2013 (Exploring Modelling Methods for Systems Analysis and Design)*, pp. 339-353, LNBIP, 2013

Georgios Plataniotis, Sybren de Kinderen and Henderik A. Proper: Relating Decisions in EA Using Decision Design Graphs In *Proceedings of EDOC 2013 (Enterprise Distributed Object Computing)*, pp. 139-146, IEEE, 2013

Georgios Plataniotis, Sybren de Kinderen and Henderik A. Proper: EA Anamnesis: An Approach for Decision Making Analysis in EA. In: *International Journal of Information Systems Modeling and Design (IJISMD)*, vol. 5, no. 3, pp. 75-95, IGI Global, 2014

Georgios Plataniotis, Sybren de Kinderen, Qin Ma and Henderik A. Proper: A Conceptual Model for Compliance Checking Support of EA Decisions. In: *Proceedings of 17th Conference on Business Informatics (CBI)*, vol. 1, no. 3, pp. 191-198, IEEE, 2015

## 3.1 Introduction

In Chapter 2 we have discussed our observation on how design rationale supports architects with regards to the maintenance and justification of EAs. Without design rationale, new designs might be constructed in an ad-hoc manner, failing to take into consideration constraints implied by past design decisions. Furthermore we identified, through a survey, that we conducted among practitioners (Section 2.3), the potential usefulness of rationalization for the domain of EA, especially when it comes to the maintenance and justification of design decisions.

In this chapter we introduce Anamnesis, a conceptual framework which captures and analyzes the decision making behind designs. Anamnesis originates from the Greek word *ανάμνησις* (/ˌænæmˈniːsɪs/), which denotes *memory* and *repair of forgetfulness.*

As we stated in Section 2.5 our conceptual framework was developed by combining, as much as possible, useful elements from existing approaches. Anamnesis is primarily grounded in well established techniques from the domain of Operations Research (OR), more specifically on Multi-Criteria Decision Making Analysis (MCDA), for the provision of reasoning and the analysis of decision making strategies behind decisions (Hillier 1995, Hansson 1996).

At this point, we would like to state that during the exploration of the domain of OR, we identified relevant and even more advanced approaches like the analytic hierarchy process (AHP) (Saaty 1988) and ordered weighting average (OWA) (Yager 1988). However, since we were aware of the lack of design rationale support for the domain of EA, we decided as a first step to provide a relative simple decision scheme which would be easily applicable and adopted by EA practitioners, but on the same time it would facilitate the structuring of their reasoning during the decision making process. By using this scheme, we provide insight into the various ways that decision makers used in order to evaluate alternatives. Moreover, we use goal modeling techiques for the representation of the design problem (Loucopoulos and Karakostas 1995, Yu 1997). Finally, based on the notion of intertwining between problem and solution space (Nuseibeh 2001, de Boer et al. 2007), we bridge the two spaces and provide an insight into how they influence each other.

Anamnesis enables *ex-post* rationalization of a design since it provides insight into the reasoning behind the executed design decisions; how the given requirements influenced the decision making process; what where (if any) the unanticipated observed impacts of design decisions; and how they were resolved. Moreover, the framework has the potential to support decision makers during the design process. In that way, they can formulate their design problem by means of requirements and they can structure their decision making based on a decision making strategy and by taking into account the given requirements. However, in this thesis our main focus is on the ex-post rationalization of EA designs. Furthermore, as we also stated in Section 1.3, the main purpose of our conceptual framework is to make design rationale explicit by capturing the underlying reasoning behind decisions. Regarding the expression of reasoning (as we also state in Section 1.3) we have defined a concrete syntax (Chapter 4) which is mainly used for the demonstration of expressing design rationale during the case studies and the demonstration of our software tool. As such, this is mainly a byproduct of our research.

Anamnesis enables stakeholders who were not involved in the past in a design process (e.g. newcomers), to have a holistic overview of the past design process and to understand what was the design problem and what their predecessor did in order to solve it. In the case of unanticipated observed impacts of design decisions, stakeholders can compare the unanticipated observed impact with the rationalization behind the decision and ascertain possible omissions of the past decision making process. We argue that this type of insight prevents designers from repeating the same bad practices and subsequently helps them with their future decision making.

This chapter is organized as follows. Section 3.2 introduces the Anamnesis conceptual framework. In Sections 3.2.2,3.2.1 we briefly present MCDA and how the two spaces interact with each other. Thereafter, we present in detail the concepts and relationships of the problem and solution spaces (Sections 3.2.3 and 3.2.4). Finally Section 3.3 concludes.

## 3.2 The Anamnesis conceptual framework

As we saw in Section 1.4, the planning and creation of a design science artifact is based on the use of the design science research paradigm (Gregor and Jones 2007, Winter 2008, Fischer et al. 2010). In accordance with that, we developed our framework in an iterative manner by taking into account the identified objectives of Section 2.4 as well as our observations during the demonstration and application of our framework in a real world context. Before the presentation of the concepts and relationships of our conceptual framework, we briefly describe multiple-criteria decision analysis which is the kernel of our conceptual framework and the idea of intertwining between problem and solution space.

### 3.2.1 Multiple-criteria decision analysis

In our framework we go a step further towards the formalization of design rationale by capturing the decision making processes and strategies behind the selection of design decisions. More specifically, the solution space formalizes the rationale which leads to specific design decisions based on MCDA. MCDA is an operation research subfield which deals with multiple criteria in a decision making environment (Figueira et al. 2005). In many circumstances there are multiple conflicting criteria that should be taken into account in order to make a decision. Implicitly, most of us weigh multiple criteria for our decision making or make decisions by using our intuition (Winston and Goldberg 2004). However, when we have to make decisions in complex environments, it is important to be able to structure our decision making and explicitly evaluate multiple criteria (Figueira et al. 2005). Decisions, such as the location of a nuclear factory or the establishment of a new business process and its underlying IT, involve a lot of criteria of a different nature. Furthermore, many stakeholders are affected by the consequences of those decisions. Using MCDA theories for structuring and analyzing the decision making problem leads to better and more informed decisions (Hansson 1996).

### 3.2.2 Intertwining between problem and solution space

The Anamnesis conceptual framework covers two main aspects of rationalization. On the one hand it captures the formulation of the design problem by means of the requirements and their interrelationships. In this way, it provides analysis in the problem space of the design. On the other hand, it captures how the decision problem was addressed by means of design decisions and their justification based on various decision making strategies. Hence, it provides analysis in the solution space. In order to make this distinction explicit our conceptual framework comprises two parts, the problem space and the solution space. The idea of using different spaces for representing different types of information is based on the ISO/IEC/IEEE 42010 standard for Architectural descriptions in Systems and Software Engineering (IEEE 2011). Figure 3.1 presents the generic Anamnesis conceptual framework separated in these two spaces.

The problem and solution spaces can be seen as independent areas where we can focus either on the analysis of the problem or the solution. At the same time, a very important aspect is the intertwining between the problem and solution space. The general relationship between the two spaces has been already investigated by the research community. An important finding is that in domains with increased agility and continuous change, it is very difficult to separate and examine separately the problem and solution spaces (Nuseibeh 2001). The two spaces should rather be analyzed simultaneously since each space influences and enriches the other. In order to address that, Nuseibeh (Nuseibeh 2001) uses a life cycle model to represent this interaction between the problem and solution spaces. Figure 3.2 presents the idea of the life cycle model. The model emphasizes the interaction of the two spaces, but it also considers problem and solution spaces as individual entities. This is also in line with the concept of a 'decision loop' presented in (de Boer et al. 2007).

In line with (Nuseibeh 2001, de Boer et al. 2007) we bridge the two spaces. By doing so, we can make explicit how the requirements are addressed by specific design decisions and which requirements were motivated by design decisions or by possible unanticipated observed impacts of design decisions. Moreover, we can capture the role of requirements in the decision making process. Existing approaches for design rationalization also provide distinction between problem and solution spaces and capture requirements (Tang et al. 2011, Tyree and Akerman 2005). However, they do not provide insight into the interaction between the two spaces.

The Anamnesis conceptual framework considers requirements as a concept with a double role in design rationalization. As we will see, requirements are classified as functional and non-functional. Functional requirements are used for the problem formulation in the

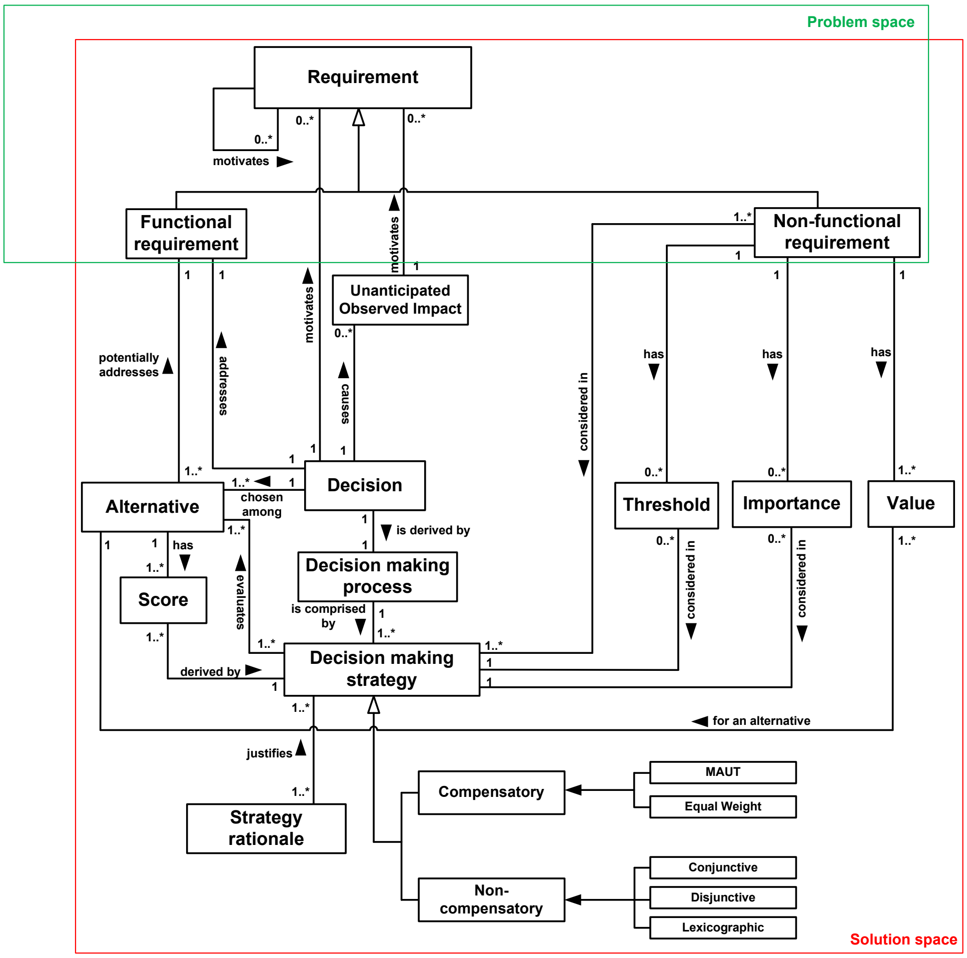


Figure 3.1: The Anamnesis conceptual framework

problem space and in parallel they are treated as design target that have to be achieved by design decision in the solution space. On the other hand, non-functional requirements represent the specific qualities that should be satisfied by the design. In the solution space non-functional requirements are treated as evaluation criteria. This dual role is represented by the overlapping area between the problem and solution spaces in our conceptual framework.

In the next sections we present in detail the problem and solution spaces as covered by the framework.

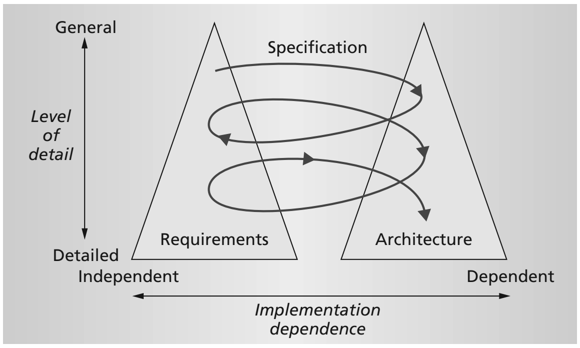


Figure 3.2: The Twin Peaks model (Nuseibeh 2001)

### 3.2.3 Problem space

Figure 3.3 presents our problem space viewpoint. This viewpoint captures functional and non-functional requirements and how a design problem is formulated. Below we provide the detailed description of the concepts and relationships of our viewpoint. We accompany the description with a decision making example for the acquisition of an application system. In this example, we present a company which wants to introduce a new online service. This online service has to be supported by an IT application.

#### 3.2.3.1 Concepts

**Requirement**
A requirement is defined as a statement of need, condition or capability that should be met by the design (Pohl 2010, The Open Group 2012). Requirements can range from high level concerns to lower level concerns which deal with design specificities. In our framework, requirements are the means for the system to achieve its goal. Moreover, new general requirements can be triggered by other specific requirements (see 'motivates' relationship below). Furthermore, since the problem and solution spaces are intertwined, new requirements can be triggered by design decisions or by the unanticipated impacts that have been observed after the execution of design decisions. This is expressed by the 'motivates' relationship among the concepts 'Decision', 'Unanticipated observed impact' and 'Requirement' as shown in Figure 3.1.

In line with requirements engineering literature (van Lamsweerde 2001, Loucopoulos and Karakostas 1995), requirements are classified in two different types, according to the

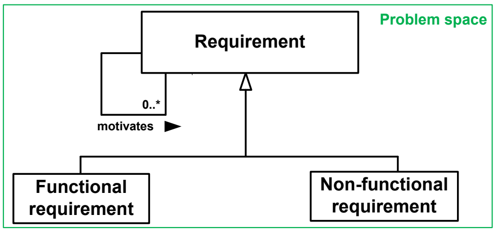


Figure 3.3: Problem space viewpoint

concerns that they address, functional and non-functional:

- *Functional requirement*

  A functional requirement describes the functionality or services or technical details that define *what* the system should do (Loucopoulos and Karakostas 1995, Lapouchnian 2005). In our framework, functional requirements are used to capture the formulation of the design problem. Moreover, functional requirements are used for analysis in the solution space. More specifically, we can have an overview of the solution alternatives that were considered because of a specific functional requirement and which decision was finally taken in order to address the requirement. As such, functional requirements provide insight into the reasons that triggered the execution of new design decisions.

  *Example: Provide web application to support the online service*

- *Non-functional requirement:*

  Non-functional requirements describe *how* the system should provide a specific functionality rather than the functionality itself, which is expressed by functional requirements. According to system engineering literature, a non-functional requirement is a requirement that specifies the criteria or qualities that can be used to judge the operation of a system, rather than specific functionalities (van Lamsweerde 2001, Loucopoulos and Karakostas 1995). In the problem space, non-functional requirements capture the qualities or criteria that have been defined by stakeholders for the optimization of the design. These can range from high level normative properties to low level system specific properties. Additionally, non-functional requirements are used in the solution space as criteria to evaluate and compare design alternatives. This is shown in Figure 3.4 where non-functional requirements are considered in a decision making strategy.

*Example: Four non-functional requirements are considered for the web application selection problem: 'uptime percentage', 'usability', 'cost' and 'scalability'.*

#### 3.2.3.2 Relationships

**Motivates**

The 'motivates' relationship makes explicit the origin of requirements. Sources of origin of requirements can be high level requirements which motivate requirements that deal with specificities at the design level. As such, the design problem can be decomposed starting from abstract requirements to more concrete ones. Moreover, as we will see in the description of the solution space concepts, design decisions and their possible unanticipated observed impacts can be other sources of origin that motivate the elicitation of new requirements. Through the 'motivates' relationship we can formulate the design problem by means of requirements and we also facilitate the bridging between the problem space and solution space.

*Example: The requirement 'introduce online service' motivates the requirement 'provide web application'.*

### 3.2.4 Solution space

The solution space viewpoint enables us to capture the various design choices by means of design decisions as well as their rationalization and to create independent representations of design decision processes (Panchal et al. 2009). By doing so, we can analyze the quality of the design decision making. As we will see below, we capture design rationale by using the concept of decision making process which summarizes the decision making strategy(ies) based on MCDA that were used for the selection of a specific decision. Moreover, design decisions can play the role of a bridge between the analysis of the design and its rationalization (Dutoit et al. 2006).

Existing design rationalization approaches capture various details of design decisions such as rejected alternatives, criteria, rationales, etcetera, by means of decision tables or models (Tyree and Akerman 2005, Tang et al. 2007). However, they capture neither the level of importance of the non-functional requirements (evaluation criteria), nor how the designer considered and balanced amongst the non-functional requirements. More particularly, they do not capture the *decision making strategy* that was used to evaluate the alternatives. For example, a decision could have been made under time pressure, and as such, a heuristic decision strategy may have been used instead of considering the respective importance of all non-functional requirements.

Some approaches allow to informally capture such a justification in free text format, where designers can write a justification for their decision. We argue that this kind on informal justification leads to (1) increased capturing effort which usually demotivates practitioners from using such an approach, and (2) difficulties in providing computer based support for the analysis of the trade-offs among the evaluation non-functional requirements. We argue, that MCDA models are relative simple and subsequently can be easily adopted by practitioners, but on the same time they provide the necessary mathematical precision which makes them suitable for computer-based decision analysis.

Our solution space viewpoint enables the stakeholders (1) to a-priori structure the decision making process in an analytic way where non-functional requirements are used as evaluation criteria and their relative importance are taken into account, (2) to review the decision making process and understand how the predecessor decision maker made a decision, which decision making strategy was used, and the rationale for selecting that strategy, (3) to create a repository of structured rationalization information with which we can retrospectively check the compliance of decisions with the given requirements, and (4) to compare ex-post the captured decision making process with the observed outcome of a decision.

Figure 3.4 presents the concepts and relationships of the solution space viewpoint. Below we also provide their description.

#### 3.2.4.1 Concepts

**Alternative**
By the notion of alternative, we designate what constitutes the object of the decision, or what decision making is directed towards (Figueira et al. 2005). The concept of alternative applies only when one distinct alternative can be put into operation and all the rest of the alternatives are excluded. This mutual exclusion facilitates the object of the decision or the goal of decision making.

Expressed mathematically, $A$ denotes the set of alternatives under consideration at a certain stage of a decision making process. The size of the set is not fixed and it can evolve throughout the decision making process. The evolution may come from changes in the environment during the decision process or the revelation of new aspects of the decision problem, which may change what is feasible or not.

We denote by $a$ an alternative. In the case of finite number of alternatives ($|A| = m$), we

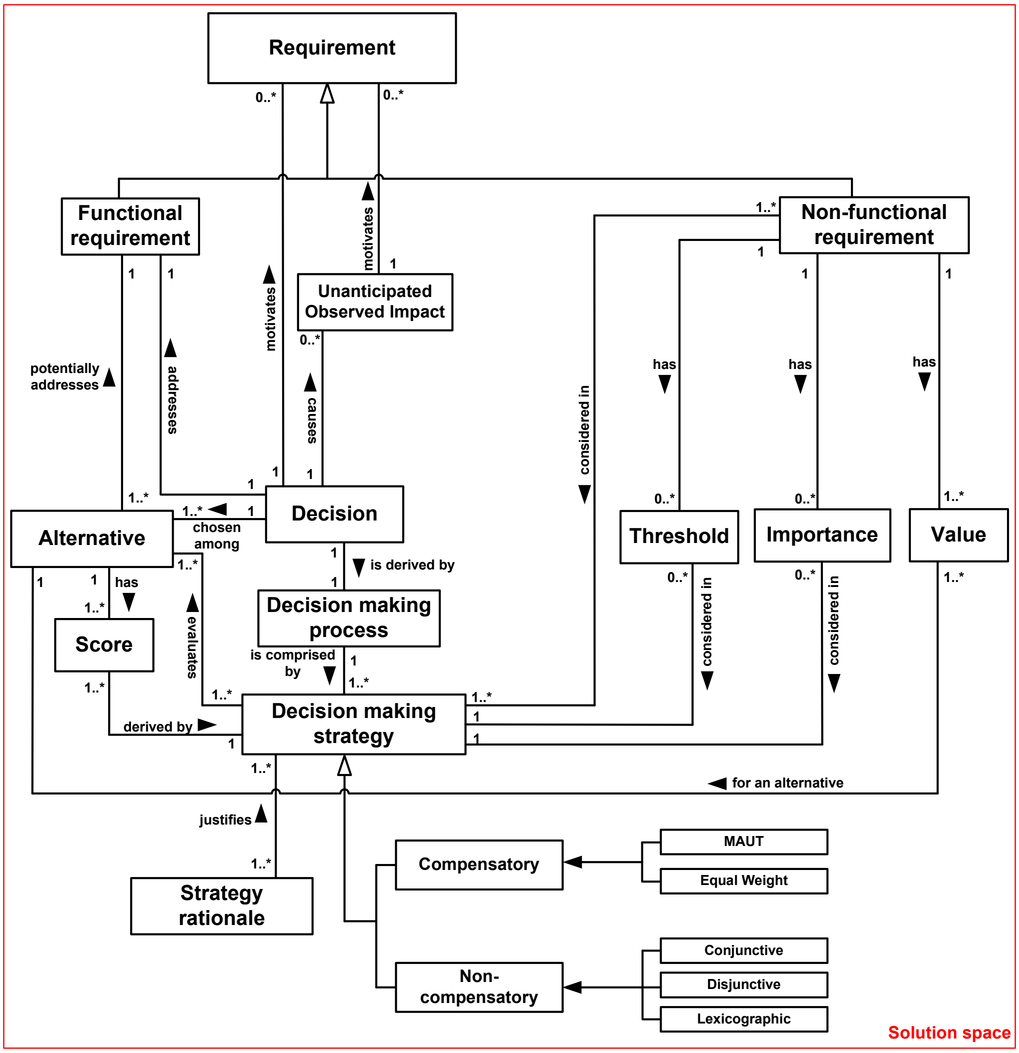


Figure 3.4: Solution space viewpoint

let:

$$A = \{a_1, ..., a_m\} \tag{3.1}$$

*Example: The available alternatives for achieving the requirement 'provide web application' are 'acquisition of web application 1', 'acquisition of web application 2', 'acquisition of web application 3'.*

**Decision:**
Based on the description of the ‘Alternative’ concept, a decision denotes which of the available alternatives was chosen to put into operation. The rest of the alternatives are excluded.

*Example: ‘acquisition of web application 3’ is selected.*

**Decision Making Strategy:**
The decision making strategy concept captures how the alternatives were evaluated during the decision making process. A decision making problem can be addressed in various ways depending on how exhaustively we want to analyze the problem and possible constraints that we have to take into consideration.

Depending on the decision making context, the decision maker uses different strategies to address the decision making problem. Based on MCDA, our conceptual framework covers the two main categories of decision making strategies: compensatory and non-compensatory (Einhorn 1970, Payne 1976, Svenson 1979, Rothrock and Yin 2008). Below we provide the description of these two types of decision making strategies and the concepts which are necessary for the evaluation of alternatives.

**Compensatory:** Compensatory decision making strategies evaluate the alternatives exhaustively, taking *all* non-functional requirements and their trade-offs into consideration. Non-functional requirements with high values compensate for non-functional requirements with lower values. Finally, the alternative with the highest score is selected.

Compensatory strategies aim to provide the best possible decision outcomes given the decision data at hand. However, compensatory strategies require full information on how alternatives score on all non-functional requirements, and they are time consuming (Einhorn 1970).

Below we provide the compensatory decision making strategies covered by our conceptual framework:

**Multi-Attribute Utility Theory:** The *Multi-Attribute Utility (MAUT)* (Figueira et al. 2005), also called *Weighted Additive (WADD)* rule, is a theory primarily based on the concepts of expected utility. The preference of the decision maker is calculated by Equation 3.2. In this equation $w_i$ is the importance factor of the $i$th non-functional requirement $x_i$ (captured by the concept ‘importance’) and $u_i$ is the value of the $i$th non-functional requirement (captured by the concept 'value'). There are two main assumptions when we use this method: 1) the values for each non-functional requirement are independent, and 2) the importance factors for each non-functional requirement can be defined independently by the importance factors of other non-functional requirements.

$$U(x_1, ..., x_n) = \sum_{i=1}^{n} w_i u_i \tag{3.2}$$

This technique gives the decision maker the ability to assign different weights to non-functional requirements in line with their importance. The alternative with the highest utility score is chosen by the decision maker (Rothrock and Yin 2008).

**Equal Weight:** The *Equal Weight* or *Simple Additive Rule* is a simplified linear compensatory model, where the non-functional requirements are not weighted and therefore the importance factor does not play a role in the calculation of the score. Equation 3.3 calculates the preference of the decision maker (Rothrock and Yin 2008).

$$U(x_1, ..., x_n) = \sum_{i=1}^{n} u_i \tag{3.3}$$

*Example (MAUT strategy): Based on a compensatory strategy three non-functional requirements were considered: 'uptime percentage' is of high importance and 'usability' and 'scalability' are of less. By doing a trade-off analysis among the non-functional requirements the web application with the highest expected utility was selected.*

**Non-compensatory:** Non-compensatory strategies are consistent with the concept of bounded rationality (Einhorn 1970). This means that the decision making process is limited by factors such as hard constraints, time constraints and the cognitive load of the decision maker. As such, non-compensatory strategies evaluate alternatives heuristically by conducting a limited trade-off analysis among the non-functional requirements. The main characteristics of non-compensatory strategies are twofold: (1) they can reduce the decision making effort, (2) they are not demanding regarding the information needed to make the decision (Payne et al. 1993).

Below we provide the non-compensatory decision making strategies covered by our conceptual framework:

**Conjunctive:** In conjunctive strategies the decision maker defines a minimum threshold (cut-off value) for every non-functional requirement (captured by the concept ‘threshold’). Alternatives that have one or more non-functional requirements values (captured by the concept ‘value’) lower than the defined non-functional requirement threshold are eliminated from the choice set (Hwang and Yoon 2012, Rothrock and Yin 2008). An alternative is classified as acceptable if:

$$\forall_{1\leq j\leq n}\left[x_j \geq x_j^0\right] \tag{3.4}$$

where $x_j^0$ is the threshold of the $j$-th non-functional requirement $x_j$. The role of the threshold values is very crucial and these values should be defined carefully by the decision maker. When it is too high every alternative will be eliminated and when it is too low the filtering of alternative is not adequate (Hwang and Yoon 2012).

Conjunctive strategies can be used with maximum thresholds as well. In this case, alternatives that have one or more non-functional requirements values higher than the defined threshold are excluded from the choice set (Van Raaij et al. 2013). In this case, an alternative is classified as acceptable if:

$$\forall_{1\leq j\leq n}\left[x_j \leq x_j^0\right] \tag{3.5}$$

**Disjunctive:** In disjunctive strategies the decision maker defines desired threshold values for every non-functional requirement. Each alternative with at least one non-functional requirement value (captured by the concept ‘value’) at or above the minimum threshold level (captured by the concept ‘threshold’) is classified as acceptable alternative (Hwang and Yoon 2012). In the case of disjunctive strategies the threshold values should be usually set at higher levels than in conjunctive strategies in order to have adequate filtering of the alternatives (Kozielecki 1982). An alternative is classified as acceptable alternative if:

$$\exists_{1\leq j\leq n}\left[x_j \geq x_j^0\right] \tag{3.6}$$

where $x_j^0$ is the desirable minimum threshold of the $j$th non-functional requirement $x_j$.

Disjunctive strategies can be used with maximum thresholds as well (Van Raaij et al. 2013). In this case an alternative is classified as acceptable if it has at lease one NFR with a value lower than the maximum threshold value:

$$\exists_{1\leq j\leq n}\left[x_j \leq x_j^0\right] \tag{3.7}$$

**Lexicographic:** In this strategy the decision maker defines a non-functional requirement as the most important for his decision making and evaluates the alternatives based on that. This means that the alternative with the best value on this non-functional requirement is chosen among the available choice set. If more than one alternative tie for that specific non-functional requirement, then the decision maker evaluates the subset of alternatives based on the second most important non-functional requirement. This process is iteratively conducted until a single alternative is chosen or until every available criterion has been considered. The ordering of importance of non-functional requirements is captured by the concept 'importance'.

Lexicographic method defines advanced trade-off analysis mechanisms for the comparison of alternatives. Due to its simplicity, it is a method widely used when we have limited resources for the processing of the decision making problem. However, it is not appropriate in the case where we want to exhaustively evaluate, since it utilizes a small part of the available information of the decision problem.

*Example (conjunctive strategy): A conjunctive strategy is used for the evaluation. In this case the decision is based on a budget constraint of €15000 for the acquisition of the Web application. Regardless of the non-functional requirements 'uptime percentage', 'usability' and 'scalability', web application 1 is excluded from the choice set based on the non-functional requirement 'cost'.*

**Evaluation concepts:**

The evaluation concepts capture the role of non-functional requirements during the decision making process. This is mainly done by assigning numerical values to evaluate the importance and performance of each non-functional requirements given a certain decision making strategy. In case that a numerical evaluation is not feasible, we also propose below a scheme which allows the decision maker to use natural or artificial language for the evaluation of non-functional requirements.

- **Value:**
  The 'value' concept captures the performance of each alternative of the choice set based on a specific non-functional requirement.

  *Example: The value of the alternative 'web application 3' in regard with the non-functional requirement 'uptime percentage' is 9 out of 10, whereas of 'web application 2' is 7 out of 10.*

- **Importance:**
  The ‘importance’ concept captures the subjective judgment of the decision maker about the relative importance of an evaluation non-functional requirement. As we will see below, in the description of the decision making strategies, the notion of importance of non-functional requirements is not used by every decision making strategy. In our framework it is used for MAUT strategy and it also used to capture the order of importance of non-functional requirements in Lexicographic strategy.

  *Example: The non-functional requirement ‘uptime percentage’ is of high importance and it is rated with 9 out of 10, whereas ‘usability’ has a lower importance with a rate 6 out of 10.*

- **Threshold:**
  The ‘threshold’ concept captures the threshold (cut-off) level of a non-functional requirement. It is only used in the case of conjunctive, disjunctive non-compensatory strategies.

  *Example: The non-functional requirement ‘cost’ has a threshold value of €15000.*

- **Score:**
  The ‘score’ concept captures the overall performance of the alternative after the evaluation with a certain decision making strategy. Depending on the decision making strategy that was used, scores are calculated with different ways.

  *Example: After the evaluation the score for ‘web application 1’ is ‘90 out of 100’, for ‘web application 2’ is ‘91 out of 100’ and ‘web application 3’ is ‘95 out of 100’.*

- **Handling non-functional requirements with linguistic variables:**
  Using numerical values for the evaluation enables us to capture in a precise way how the decision maker evaluated the alternatives given a set of non-functional requirements. However, in certain cases this *mechanistic* way of dealing only with numerical data leads to the *principle of incompatibility* (Zadeh 1975), where high precision is incompatible with the great complexity of human thought. Given this fact, it is suggested that sometimes it may be necessary to abandon the high standards of precision as it is defined in mathematics and be more tolerant close to a *humanistic* perspective (Zadeh 1975). In such a context, linguistic variables can be used to reduce the gap between mathematics precision and the complexity of human thought. Linguistic variables are variables the values of which are not numbers but

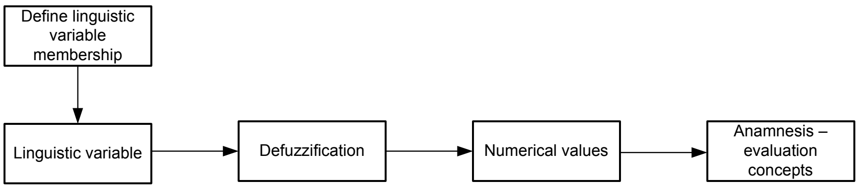

Define linguistic variable membership
Linguistic variable
Defuzzification
Numerical values
Anamnesis – evaluation concepts

Figure 3.5: Defuzzification scheme

words or sentences in a natural of artificial language. For example, someone can express their perception about the weather temperature by using the words 'cool', 'moderate' and 'hot'.

As a first attempt to deal with this issue, we use a simple defuzzification scheme, which is shown is Figure 3.5.

The scheme uses as an input linguistic variables for the evaluation of non-functional requirements and produces through defuzzification numerical values that subsequently can be used by the various decision making strategies of our conceptual framework. To do so, we use trapezoidal fuzzy numbers, one of the most used types for the representation of the membership of linguistic variables (Banks 2008, Isabels and Uthra 2012) and the mean-max membership (middle-of-maxima) method for fuzzy to crisp conversion (Isabels and Uthra 2012).

As an example consider the linguistic variable 'temperature' for the perception of temperature and more specifically a 'moderate' temperature. We first define the membership of moderate temperature through the trapezoidal fuzzy number (15, 20, 25, 30) as shown in Figure 3.6. Thereafter, we calculate the crisp numerical value for moderate temperature *temp* having as an input the trapezoidal fuzzy

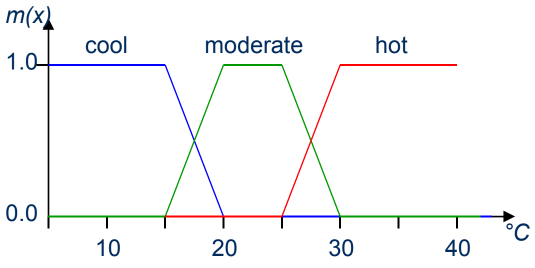


Figure 3.6: Linguistic variable 'temperature'

number (15, 20, 25, 30) as follows:

$$temp = (a + b)/2 = (20 + 25)/2 = 22,5$$

This relative simple scheme allows us to use MCDA decision strategies and in parallel to take advantage of linguistic variables for the evaluation of non-functional requirements. For example, decision makers can use numerical values to evaluate the majority of non-functional requirements and linguistic variables where the evaluation by numerical values of non-functional requirements is not feasible.

**Decision making process:**
The decision making process concept provides an overview of the decision making in terms of the decision making strategies that were used for a specific decision. As it is also stated in our conceptual framework, a decision making process can comprise one or more decision making strategies. Sometimes it is not satisfactory to address a decision making problem with a single decision making strategy (Rothrock and Yin 2008, Elrod et al. 2004, Jeffreys 2004).

As an example consider a case where the decision makers have to take into account constraints such as budget or time during the evaluation process. In such a case they may start the evaluation with a non-compensatory strategy where they exclude quickly alternatives that do not comply with the given constraints. Thereafter, they can evaluate exhaustively the rest with a compensatory strategy.

In the case of multiple decision making strategies, our conceptual framework captures per used strategy the scores of alternatives as well as the rates threshold, importance and value of the evaluation non-functional requirements.

*Example: The decision maker uses a conjunctive strategy to exclude alternatives with a price higher than €15000 and then evaluates the remaining ones with a MAUT strategy based on the non-functional requirements 'uptime percentage', 'usability' and 'scalability'.*

**Strategy rationale:**
In the context of a decision making process, decision makers not only have to choose amongst alternatives (actual decision making process), but also have to select the decision strategy that satisfies their current evaluation needs. Typical reasons for the adoption of different strategies are constraints such as budget or time. The capturing of this information justifies the decision strategy that was selected for the evaluation process. As stated in the conceptual framework, one strategy rationale can justify one or more decision making strategies.

*Example: The decision maker uses a conjunctive decision making strategy because of the budget threshold of €15000.*

**Unanticipated observed impact:**
This concept captures an unanticipated consequence of an already made decision. The outcome of a decision can be observed during the ex-post analysis of the design (Baron and Hershey 1988). Some of the consequences of decisions are revealed during the implementation phase or during the maintenance of the existing design.

For us the main usefulness of capturing unanticipated observed impacts is that they can be used in future design iterations in order to avoid decisions with negative consequences. In the case of a negative/unwanted consequence the 'unanticipated observed impact' concept is used as an intertwining concept, since it motivates the elicitation of new requirements in the problem space, which should be addressed by new design decisions back in the solution space, in order to resolve the problematic situation.

*Example: 'acquisition of web application 3' led to difficulties in the integration with other systems of the company.*

#### 3.2.4.2 Relationships

**Potentially addresses:**

The 'potentially addresses' relationship captures which alternatives were considered as candidates for the satisfaction of a specific functional requirement.

*Example: The alternative 'acquisition of web application 1' 'potentially addresses' the functional requirement 'provide web application'.*

**Addresses:**

The 'addresses' relationship captures which decision addresses the given functional requirement.

*Example: The decision 'acquisition of web application 3' 'addresses' the functional requirement 'provide web application'.*

**Chosen among:**

The 'chosen among' relationship captures which alternative was selected among the choice set to be put into operation as a decision.

*Example: The decision 'acquisition of web application 3' was 'chosen among' the alternatives set 'acquisition of web application 1', 'acquisition of web application 2' and 'acquisition of web application 3'.*

**Evaluates:**

The 'evaluates' relationship denotes which decision making strategy was used for the evaluation of the alternatives.

*Example: a 'conjunctive' decision making strategy 'evaluates' the alternatives 'acquisition of web application 1', 'acquisition of web application 2' and 'acquisition of web application 3'.*

**Derived by:**

The relationship 'derived by' relates the score of an alternative with the decision making strategy that was used its the evaluation.

*Example: The score of the alternative 'acquisition of web application 1' 'derived by' a 'conjunctive' decision making strategy.*

**Is comprised by:**

The relationship 'is comprised by' is used to capture the decision making strategy or strategies that constitute the decision making process for a design decision.

*Example: The decision making process for design decision 'acquisition of web application 3' 'is comprised by' a 'conjunctive' and a 'MAUT' decision making strategy.*

**Is considered in:**

The relationship 'is considered in' connects the non-functional requirements and evaluation concepts (threshold, importance, value) with the decision making strategy that was used for this evaluation.

*Example: The non-functional requirement 'uptime percentage' 'is considered' in a 'MAUT' decision making strategy.*

## 3.3 Conclusion

In this chapter we introduced our generic conceptual framework for design rationalization. More specifically, we answered *RQ2* by providing a conceptualization of the reasoning behind design decisions using MCDA techniques. Furthermore, we have addressed *RQ3* by taking into account the role of requirements for the formulation of the design problem and for the rationalization of the design decisions.

# CHAPTER 4

# A conceptual framework for EA design rationalization

In this chapter we present EA Anamnesis, a specialization of our generic conceptual framework for design rationalization for the domain of EA. EA Anamnesis captures the role of goals and principles, concepts which are widely used in EA practice, for the formulation of the design problem. It structures design rationale according to the enterprise perspective (Section 2.1.2) and makes explicit their cross-perspective interrelationships. For example, EA Anamnesis reveals the relationship between a business design decision and an IT requirement. Moreover, the conceptual framework provides traceability from an EA design to its design decisions and vice versa.

The content of this chapter is based on work published as:

Georgios Plataniotis, Sybren de Kinderen and Henderik A. Proper: EA Anamnesis: Towards an Approach for EA Rationalization. In: *Proceedings of the 2012 Workshop on Domain-specific Modeling*, pp. 27-32, ACM, 2012

Georgios Plataniotis, Sybren de Kinderen and Henderik A. Proper: Capturing Decision Making Strategies in EA – A Viewpoint. In *Proceedings of EMMSAD 2013 (Exploring Modelling Methods for Systems Analysis and Design)*, pp. 339-353, LNBIP, 2013

Georgios Plataniotis, Sybren de Kinderen and Henderik A. Proper: Relating Decisions in EA Using Decision Design Graphs In *Proceedings of EDOC 2013 (Enterprise Distributed Object Computing)*, pp. 139-146, IEEE, 2013

Georgios Plataniotis, Sybren de Kinderen and Henderik A. Proper: EA Anamnesis: An Approach for Decision Making Analysis in EA. In: *International Journal of Information Systems Modeling and Design (IJISMD)*, vol. 5, no. 3, pp. 75-95, IGI Global, 2014

Georgios Plataniotis, Sybren de Kinderen, Qin Ma and Henderik A. Proper: A Conceptual Model for Compliance Checking Support of EA Decisions. In: *Proceedings of 17th Conference on Business Informatics (CBI)*, vol. 1, no. 3, pp. 191-198, IEEE, 2015

## 4.1 Introduction

In Chapter 3 we presented Anamnesis, a conceptual framework for design rationalization. We have seen how the framework captures rationalization information and how it provides insight both on the problem and solution space of the design. We have also seen how the framework bridges the two spaces by making explicit how the requirements affect the decision making and how the design decisions and their possible unanticipated impacts motivate the elicitation of new requirements.

However, as discussed in Chapter 2, rationalizing an EA imposes some additional challenges due to its cross-domain and socio-technical nature. The decision making in EA involves the consideration and management of requirements originating from different domains and being of different nature, ranging from high level goals, to business and IT requirements. During the decision making process, enterprise architects should take into account those differing and in some cases contradictory requirements and provide solutions that satisfy them. In addition to that, as discussed in Section 2.1.3, their design decisions have to be compliant with the given EA principles.

In this chapter we address the identified objectives of the domain of EA (Section 2.4) and we provide a specialization of our Anamnesis conceptual framework, which we will refer to as 'EA Anamnesis'. The framework captures the formulation of the design problem by means of goals, principles and requirements and it categorizes and interrelates design rationale depending on its perspective. Furthermore, it enables us to trace design rationale behind the design and vice versa.

An important clarification for the positioning of our work in the domain of EA, is that EA Anamnesis focuses on the rationalization (based on various decision making strategies) of individual design decisions and have an immediate impact on the implementation of the architecture. This differentiates our framework from approaches that aim to rationalize the execution of the design process (design steps) across the enterprise.

Examples of such type of approaches are the TOGAF Architecture Development Method (ADM) (The Open Group 2011) and the Integrated Architecture Framework (IAF) (van't Wout et al. 2010). These approaches provide roadmaps for the execution of the decision making and specify flows of decision making which range from strategic level decisions to EA design decisions. Furthermore, they provide decision support regarding the change management of the architecture. Another example is the Design & Engineering Methodology for Organizations (DEMO) (Dietz 2006) which distinguishes between decisions that are implementation independent and implementation dependent. By implementation independent, we mean decisions which define the network of transactions of the enterprise and actor roles independent of any implementation. EA Anamnesis does not deal with the rationalization of implementation independent design decisions. However, it can complement DEMO in the sense that it can assist stakeholders with the rationalization of their implementation dependent design decisions.

The chapter is organized as follows. Section 4.2 presents the EA Anamnesis conceptual framework in detail, comprising both problem (Section 4.2.1) and solution space (Section 4.2.2). Section 4.3 illustrates how the concepts of our framework are used for capturing design rationale in EA. Finally, Section 4.4 concludes this chapter.

## 4.2 The EA Anamnesis conceptual framework

Figure 4.1 presents the EA Amamnesis conceptual framework. In line with EA literature and practice (Greefhorst and Proper 2011, Lankhorst 2013) we have taken into account the role of high level goals and principles for the elicitation of requirements and subsequently for the steering of the decision making.

An important issue, while we consider goals and principles in our analysis, is the relativity between the problem and solution space. By relativity we mean that something about what constitutes a problem space and a solution space is not an absolute given, but rather dependent to the situation at hand. In line with requirements engineering (Lapouchnian 2005) and the motivation extension of ArchiMate (The Open Group 2012), principles and requirements are concepts which further refine the high level needs described by goals. When the problem is refined enough through requirements then it is solved by specific design decisions. As such, we consider design decisions which actually address those refined needs as part of the solution space. Subsequently, goals and principles which are used for the problem formulation, are part of the problem space.

In the next sections we present in detail the problem and solution space respectively of the EA Anamnesis conceptual framework.

### 4.2.1 Problem space

The upper part of Figure 4.1 presents the problem space viewpoint. The viewpoint enhances the provision of rationalization in the problem space for the EA domain by taking into account the role of goals and principles, terms which are widely used in the practice of EA (Greefhorst et al. 2013, The Open Group 2012). Moreover it uses the notion of EA perspective for the categorization of requirements of various perspectives of the enterprise. By doing so, we can also have an overview of the relationships of design rationale of different perspectives. For example, a business requirement that triggers an IT one.

The development of our problem space viewpoint was done by taking into account the EA practice and by also considering the existing state of the art for the domain of EA. In the case of problem space analysis, we took the motivational extension of ArchiMate which was inspired by the ARMOR language (Quartel et al. 2009). Moreover, we investigated the current practice of the motivational extension of ArchiMate. One of the findings was that the motivation extension of ArchiMate (The Open Group 2012) is comprised by a quite significant number of concepts.

According to (Engelsman and Wieringa 2014; 2012) the large amount of concepts combined with their sometimes ambiguous definition, introduces difficulties in the usability of the motivational extension by EA practitioners. The authors identified that only a limited set of concepts was well understood. Interestingly, the large amount of the ambiguously

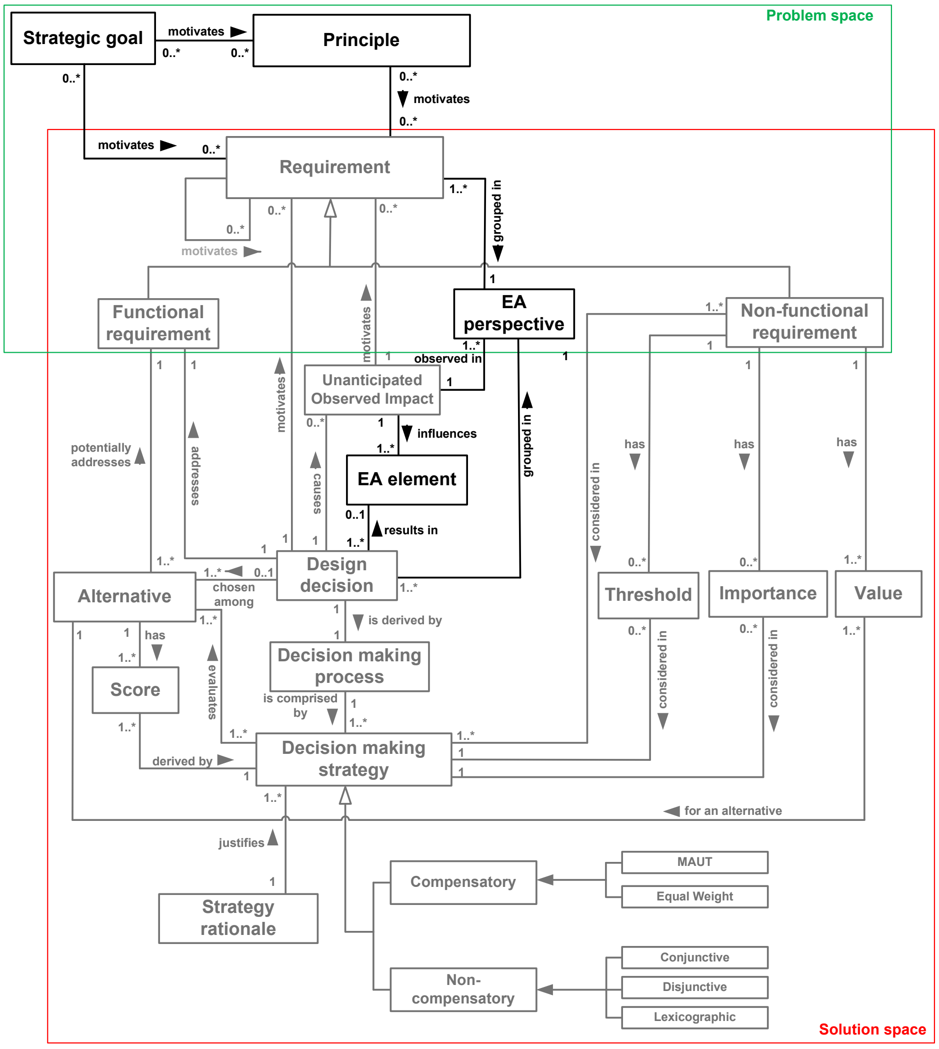


Figure 4.1: The EA Anamnesis conceptual framework

defined concepts also hints that the motivational extension is at odds with one of the key design principles behind the ArchiMate language: 'the language should be as compact is possible' (Lankhorst et al. 2010).

Our viewpoint takes these findings into account, and therefore uses a limited set of concepts. More specifically, we extend our generic Anamnesis problem space viewpoint and we provide as far as necessary together with the concept 'requirement', the concepts ' strategic goal', 'principle' and 'EA perspective'. We use the term 'strategic goal' to signify that we deal with high level design independent needs. By doing so, we aim to reduce the ambiguity that was observed in the aforementioned study between the concepts of goal and requirement. Unfortunately, the concept 'principle' is not investigated in this study, but we use it since it is a widely used term in the domain of EA (Greefhorst and Proper 2011).

Below we provide the description of the concepts which were newly added to the EA Anamnesis conceptual framework. These concepts are highlighted on the problem space part of Figure 4.1. The rest of the concepts (shaded with gray color) have been already described in Section 3.2.3 and presented in the Figure 3.4. We use again the application acquisition example of Section 3.2.3 for the illustration of the concepts.

**Strategic goal:**
According to EA practice and literature, a goal is defined as an end state that a stakeholder intends to achieve (Lankhorst 2013, The Open Group 2012). Based on the identification of the goals of stakeholders, the requirements on the artifact can be derived. In that sense, goals provide the motivation, i.e. the why of the requirements (Greefhorst and Proper 2011). As stated above, to be clear with the distinction between goals and requirements, we use the concept of ' strategic goal' to capture high level concerns which are EA design independent.

*Example: The strategic goal of the company is to 'improve online marketing presence'.*

**Principle:**
A principle is a normative property which guides the way that the system will be realized (Greefhorst and Proper 2011). Principles are closely related to goals and requirements. Similarly to requirements, they define the intended properties of a system. However, a principle is positioned at a higher level of abstraction than requirements (The Open Group 2012).

*Example: The online service should be provided without interruption. Stakeholders defined the principle 'Service availability'.*

**EA perspective:**
As we have seen in Section 2.1.2, EA provides a structured way to view and define the enterprise through various perspectives. Our framework, follows the notion of perspectives in order to categorize design rationales. More specifically, in the problem space we use the concept of 'EA perspective' for the categorization of requirements. As such,

we can have an overview of how the design problem was formulated across the various EA perspectives. Moreover, the categorization of requirements in perspectives combined with their relationships (captured by the 'motivates' relationship) enables us to have an overview of the cross-perspective relationships of requirements in the EA. By doing so, we have insight into how the enterprise architect translates the requirements of a perspective into requirements of another.

*Example: The functional requirement 'provide web application' is part of the ArchiMate perspective 'application layer'.*

### 4.2.2 Solution space

The lower part of Figure 4.1 presents the solution space viewpoint of EA Anamnesis conceptual framework. The viewpoint enhances the generic viewpoint of Section 3.2.4. Similarly with the problem space it incorporates the notion of 'EA perspective' to facilitate the categorization of design decisions. This enables enterprise architects to have a holistic overview of the decision making in the enterprise, since they can see which design decisions were executed, in which perspective. The same categorization applies for the anticipated observed impacts of design decisions. By doing so, the framework not only categorizes the observed impacts, but also captures the cross-perspective consequences of design decisions in the EA. For example, a problematic IT decision for an application user interface can cause usability issues in the execution of a specific business process.

Furthermore, the viewpoint provides traceability between the EA design and its design rationalization and vice versa. This enables practitioners to start their analysis either by inspecting specific EA elements and then zooming into the design rationales that are related with them, or by inspecting design rationales and then identify how the design was impacted by them.

Below we provide the description of the newly added concepts and relationships of the EA solution space viewpoint (highlighted in Figure 4.1). For the existing concepts please refer to Section 3.2.4.

#### 4.2.2.1 Concepts

**EA Element:**

An EA element (similar to concept of an architecture element from Tang et al. (2007)) is either the direct result produced from a set of executed EA design decisions, or a representation of this result. We use this concept to refer to architectural representations.

Specifically, we use it as a bridging concept towards EA modeling languages, like ArchiMate, whereby an EA element allows us to link design decisions with the concepts of those languages.

*Example: The EA Element 'web application' is the direct result produced by the design decision 'acquisition of web application 3'.*

**EA perspective:** As explained in Section 4.2.1 the concept of 'EA perspective' is used for the categorization of requirements. In the solution space, we use the 'EA perspective' concept to categorize design decisions, their possible unanticipated observed impacts and EA elements. As such, we have an overview of the distribution of the decision making and its results across the EA by means of design decisions and elements. Moreover, we use the 'EA perspective concept' to categorize unanticipated observed impacts, separately from the design decisions from which they originate, in order to reveal possible cross-perspective consequences of design decisions in the EA. For example, a problematic IT decision can have an unanticipated observed impact on the business and vice versa.

*Example: The design decision 'acquisition of web application 3' is part of the ArchiMate perspective 'application layer'.*

#### 4.2.2.2 Relationships

**Results in:**

The 'results in' relationship facilitates the linkage between design decisions with EA elements.

*Example: The design decision 'acquisition of web application 3' 'results in' the EA Element 'web application'.*

**Grouped in:**

The 'grouped in' relationship associates design decisions with their corresponding EA perspectives.

*Example: The design decision 'acquisition of web application 3' is 'grouped in' the EA perspective 'Business Layer'.*

**Influences:**

The ‘influences’ relationship allows us to capture the influences of unanticipated observed impacts on various EA elements across the EA design.

*Example: The unanticipated observed impact ‘degraded user experience-problematic user interface’ ‘influences’ the EA Element ‘customer profile registration-business process’.*

## 4.3 Illustration

In this section we introduce an insurance case study, and subsequently use it to illustrate EA Anamnesis conceptual framework. Note that the insurance case study is fictitious, yet realistic. This is because it is based on the running case study used to illustrate the ArchiMate specification, which in turn is based on a real insurance company (The Open Group 2012). In order to illustrate the categorization of design rationales into EA perspectives, we use the ‘Business’, and ‘Application’ layers of ArchiMate.

### 4.3.1 Case study: ArchiSurance

ArchiSurance is an insurance company that sells car insurance products using a direct-to-customer sales model. It does so in order to reduce both its operational as well as its production costs.

Figure 4.2 presents the ArchiSurance direct-to-customer sales model, modeled with ArchiMate. Two business services support the sales model of ArchiSurance: ‘car insurance registration service’ and ‘car insurance service’. ArchiMate helps us to understand the dependencies amongst different perspectives of an enterprise. For example, in Figure 4.2 we see that the business service ‘car insurance registration service’ is realized by a business process ‘register customer profile’. In turn, we also see that this business process is supported by the application service ‘customer administration service’.

Although disintermediation reduces operational costs, it also increases the risk of adverse risk profiles (Cummins and Doherty 2006), namely incomplete or faulty risk profiles of customers. These adverse profiles lead insurance companies to calculate unsuitable premiums or, even worse, to wrongfully issue insurances to customers. As a response, ArchiSurance decided to use intermediaries to sell its insurance products. After all, compiling accurate risk profiles is part of the core business of an intermediary (Cummins and Doherty 2006).

In our scenario, an external enterprise architect called *John*, was hired by ArchiSurance to guide the change to an intermediary sales model.

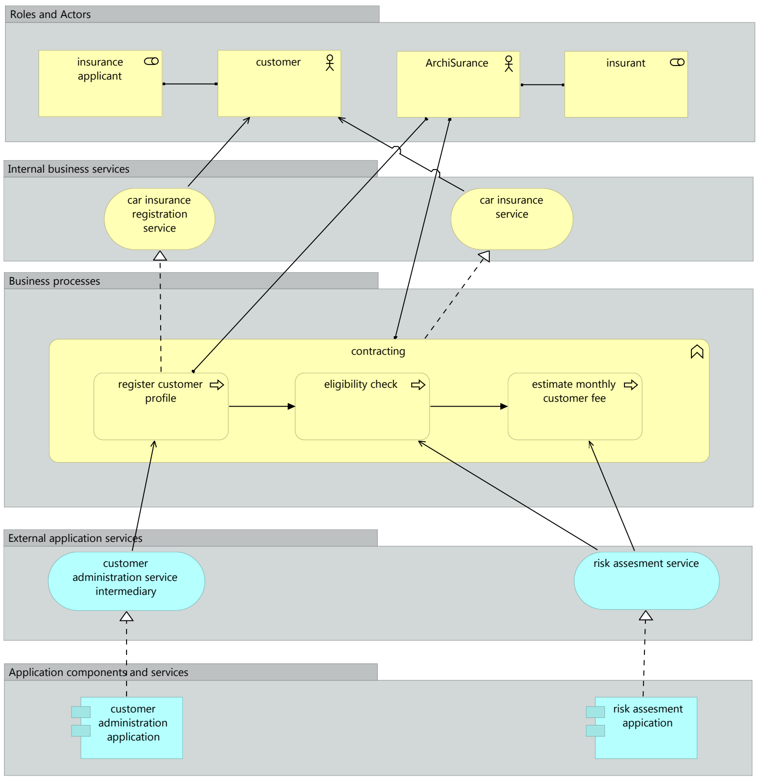

Roles and Actors
insurance applicant
customer
ArchiSurance
insurant
Internal business services
car insurance registration service
car insurance service
Business processes
contracting
register customer profile
eligibility check
estimate monthly customer fee
External application services
customer administration service intermediary
risk assesment service
Application components and services
customer administration application
risk assesment appication

Figure 4.2: ArchiSurance direct-to-customer EA model

John uses ArchiMate to capture the impacts that selling insurance via an intermediary has, in terms of business processes, IT infrastructure and more. For illustration purposes we will focus on the realization of the new business process 'customer profile registration' through EA elements in the application layer. The resulting ArchiMate model is depicted in Figure 4.3. The business collaboration is realized by the collaboration of two IT applications 'customer administration ArchiSurance' and 'customer administration intermediary'.

## 4.3.2 Capturing design rationales with EA Anamnesis

As we can see in Figures 4.2 and 4.3, John captured the EA change from direct-to-customer model to intermediary via ArchiMate. However, these models do not capture the design decisions and the rationale behind the design. To capture design rationale, John relies on

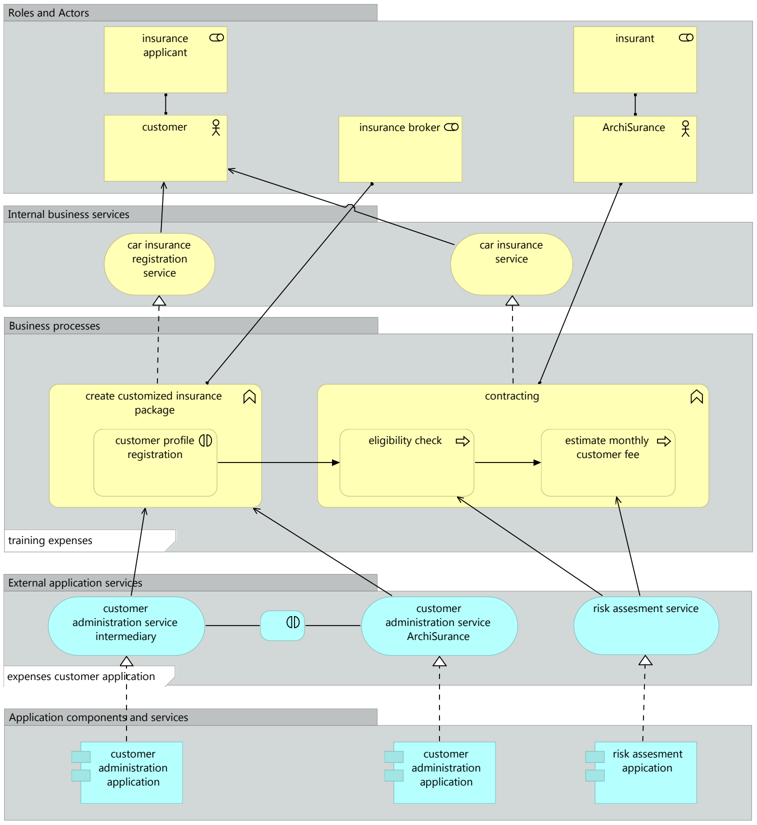

Roles and Actors
insurance applicant
insurant
customer
insurance broker
ArchiSurance
Internal business services
car insurance registration service
car insurance service
Business processes
create customized insurance package
customer profile registration
contracting
eligibility check
estimate monthly customer fee
training expenses
External application services
customer administration service intermediary
customer administration service ArchiSurance
risk assesment service
expenses customer application
Application components and services
customer administration application
customer administration application
risk assesment appication

Figure 4.3: ArchiSurance intermediary EA model

the EA Anamnesis approach.

## Design rationalization graph

Figure 4.4 provides the design rationalization graph of the ArchiSurance enterprise transformation. As we mentioned in Section 1.4.2, we use this graph as a concrete syntax which enables us to illustrate and evaluate EA Anamnesis. Here, we can see how the design problem is formulated by means of goals and requirements and how it is addressed by specific design decisions. The strategic goal ‘reduce the risk of adverse risk profiles’ is refined through the ‘motivates’ relationship to the functional requirement FR01 ‘establish customer registration’ in the business layer of the enterprise. FR01 was addressed by design decision D01 ‘create customer profile registration business process’. D01 led to the

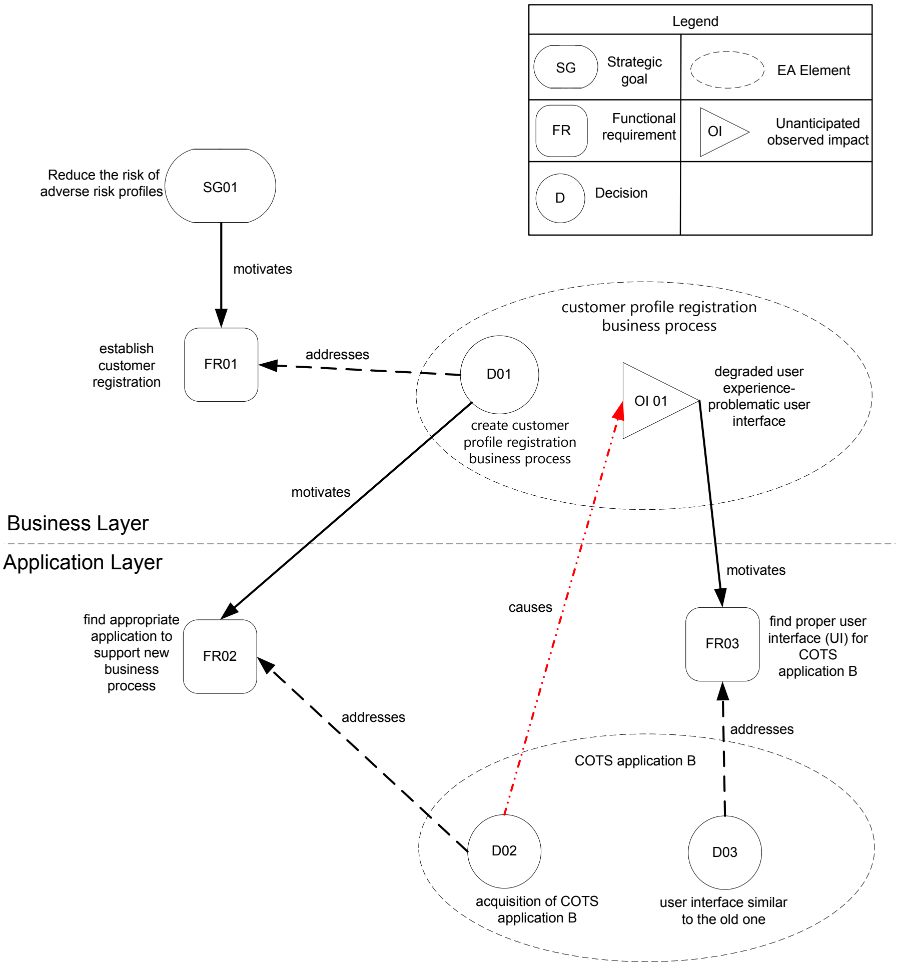

Legend
SG
Strategic goal
EA Element
FR
Functional requirement
OI
Unanticipated observed impact
D
Decision
Reduce the risk of adverse risk profiles
SG01
motivates
establish customer registration
FR01
addresses
customer profile registration business process
D01
create customer profile registration business process
OI 01
degraded user experience-problematic user interface
motivates
Business Layer
Application Layer
motivates
causes
find appropriate application to support new business process
FR02
addresses
FR03
find proper user interface (UI) for COTS application B
addresses
COTS application B
D02
acquisition of COTS application B
D03
user interface similar to the old one

Figure 4.4: ArchiSurance design rationalization graph

creation of the EA element 'customer profile registration business process'. Subsequently, the business layer decision D01 motivated the elicitation of FR02 'find appropriate application to support new business process' in the application layer of the enterprise. This is an example of a cross-perspective relationship between a decision and a requirement in the EA. FR02 was addressed by design decision D02 'acquisition of COTS application B', and led to the creation of EA element 'COTS application B'. Despite the fact that the operational and technical characteristics of COTS application B were adequate, John identified and captured the unanticipated observed impact OI01 'degraded user experience-problematic user interface'. The responsible users for the execution of the business process were not familiar with the newly introduced user interface. This resulted in problematic data entries and limited productivity. This is an example of how EA Anamnesis captures cross-perspective observed unanticipated impacts. In order to deal with the problematic situation, John elicited the functional requirement FR03 'find proper user interface (UI) for COTS application B'. Subsequently, FR03 was addressed by design decision D03 'user interface similar with the old one'. Users were now able to run the customer profile registration business process smoothly.

**Analysis per design decision:**
Now we illustrate how EA Anamnesis reveals the rationalization details behind specific design decisions. We zoom into the design decision D02 'Acquisition of COTS application B', which is summarized in Table 4.1. Similarly to graphs, we use tables as another mean of concrete syntax of EA Anamnesis which summarize design rationale.

As we discussed, the decision was to address F02 to provide an appropriate application system for the support of the newly introduced intermediary business process. As we can see from Table 4.1, John evaluated a total of four alternatives: the selected one 'acquisition of COTS application B', which is the actual executed decision, and 'acquisition of COTS application A', 'acquisition of COTS application C' and 'upgrade existing application (inhouse)', which are the three rejected alternatives. Four non-functional requirements were considered during the evaluation process: 'usability', 'interoperability', 'scalability' and 'cost'.

Figure 4.5 provides a design decision graph of the decision making process for D02. Here we can see the non-functional requirements that were considered during the evaluation and the role of EA principles for the elicitation of non-functional requirements. John had to steer his decision making by taking into account the EA principle 'IT systems adhere to open standards'. John selected two non-functional requirements that were motivated by the given principle, NFR01 'scalability' and NFR02 'interoperability'. This is signified by the relationship 'motivates' between the NFRs and the principle. By doing so, EA Anamnesis provides insight into the reasons behind the elicitation of non-functional requirements.

**Replaying decision making processes**
We continue our analysis by further zooming into the decision making process for D02 'acquisition of COTS application B' (Figure 4.5). Based on the notion of decision making strategies, we now replay the decision process leading up to the creation of D02.

John relied on EA Anamnesis for the structuring of his decision making problem. To start

Table 4.1: EA design decision 02 details

| | |
|---|---|
| **Decision:** | D02: acquisition of COTS application B |
| **Functional requirement:** | F02: find appropriate application to support new business process |
| **EA perspective:** | application layer |
| **Alternatives:** | AL01: acquisition of COTS application A<br>AL02: acquisition of COTS application B<br>AL03: acquisition of COTS application C<br>AL04: upgrade application |
| **Non-functional requirements:** | NFR01: scalability, NFR02: interoperability, NFR03: usability, NFR04: cost |
| **Unanticipated observed Impact:** | degraded user experience in the application use |

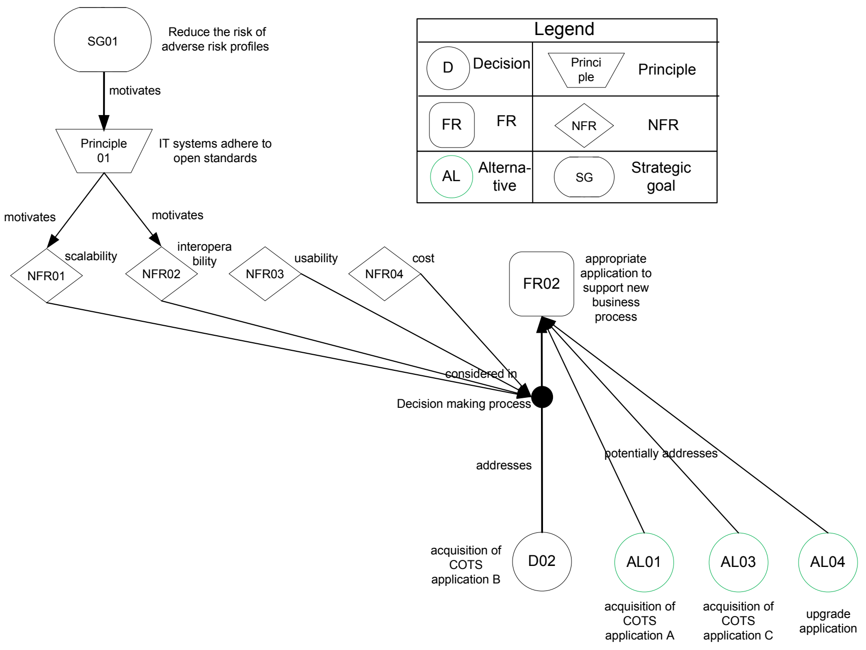


Figure 4.5: Decision making process of D02

the decision making process, John was given the functional requirement FR02 ‘appropriate application to support new business process’, elicited the non-functional requirements that the new application had to satisfy. The non-functional requirements for application selection are grounded in (Jadhav and Sonar 2009).

For our illustrative example, John considered four non-functional requirements. He started with the NFRs‘usability’, ‘interoperability’ and ‘scalability’ and afterwards he took into account NFR ‘cost’. Thereafter, he identified four alternatives to choose from: three alternative COTS applications and the fourth one to upgrade the existing application in house.

John, through an investigation of various specification documents for application systems in the enterprise, identified the numerical weights of importance for each non-functional requirement. He selected the MAUT decision making strategy in order to make an trade-off analysis among the alternatives.

In the meanwhile, John received a constraining budget limit of €10000 for the acquisition of the new IT system. On the one hand, he wanted to carefully evaluate the four alternatives w.r.t. the non-functional requirements ‘usability’, ‘interoperability’ and ‘scalability’ (via a compensatory strategy), but on the other hand he had to account for the hard constraint of ‘cost’. John decided to use for the latter case a non-compensatory strategy. In order to discard alternatives that failed to meet the budget constraint, he used a disjunctive decision making strategy.

Table 4.2 summarizes the results of his non-compensatory evaluation. ‘Acquisition of COTS application C’ is eliminated from the choice set because it failed to meet the threshold value for the non-functional requirement ‘cost’.

For the compensatory part, John conducted an extensive trade-off analysis among the alternatives by taking into account the performance of alternatives w.r.t non-functional requirements under evaluation.

As we saw above, the non-functional requirement NFR01 ‘scalability’ and NFR02 ‘inter-

Table 4.2: EA decision 02 disjunctive strategy

| **alternatives** | **threshold** | **value** | **score** |
|---|---|---|---|
| acquisition of COTS app A | € 10000 | € 9000 | pass |
| acquisition of COTS app B | € 10000 | € 8000 | pass |
| acquisition of COTS app C | € 10000 | €12000 | fail |
| upgrade application | € 10000 | € 5000 | pass |

operability' were motivated by the EA principle 'IT systems adhere to open standards'. John's decision making was steered by the given principle and he assigned high relative importance weights to these non-functional requirements. NFR03 'usability' was a concern raised by the domain stakeholders, but John did not considered it as a concern of high importance.

John, relied on EA Anamnesis to structure and capture his trade-off analysis based on a MAUT compensatory decision making strategy. The scores for each alternative, as well the performance values and weights of importance of non-functional requirements are shown in Table 4.3. According to the Formula 3.2, the alternative 'acquisition of COTS application B' has the highest score and it was therefore selected.

The combination of a MAUT and a disjunctive strategy is a typical example of a decision making process with multiple strategies. Figure 4.6 shows the summarization of decision making process for D02.

**Reflecting upon the captured decision making process:**
So far, we have illustrated the decision process as captured by John. We now move two years forward to further illustrate how EA Anamnesis supports future decision making. The customer profiles of the car insurance of ArchiSurance are better calculated by using insurance intermediaries. As a result, the management of ArchiSurance decides to also rely on intermediaries for the remainder of its insurance products. *Mary*, a new enterprise architect, is responsible for establishing the use of intermediaries in the EA. To do so she inspects the already established EA that support the intermediaries scenario. She identifies through EA models the intermediary business process and she realizes that she can implement the EA with a similar way.

At this stage Mary, wants to know how her predecessor has supported the introduction of an intermediary with IT applications. To this end, she can rely on the rationale information of a similar issue captured by her predecessor in EA Anamnesis. Mary identifies that her predecessor, John, had to address the functional requirement 'Appropriate ap-

Table 4.3: EA decision 02 MAUT strategy

| alternatives | usability | | interoperability | | scalability | | score |
|---|---|---|---|---|---|---|---|
| | value | weight | value | weight | value | weight | |
| acquisition of COTS app A | 7 | 2 | 7 | 5 | 7 | 10 | 119 |
| acquisition of COTS app B | 8 | 2 | 3 | 5 | 9 | 10 | 121 |
| upgrade application | 9 | 2 | 5 | 5 | 4 | 10 | 83 |

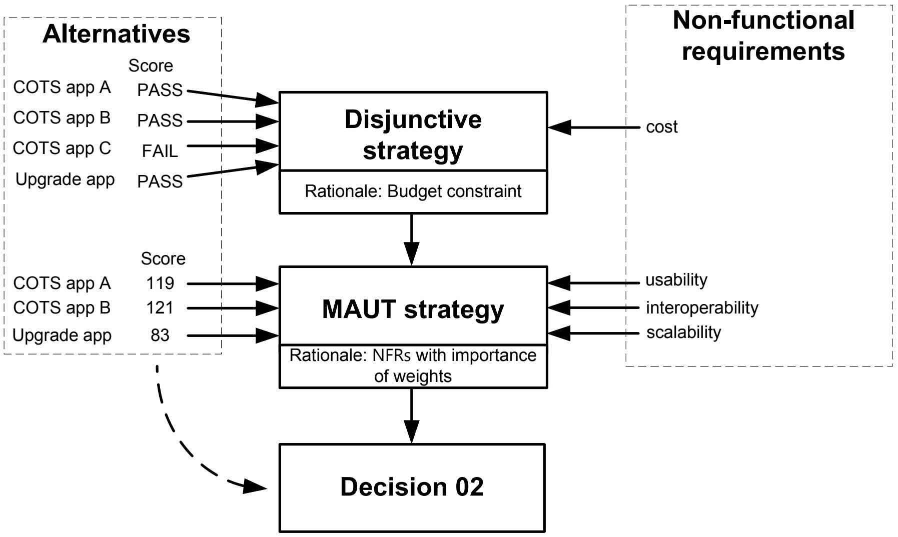


Figure 4.6: Decision making strategies of D02

plication to support new business process'. By inspecting the graph of Figure 4.4 she can identify how the specific functional requirement FR02 was addressed by design decision D02. Mary can see that D02 belongs to the Application layer of the enterprise and more specifically results in the EA element 'COTS application B'.

Mary can now analyze the rationale behind D02 and identify (1) the used decision making strategy, (2) why this strategy was selected, (3) the importance of non-functional requirements for the evaluation process and (4) the unanticipated observed impact of the design decision across the EA.

Mary inspects the captured decision making strategies, as they are shown in Figure 4.6, and understands that a hybrid decision strategy scheme was used. She realizes that the non-functional requirement 'cost' was used in a disjunctive non-compensatory strategy to discard the alternative 'acquisition of COTS application C'. Furthermore, Mary observes that a MAUT compensatory strategy was used to evaluate the remaining alternatives. She realizes that her predecessor used this strategy, because the non-functional requirements 'usability', 'interoperability' and 'scalability' did not have the same importance for the selection of an appropriate IT application for supporting the new business process of ArchiSurance.

Mary realizes that D02 'Acquisition of COTS application B' created an unanticipated observed impact on the business layer of the architecture. More specifically, on the newly introduced business process ‘customer profile registration’. More specifically, observed impact OI01 ‘degraded user experience in the application use’ imposed difficulties on the users executing the business process and resulted in limited productivity. She can also see that OI01 motivated a new functional requirement FR03 ‘find proper user interface (UI)’ and that was addressed by D04 ‘user interface similar to the old one’. In other words, John solved the problem imposed by the problematic user interface by selecting a user interface that users are already familiar with.

Based on the analysis of past design decisions and their implications on the EA, Mary can consider for her current decision making problem the decision making strategies, alternatives and non-functional requirements, as well as their relative importance. She can avoid decisions or problems that may come along. For example, by inspecting Table 4.3, Mary can realize that her predecessor assigned a low importance value to the non-functional requirement ‘usability’. Mary will now consider more carefully the non-functional requirement ‘usability’ during the decision making process and she can inform the relevant stakeholders for possible upcoming problems in the execution of the new business processes. She can either propose that the new IT application should have a user interface that users are familiar with or that an organized training session should be offered to make users familiar with new user interface.

In summary, we have illustrated that the reconstruction of the decision making process makes transparent *how* a design decision has been made. Amongst others, this transparency allows an architect to compare the outcome of a design decision with the non-functional requirements that led to this decision. As a result, architects can learn which factors in the decision making process had a positive/negative impact on the EA design and follow/avoid these decision making practices for future decisions.

## 4.4 Conclusion

In this chapter we introduced our specialized conceptual framework for EA design rationalization. We introduced new concepts for the formulation of the design problem based on the current EA practice and relevant EA approaches. By doing so we provided an EA specific answer to *RQ3* (Section 1.2). Moreover, we answered *RQ4* (Section 1.2) by using the notion of EA perspective, which helped us categorize design rationales and make explicit their possible cross-domain relationships.

# Part III

# Evaluation and Reflection on EA Anamnesis

# CHAPTER 5

# Implementation of a software demonstrator

In this chapter we present a software demonstrator that was implemented during the iterative process of the development of our design rationale conceptual framework. We used this software tool for the assessment of an intermediary version of the EA Anamnesis conceptual framework, which subsequently helped us improve and extend the metamodel.

The content of this chapter is based on work published as:

Georgios Plataniotis and Sybren de Kinderen and Henderik A. Proper: A computational approach for design rationalization in Enterprise Architecture. In: *Proceedings of 8th International Conference on Research Challenges in Information Science (RCIS)*, pp. 1-2, IEEE, 2014

Georgios Plataniotis and Sybren de Kinderen and Henderik A. Proper: Implementing a Software Prototype for Enterprise Architecture Rationalization: Lessons Learned. In *Proceedings of the 18th International Enterprise Distributed Object Computing Conference Workshops and Demonstrations (EDOCW)*, pp. 41-46, IEEE, 2014

## 5.1 Objectives and development environment

As we have discussed in Section 1.4, the development of a design artifact involves intermediate iteration steps, which are used for the further improvement and extension of the artifact. In the context of this iterative process, we developed a software tool. To do so, we used the conceptual framework as a language with a basic concrete syntax. The main goal of this implementation was to demonstrate our conceptual framework to practitioners and provide evidence that the conceptual framework can indeed be implemented with a software tool. Furthermore, we made a first step towards the evaluation of our conceptual framework through computational assessment.

We have three key aims for developing tool support:

(1) to showcase a (rudimentary) software tool support to practitioners as a means to demonstrate implementability of our conceptual framework. We consider this relevant as tool support fosters the practical uptake of modeling languages and frameworks (Malavolta et al. 2013),

(2) to process practitioner feedback for further improvement of our design artifact. Here, we aim at showing the tool as a proof-of-concept during case studies, so that practitioners can react to the presented tooling support in terms of, for example, usefulness of the framework concepts, and missing concepts and/or functionality. Subsequently, practitioner feedback can be processed concurrently in the conceptual framework and software tool,

(3) to provide for a computational assessment of our conceptual framework. Here, we aim at testing to what extent the conceptual framework and corresponding concrete syntax (see Section 5.2) can indeed be implemented with a software tool. Furthermore, the

computational assessment forces one to be specific about the conceptual framework, thus possibly leading to conceptual framework changes.

Following objectives (1) and (2), a key requirement is the ability to develop our software tool by rapid prototyping, so that practitioner feedback and conceptual framework amendments can be processed without the need for extensive coding. The Microsoft Visual Studio 2013 DSL environment, which we used for tool development, allows for such rapid prototyping. It provides an graphical editor that allows us to implement our conceptual framework and its corresponding concrete syntax. No further coding is required.

Figure 5.1 presents a sample of the development environment during tool implementation. In this figure, we can see the concept of 'EADecision', and the relation of this concept with other concepts of our conceptual framework, including relevant cardinalities.

The source files of our prototype implementation, as well as the instantiation presented in the next section, can de downloaded from:
https://github.com/georgeplat/EA-Anamnesis-prototype/[1].

[1]Requires a (trial) version of visual studio

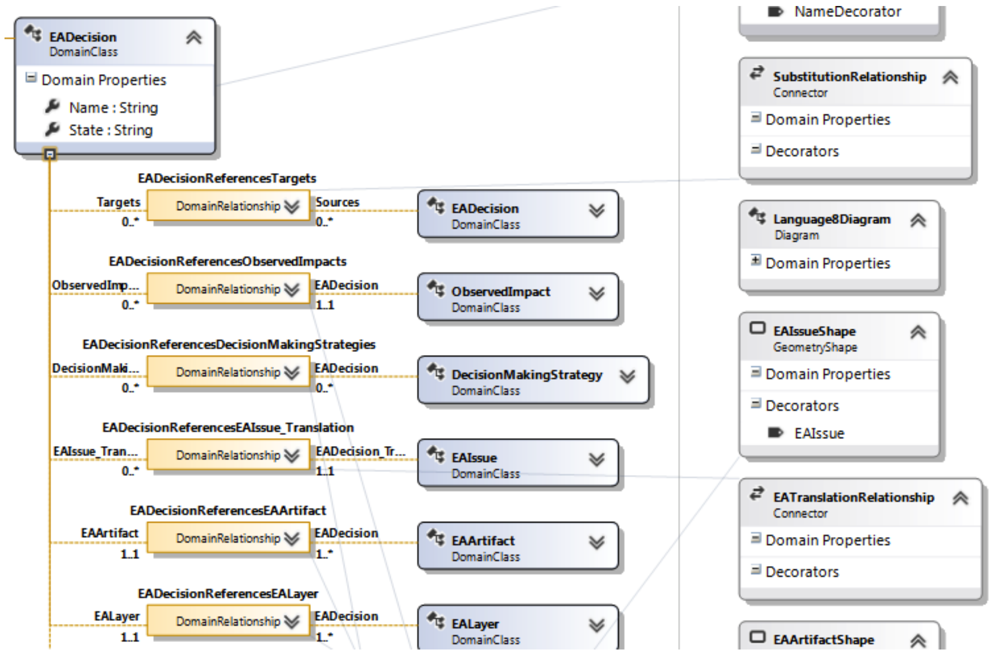


Figure 5.1: Prototype development

**Software tool functionality:**

As described above, the prototype implementation is a based on an intermediate version of EA Anamnesis conceptual framework. Therefore, we have to state here that some of the elements of this tool do not exactly conform to the latest version of the conceptual framework. For example, the tool does not address aspects related with problem space analysis. Nonetheless, that was exactly one of the goals of this prototype implementation, to be able to illustrate our framework to practitioners and gain insights that would help us to identify what is additionally needed or what has to be improved.

Our software tool allows architects to rationalize architectures through a Graphical User Interface that instantiates our conceptual framework. Furthermore, the software tool can export instantiations of the conceptual framework to a machine-interpretable, XML-based, output. By doing so, our software tool allows architects to rationalize architectures through an accessible interface and export it for further processing, hiding at the same time the technicalities of the conceptual framework.

## 5.2 Tool illustration

Here we illustrate how the prototype gives us visual representation. For the illustration we use the Archisurance scenario of Section 4.3.1.

Figure 5.2 presents a snapshot of the design rationale model that was created based on our conceptual framework. On this figure, we can see the Toolbox that is provided by the prototype development environment which contains the elements of our metamodel. As we previously mentioned, the tool implementation does not conform exactly to the latest version of the metamodel. For example, at the time of the prototype implementation, EA Anamnesis did not support a problem space analysis. Subsequently for some tool elements we used different terminology. For better comprehension of the tool visualization, we provide in Table 5.1 the mapping between the concepts and relationships of the latest EA Anamnesis metamodel and the tool elements.

The prototype environment allowed us to create instantiations of our conceptual framework by dragging and dropping the appropriate elements from the toolbox to the area of the DSL definition diagram (right part of Figure 5.2). For our purposes we created an instantiation based on the Archisurance example of Section 4.3. As an example we present the instantiation of the conceptual framework based on the design rationale graph presented in Figure 4.4.

We started by using the elements 'EAIssue' and 'Decision' in order to capture the motivational reason (IS01 - 'establish business process intermediary') behind the execution of

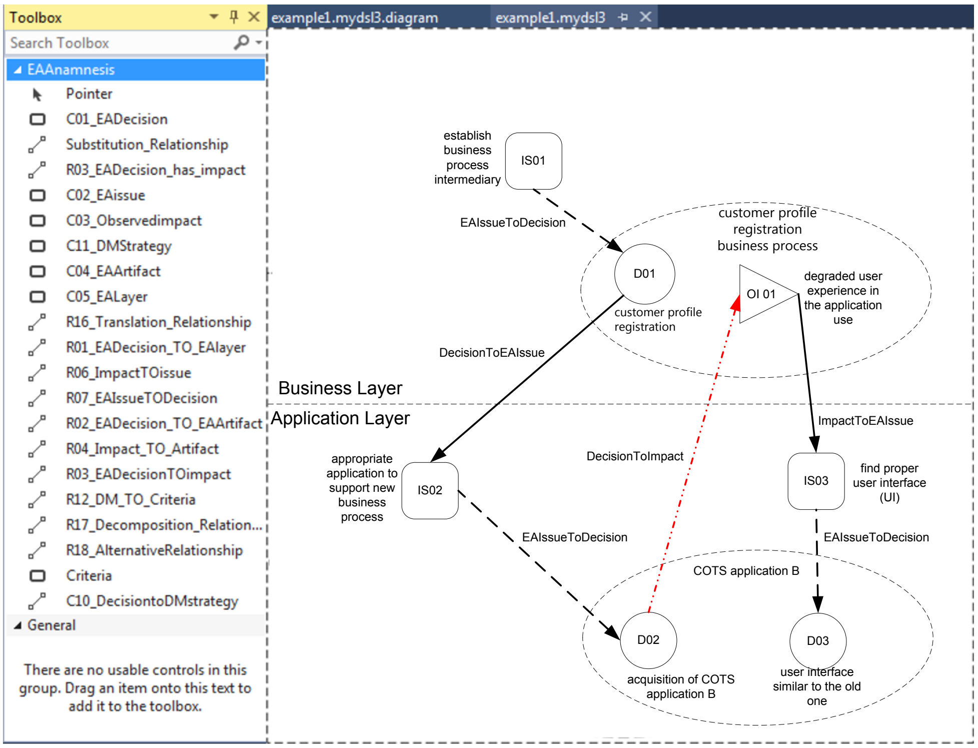

Toolbox
Search Toolbox
EAAnamnesis
Pointer
C01_EADecision
Substitution_Relationship
R03_EADecision_has_impact
C02_EAissue
C03_Observedimpact
C11_DMStrategy
C04_EAArtifact
C05_EALayer
R16_Translation_Relationship
R01_EADecision_TO_EAlayer
R06_ImpactTOissue
R07_EAIssueTODecision
R02_EADecision_TO_EAArtifact
R04_Impact_TO_Artifact
R03_EADecisionTOimpact
R12_DM_TO_Criteria
R17_Decomposition_Relation...
R18_AlternativeRelationship
Criteria
C10_DecisiontoDMstrategy
General
There are no usable controls in this group. Drag an item onto this text to add it to the toolbox.
example1.mydsl3.diagram
example1.mydsl3
establish business process intermediary
IS01
EAIssueToDecision
customer profile registration business process
D01
customer profile registration
OI 01
degraded user experience in the application use
DecisionToEAIssue
Business Layer
Application Layer
ImpactToEAIssue
DecisionToImpact
appropriate application to support new business process
IS02
IS03
find proper user interface (UI)
EAIssueToDecision
EAIssueToDecision
COTS application B
D02
D03
acquisition of COTS application B
user interface similar to the old one

Figure 5.2: Prototype tool - Metamodel instantiation

design decision D01 ‘customer profile registration’. We also used the relationship ‘EAIssueToDecision’ in order to connect the two elements. Furthermore, we associated the elements of our instantiations with their corresponding EA artifacts and EA layers. For example D01 is associated with ‘Business Layer’ and with EA artifact ‘customer profile registration business process’. D01 triggered a new EA Issue IS02 ‘appropriate application

Table 5.1: Mapping between EA Anamnesis metamodel and tool elements

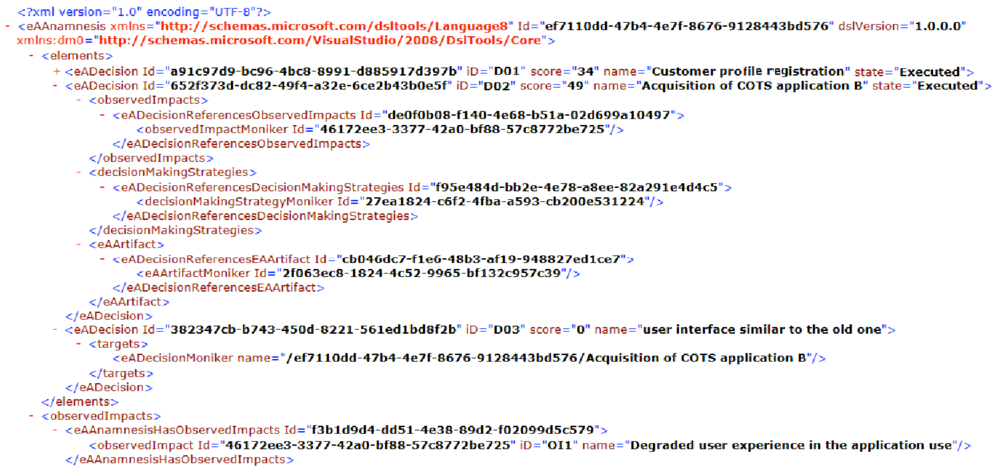
| | **EA Anamnesis** | **Prototype tool** |
|---|---|---|
| **Concept** | Functional requirement | EA issue |
| **Relationship** | Addresses | EAIssueToDecision |
| **Relationship** | Motivates | DecisionToEAIssue |
| **Relationship** | Causes | DecisionToImpact |

to support new business process'. We used the relationship 'DecisionToEAIssue' to connect D01 and IS02. Following the same process we interrelated IS02 with D02 'acquisition of COTS application B'. As we saw in Section 4.3, D02 has an anticipated observed impact on the business layer of the architecture. We used the tool element observed impact OI01 - 'degraded user experience in the application use' and the relationship 'ImpactToEAIssue' to connect OI01 with IS03 'find proper user interface (UI)'. IS03 was addressed by D03'.

**XML output:**

Through the DSL instantiation environment we were able to create a visualization of the design rationale information from the Archisurance example. Furthermore, the DSL development environment enabled us to extract the instantiation of our example to its corresponding XML format. A sample of the extracted instantiations in to XML, is shown in Figure 5.3. The implementation of the software prototype and the extraction of design rationale instantiations showcases that our conceptual framework can be used as a basis for additional computational support. As such our conceptual framework can be used as a basis for the satisfaction of the 'Technical actor interpretation appropriateness' quality as discussed in Section 1.4.3.

For example, data mining techniques can be based on our prototype to further process design rationale information, that will allows us to identify architecture patterns, good or bad practices of decision making etcetera.



```
  <?xml version="1.0" encoding="UTF-8"?>
- <eAAnamnesis xmlns="http://schemas.microsoft.com/dsltools/Language8" Id="ef7110dd-47b4-4e7f-8676-9128443bd576" dslVersion="1.0.0.0"
  xmlns:dm0="http://schemas.microsoft.com/VisualStudio/2008/DslTools/Core">
  - <elements>
    + <eADecision Id="a91c97d9-bc96-4bc8-8991-d885917d397b" iD="D01" score="34" name="Customer profile registration" state="Executed">
    - <eADecision Id="652f373d-dc82-49f4-a32e-6ce2b43b0e5f" iD="D02" score="49" name="Acquisition of COTS application B" state="Executed">
      - <observedImpacts>
        - <eADecisionReferencesObservedImpacts Id="de0f0b08-f140-4e68-b51a-02d699a10497">
            <observedImpactMoniker Id="46172ee3-3377-42a0-bf88-57c8772be725"/>
          </eADecisionReferencesObservedImpacts>
        </observedImpacts>
      - <decisionMakingStrategies>
        - <eADecisionReferencesDecisionMakingStrategies Id="f95e484d-bb2e-4e78-a8ee-82a291e4d4c5">
            <decisionMakingStrategyMoniker Id="27ea1824-c6f2-4fba-a593-cb200e531224"/>
          </eADecisionReferencesDecisionMakingStrategies>
        </decisionMakingStrategies>
      - <eAArtifact>
        - <eADecisionReferencesEAArtifact Id="cb046dc7-f1e6-48b3-af19-948827ed1ce7">
            <eAArtifactMoniker Id="2f063ec8-1824-4c52-9965-bf132c957c39"/>
          </eADecisionReferencesEAArtifact>
        </eAArtifact>
      </eADecision>
    - <eADecision Id="382347cb-b743-450d-8221-561ed1bd8f2b" iD="D03" score="0" name="user interface similar to the old one">
      - <targets>
          <eADecisionMoniker name="/ef7110dd-47b4-4e7f-8676-9128443bd576/Acquisition of COTS application B"/>
        </targets>
      </eADecision>
    </elements>
  - <observedImpacts>
    - <eAAnamnesisHasObservedImpacts Id="f3b1d9d4-dd51-4e38-89d2-f02099d5c579">
        <observedImpact Id="46172ee3-3377-42a0-bf88-57c8772be725" iD="OI1" name="Degraded user experience in the application use"/>
      </eAAnamnesisHasObservedImpacts>
```

Figure 5.3: XML output sample of the illustrative example

## 5.3 Lessons

The prototype software tool helped us, amongst others, to identify some of the limitations of the intermediate version of EA Anamnesis. More specifically, while we were experimenting with the DSL instantiation environment, we identified some information redundancy problems. We illustrate through two lessons how our prototype tool helped us to further improve EA Anamnesis. These findings have been embedded in the latest version of our metamodel.

### Lesson 1: Redundancy of information regarding decision making strategies

The intermediate version of EA Anamnesis did not support the assignment of a decision making strategy to multiple alternatives. More specifically, while we were capturing the design making strategies of the Archisurance scenario, we were faced with the situation where we had to relate the element 'decision making strategy' to each of the alternatives of the evaluation set. As such, we had to create multiple copies of the same decision making strategy and connect it to each of the alternatives. This resulted in unjustifiable redundancy of information. We could instead use a single decision making strategy for the particular evaluation scenario and be able to connect this element with the alternatives that were considered during the evaluation process. We improved our metamodel by changing the multiplicity between the concept 'decision making strategy' and 'alternative'. Figure 5.4 depicts the modification in the metamodel, and Figure 5.5 depicts the modified instantiation of the metamodel with regard to this aspect.

### Lesson 2: Redundancy of information regarding the status of alternatives

The prototype implementation was based in a metamodel version that contained the concept 'state', which was used for the determination of the status of design decisions

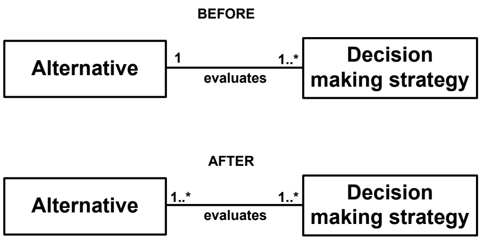


Figure 5.4: Modification in the metamodel for Lesson 1 (decision making strategies)

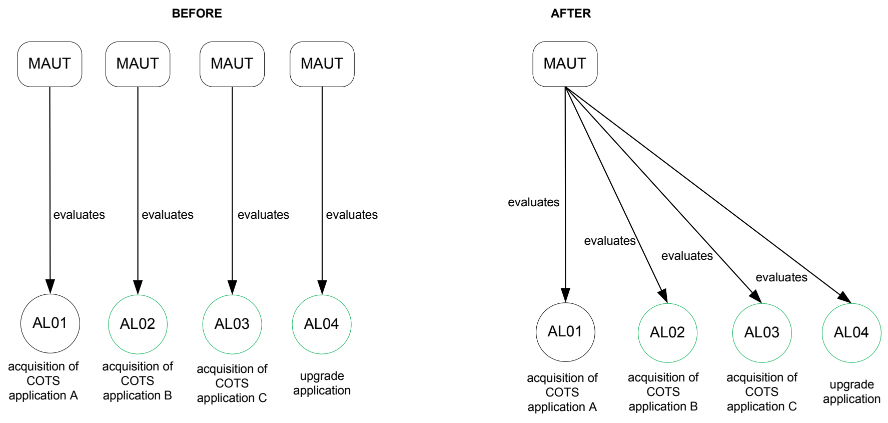

BEFORE
MAUT
MAUT
MAUT
MAUT
evaluates
evaluates
evaluates
evaluates
AL01
AL02
AL03
AL04
acquisition of COTS application A
acquisition of COTS application B
acquisition of COTS application C
upgrade application
AFTER
MAUT
evaluates
evaluates
evaluates
evaluates
AL01
AL02
AL03
AL04
acquisition of COTS application A
acquisition of COTS application B
acquisition of COTS application C
upgrade application

Figure 5.5: Modified instantiation of the metamodel with regard to Lesson 1

'executed' or 'rejected'. The experimentation with the DSL instantiation made us realize that this concept was also resulting to unjustifiable information redundancy. Instead of capturing the state of decisions and alternatives, we simply capture the selected (executed) alternative, by using the concept 'decision' and the rejected alternatives, by using the concept 'alternative'. Figure 5.6 depicts the modification in the metamodel and Figure 5.7 depicts the modified instantiation of the metamodel with regard to this aspect.

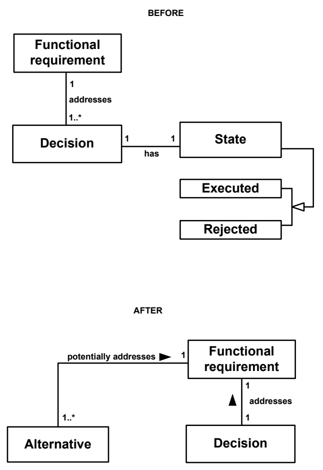


Figure 5.6: Modification in the metamodel for Lesson 2 (state)

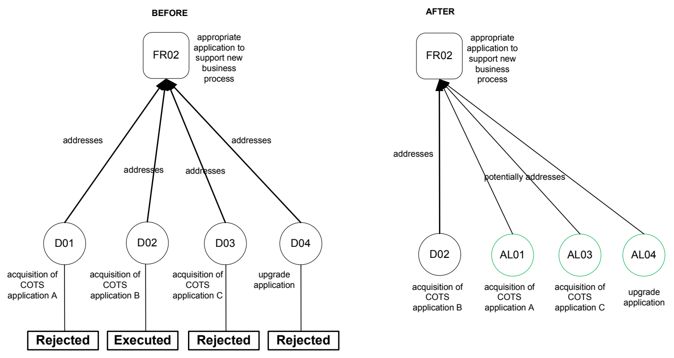

BEFORE
FR02
appropriate application to support new business process
addresses
addresses
addresses
addresses
D01
D02
D03
D04
acquisition of COTS application A
acquisition of COTS application B
acquisition of COTS application C
upgrade application
Rejected
Executed
Rejected
Rejected
AFTER
FR02
appropriate application to support new business process
addresses
potentially addresses
D02
AL01
AL03
AL04
acquisition of COTS application B
acquisition of COTS application A
acquisition of COTS application C
upgrade application

Figure 5.7: Modified instantiation of the metamodel with regard to Lesson 2

## 5.4 Conclusion

In this chapter we introduced a prototype and conducted a computational assessment of EA Anamnesis. The computational assessment and the corresponding visualization showed the implementability of our conceptual framework and in parallel provided us with some important lessons. Based on these lessons, some modifications were proposed. Furthermore, the selection of a rapid prototype implementation strategy gave us the capability to be flexible to adapt our design based on the feedback that we received on the next steps of our research.

# CHAPTER 6

# Applying EA Anamnesis in a Luxembourgish RTO

In this chapter we evaluate the EA Anamnesis in terms of its ability to capture design rationale in the context of a real life case study. Together with stakeholders from the business and IT domains of a Luxembourgish Research and Technology Organization (LuxRTO), we captured the design rationale behind the introduction of a new budget forecast business process in EA Anamnesis. Our case study shows that EA Anamnesis can reflect the design rationale and link business and IT concerns. Furthermore, our study shows that, for this particular case, the stakeholders often used heuristics (commonsensical 'short cuts') to make their decision, or even made decisions without considering alternative choices.

The chapter is structured as follows. Section 6.1 presents the objectives and case setup and Sections 6.2, 6.3 introduce the LuxRTO case and details of the enterprise transformation. Thereafter, Section 6.4 presents the design rationale captured by EA Anamnesis and Section 6.5 the outcomes of the evaluation. Finally, Section 6.6 presents the lessons learned and Section 6.7 concludes.

The content of this chapter is based on work published as:

Georgios Plataniotis, Sybren de Kinderen and Henderik A. Proper: Capturing Design Rationales in Enterprise Architecture: A Case Study. In: *Proceedings of 7th IFIP WG 8.1 working conference on the Practice of Enterprise Modelling (PoEM)*. Springer, 2014

Georgios Plataniotis, Sybren de Kinderen, Qin Ma and Henderik A. Proper: A Conceptual Model for Compliance Checking Support of Enterprise Architecture Decisions. In: *Proceedings of 17th Conference on Business Informatics (CBI)*, vol. 1, no. 3, pp. 191-198, IEEE, 2015

## 6.1 Objectives and case setup

The main goal of this case study is to review to what extent EA Anamnesis satisfies, in the context of a *real life* enterprise transformation, the quality characteristics as discussed in Section 1.4.3.

To this end, we study one particular transformation: the introduction of a new budget management business process at LuxRTO. We organized interviews with two key stakeholders that were involved in the transformation: The financial officer, and the IT architect. Both stakeholders provided a good starting point for the domain knowledge that we had to capture. On the one hand, the financial officer possessed significant business expertise on this enterprise transformation project. Being involved from the start of the transformation project, she had knowledge about the drivers that initiated this transformation and how the business process design evolved over time. On the other hand, the IT architect had significant IT expertise on the transformation project. Furthermore, the stakeholders provided us with the documentation of this transformation project (text documents, presentations, emails). We used the ArchiMate modeling language and more specifically the EA perspectives 'business layer', 'application layer' to capture the various states of the EA.

We started our case study by presenting EA Anamnesis to the financial officer and the IT architect. We explained the goals and challenges of our case study, and we illustrated our conceptual framework using the example case of Section 4.3.1. This example case helped the stakeholders to understand EA Anamnesis.

After the presentation of EA Anamnesis, we conducted a collaborative modeling exercise with the two stakeholders. The goal of this exercise was to evaluate EA Anamnesis based on the qualities, as discussed in Sections 1.4.2, 1.4.3. More specifically, we evaluate EA Anamnesis in terms of its:

- Domain appropriatenes
- Participant language knowledge appropriateness
- Knowledge externalizability appropriateness

For example, our goal was to identify if EA Anamnesis was able to capture design rationale of this transformation and identify the perception of stakeholders regarding the concepts of EA Anamnesis.

Note that the setup above is inspired by the main steps for conducting case study research set out in Runeson and Host (2009). For example: prior to the collaborative modeling we demonstrated EA Anamnesis to practitioners through the software tool.

**Limitations:**

In this part, we discuss limitations that have potentially played a role in the application of EA Anamnesis in LuxRTO and in the interpretation of the results of this study.

The first limitation is that the actual enterprise transformation was held around two years before the case study. This implies that stakeholders may had a bias in what information they captured during the case study (colored memory) or they may have forgotten certain things. Another limitation is the number of stakeholders that participated in the case study. Normally, multiple stakeholders participate in an EA transformation. In our case, we interviewed two stakeholders (one from the business domain and one from IT). In that sence, our LuxRTO case study was conducted in a more ‘controlled’ way.

## 6.2 Budget forecasting at LuxRTO

Here we present the introduction of a new budget management business process, and how this process was supported by information systems in the context of an EA transformation.

During the last years, the Luxembourgish government introduced stricter rules on the budget spending of research institutions. This policy had to be incorporated by the research institutions, meaning that the institutions should be able to establish long term financial projection plans. This would give institutions a better awareness regarding the availability of resources and in turn the planning of future projects and personnel hiring.

LuxRTO did not have an established business process for the budget estimation. Stakeholders, from the management side of LuxRTO, had to design this new business process from scratch. Their initial objectives were that this business process should provide (1) a clear view on human resources and projects coverage, (2) an input for the future hiring plan, (3) a comparison between the forecasted and valuable budget, and (4) in general robustness of the financial data of the organization. Last but not least, a training for the users of this new business process should be organized.

## 6.3 Enterprise Transformation

In this Section we describe how the enterprise design was changed in order to support the new budget estimation business process. Note that LuxRTO had already established IT systems that supported other types of financial, project and human resources business processes. Before we present the transformation, we briefly describe the new business process and the already established IT systems. In order to express the EA design of the budget forecast project, we used the ArchiMate modeling language.

**Budget forecast business process:**

The main objectives of this business process are the estimation and the planning of resources to ensure the planning activities, the assessment of the need for additional resources, the estimation of the associated budgets and the checking of the forecast in relation to the available budget in LuxRTO. The role of the business process is to provide annual budget estimates, which should be validated and approved by the finance department.

### 6.3.1 Baseline architecture

Figure 6.1 presents the EA model of the baseline architecture before the incorporation of the budget forecast business process. Below we provide a description of the existing IT systems.

Application A is the main financial application of the organization. The main functionalities of this application are management of procurements, traveling costs, personal costs, overhead costs calculation, salaries payment and project dashboard. The access to this application is controlled and only financial officers are allowed to use it.

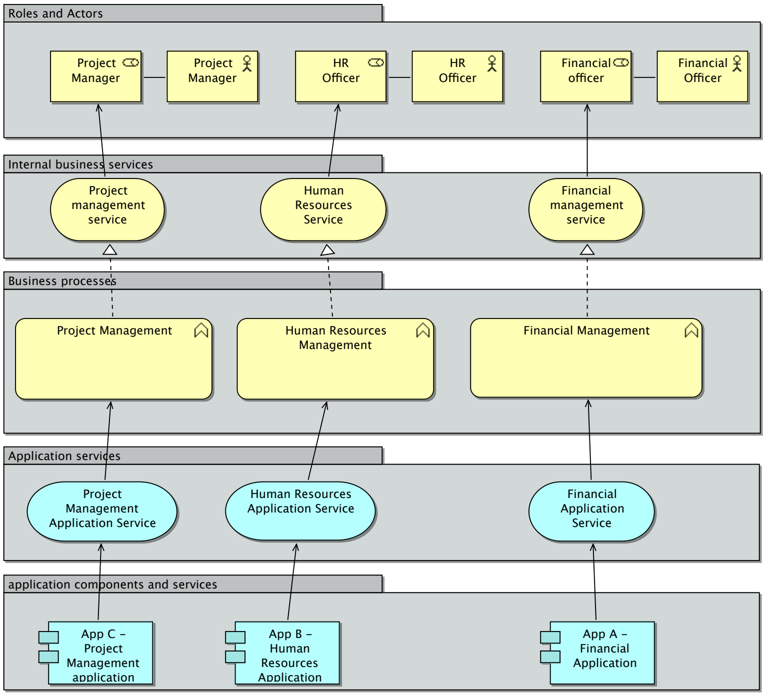

Roles and Actors
Project Manager
Project Manager
HR Officer
HR Officer
Financial officer
Financial Officer
Internal business services
Project management service
Human Resources Service
Financial management service
Business processes
Project Management
Human Resources Management
Financial Management
Application services
Project Management Application Service
Human Resources Application Service
Financial Application Service
application components and services
App C – Project Management application
App B – Human Resources Application
App A – Financial Application

Figure 6.1: LuxRTO enterprise transformation - Baseline architecture

Application B is the human resources management application. Tasks like resource allocation, start/end dates of work contracts, weekly calendar, different types of leaves (sickness, vacation etcetera) are executed by this application.

Application C is the project management application of the organization. The actual hours assigned per project in the organization are maintained in this application.

## 6.3.2 First iteration of the transformation

Figure 6.2 depicts the EA model after the incorporation of the budget forecast business process. From this model, we can observe that the business process was supported by the interaction and collaboration among Applications A, B, C and an additional spreadsheet application. However, due to some problems, which can not been described by ArchiMate, stakeholders had to do additional changes in the EA design.

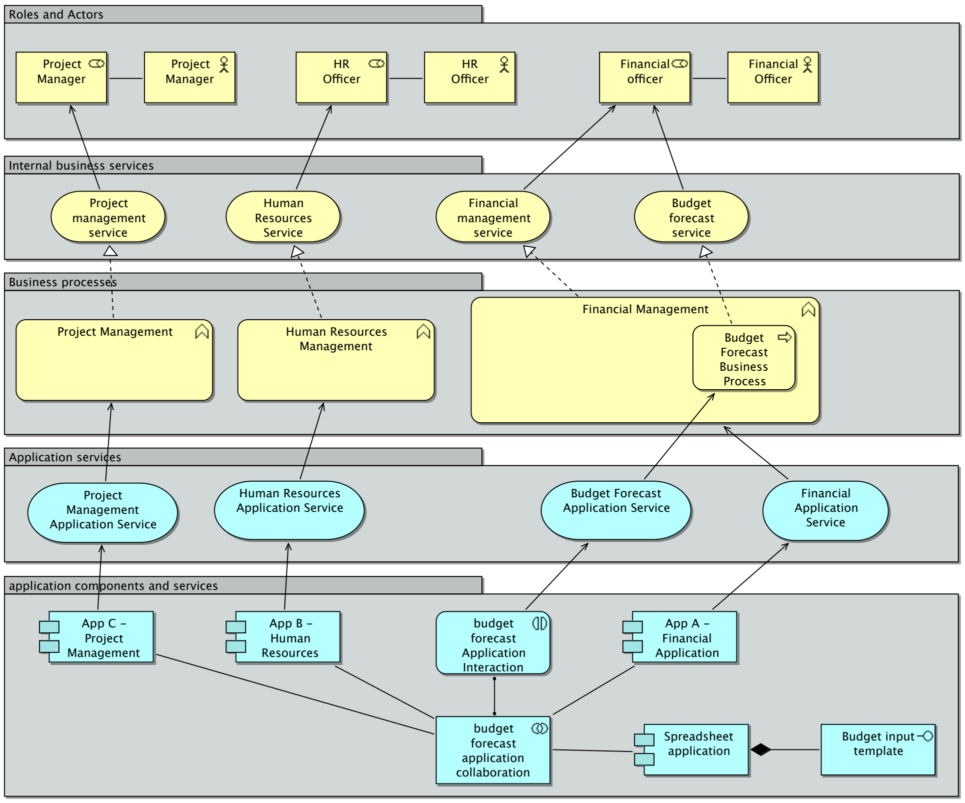

Roles and Actors
Project Manager
Project Manager
HR Officer
HR Officer
Financial officer
Financial Officer
Internal business services
Project management service
Human Resources Service
Financial management service
Budget forecast service
Business processes
Project Management
Human Resources Management
Financial Management
Budget Forecast Business Process
Application services
Project Management Application Service
Human Resources Application Service
Budget Forecast Application Service
Financial Application Service
application components and services
App C – Project Management
App B – Human Resources
budget forecast Application Interaction
App A – Financial Application
budget forecast application collaboration
Spreadsheet application
Budget input template

Figure 6.2: LuxRTO enterprise transformation - First iteration

### 6.3.3 Second iteration of the transformation

Figure 6.3 depicts the final iteration of the enterprise transformation. With this iteration stakeholders managed to address the aforementioned problems. Instead of using spreadsheets for entering the budget data, a new application interface was added in the financial Application A.

## 6.4 The rationale behind the budget forecast design

In the previous subsections we described the changes happened in the EA design, in order to support the new budget forecast business process. However, the rationale behind this design is not captured by the EA models. Based on the case study, we could potentially

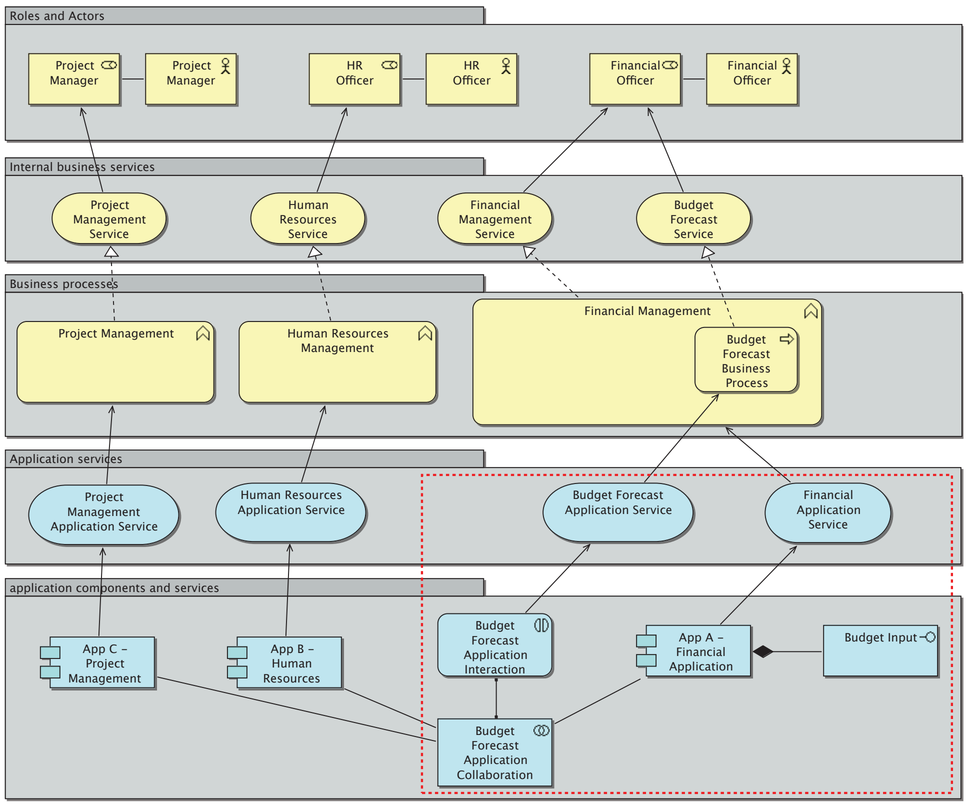

Roles and Actors
Project Manager
Project Manager
HR Officer
HR Officer
Financial Officer
Financial Officer
Internal business services
Project Management Service
Human Resources Service
Financial Management Service
Budget Forecast Service
Business processes
Project Management
Human Resources Management
Financial Management
Budget Forecast Business Process
Application services
Project Management Application Service
Human Resources Application Service
Budget Forecast Application Service
Financial Application Service
application components and services
App C – Project Management
App B – Human Resources
Budget Forecast Application Interaction
App A – Financial Application
Budget Input
Budget Forecast Application Collaboration

Figure 6.3: LuxRTO enterprise transformation - Second iteration

ask these questions:

- Why were these IT systems selected for the realization of the business process?
- Were there any other alternatives?
- How do the EA design decisions comply with the given goal, principles and requirements?
- What was the role of goals, principles and requirements during the decision making process for the selection of the best alternatives?
- What were the unanticipated consequences of these decisions in the EA?

By answering these questions we provide traceability within the solution space and the problem space of the architecture, but also a bridging between the two spaces. The answers to these questions provide a useful insight in the understanding of the EA design, which can not be achieved just by examining EA models.

Below, we use EA Anamnesis to capture design rationale. We follow the concrete syntax of Section 4.3.2 and we provide the captured design rationale by means of a design rationalization graph, which represents design rationale across the architecture design and their various interrelationships. Subsequently, we present an analysis of a specific design decision in order to graphically present which requirements were considered for the selection of a specific design decision, their compliance with the given principles and what were the rejected alternatives. Finally, we further zoom into the decision making problem and we present the reasoning of the involved stakeholders.

### 6.4.1 Design rationalization graph

Figure 6.4 provides the design rationalization graph of the LuxRTO enterprise transformation. Here, we can see how the design problem is formulated by means of goals and requirements and how it is addressed by specific design decisions in the EA design. We can also see unanticipated observed impacts of these design decision in the architecture. The graph is accompanied by Table 6.1, which provides a summary of the design rationale information.

Our analysis starts with the strategic goal SG01 'having long term financial projection plans'. G01 motivated the elicitation of a functional requirement in the business layer of the architecture FR01 'having budget forecast in the long term'. FR01 was addressed by design decision D01 'create budget forecast business process'. D01 leads to the creation of the EA element 'budget forecast business process'. The creation of the budget forecast business process triggered the elicitation of two new functional requirements. Stakeholders had to define the frequency for the execution of this business process. This is captured by FR02 'find storing budget estimation frequency' in the business layer of the enterprise. At the same time, the new business process had to be supported by IT applications. This implies FR03 'find IT solution for storing and processing budget' in the application layer of the enterprise. This is an example of a cross-perspective relationship of design rationale.

In the application layer, FR03 was addressed by D03 'use application A' which concerns the EA element 'application A'. The financial application A was already able to support the storage of the financial data, but this information should somehow be entered in the

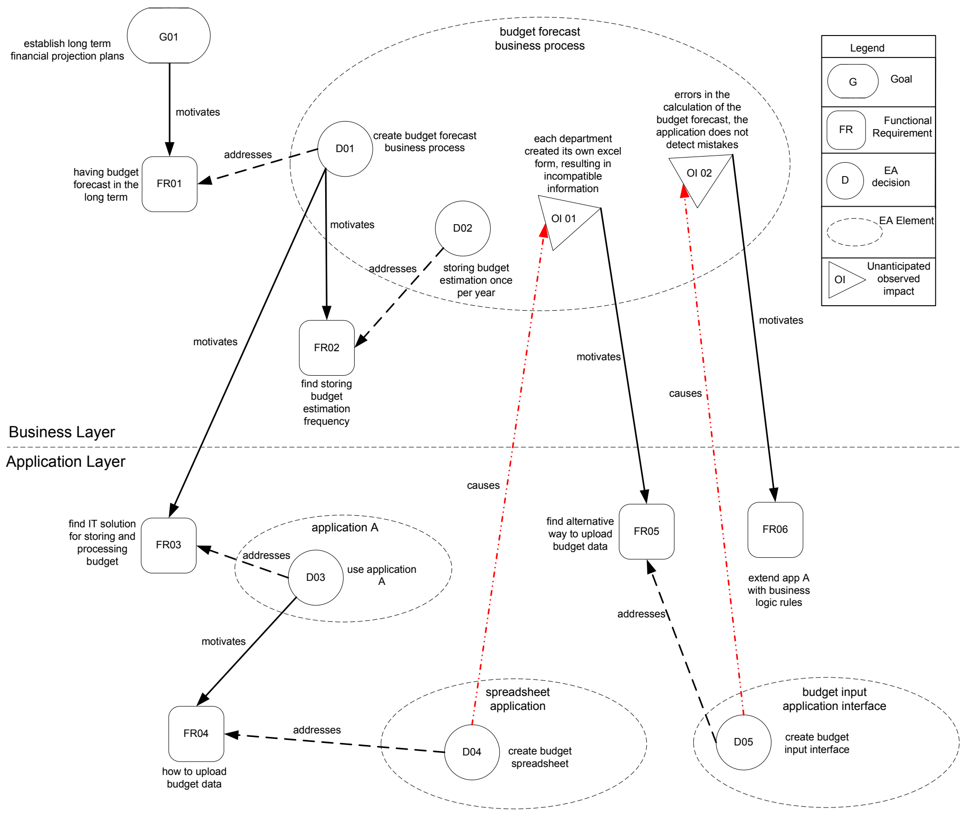

G01
establish long term financial projection plans
motivates
having budget forecast in the long term
FR01
addresses
budget forecast business process
D01
create budget forecast business process
motivates
D02
storing budget estimation once per year
addresses
FR02
find storing budget estimation frequency
each department created its own excel form, resulting in incompatible information
OI 01
errors in the calculation of the budget forecast, the application does not detect mistakes
OI 02
motivates
causes
motivates
Legend
G
Goal
FR
Functional Requirement
D
EA decision
EA Element
OI
Unanticipated observed impact
Business Layer
Application Layer
find IT solution for storing and processing budget
FR03
application A
addresses
D03
use application A
motivates
FR04
how to upload budget data
addresses
spreadsheet application
D04
create budget spreadsheet
causes
find alternative way to upload budget data
FR05
addresses
FR06
extend app A with business logic rules
budget input application interface
D05
create budget input interface

Figure 6.4: Budget forecast design rationalization graph

system. This is captured by FR04 ‘how to upload budget data’. Stakeholders, having in mind again the budget restriction, decided to use a standardized spreadsheet template captured by D04 ‘create budget spreadsheet’ which results in EA element ‘spreadsheet application’. This template was distributed by the financial department to different departments of LuxRTO. The users of the other departments had to fill the spreadsheet template and send it back to the financial department for further processing. This flow of requirements and design decisions comprises the underlying rationale of the EA model of Fig. 6.2.

However, several unanticipated consequences occurred after the execution of these decisions. The use of spreadsheet based templates for the insertion of budget data was problematic. More specifically, the users of each department started modifying the template and the order of the data fields. The financial officer who was receiving the input

Table 6.1: Budget forecast design rationale summarization table

| | |
|---|---|
| **G01** | establish long term financial projection plans |
| **F01** | having budget forecast in the long term |
| **D01** | create budget forecast business process |
| **F02** | find storing budget estimation frequency |
| **D02** | storing budget estimation once per year |
| **FR03** | find IT solution for storing and processing budget |
| **D03** | use application A |
| **FR04** | how to upload budget data |
| **D04** | create budget spreadsheet |
| **OI01** | each department created its own excel form, resulting in incompatible information |
| **FR05** | find alternative way to upload budget data |
| **D05** | create budget input interface |
| **OI02** | errors in the calculation of the budget forecast, the application does not detect mistakes |
| **FR06** | extend app A with business logic rules |

budget data had serious problems with the processing of this information and in turn with the calculation of the budget forecast. The usability of the budget forecast business process was deteriorated. The observed impact (OI01) ‘each department created its own excel form, resulting in incompatible information’ represents this problem. Here, we see how a cross-domain implication of a design decision, an application layer design decision impacts the the performance of the business process.

Another solution to upload budget data should be identified. That was captured by FR05 ‘find alternative way to upload budget data’. Stakeholders decided to upgrade the existing financial application A with a budget input application interface (D05). Decision D05 ‘create budget input interface’ solves the unanticipated consequences of D05. The resulting EA model after these modifications is depicted in Figure 6.3. Despite the fact that this EA model represents the final outcome of this enterprise transformation, there were still some unanticipated problems that were not addressed.

After the incorporation of the budget input application interface, it was found that this module lacks business logic error checking functionality during the data entry of the budget input. The problem is depicted in OI02 ‘errors in the calculation of the budget forecast’. The application does not detect mistakes’. As a result, users of this application who are not familiar with financial parameters can create serious mistakes during the calculation of budget forecast. FR06 ‘extend the application with business logic rules’ was also captured by EA Anamnesis. Despite the fact that stakeholders were aware of the problem, they were not able to take additional decisions. This is because of the upcoming merge of LuxRTO with another organization (see Section 6.4.3). FR06 remained unaddressed.

### 6.4.2 Analysis per design decision

In this subsection, we elaborate on a specific EA design decision from our case study which shows how EA Anamnesis makes explicit the considered requirements. Moreover, we show the role of requirements for the selection of a specific design decision, their compliance with the given principles and what were the rejected alternatives.

We examine design decision D03 ‘upgrade Application A’ in the application layer of the enterprise, which is summarized in Table 6.2. The decision concerns the selection of an appropriate application system for the support of the newly introduced intermediary business process. The IT architect was between two solutions, ‘upgrade Application A’ and ‘COTS application’. For his decision making problem, he took into account the ‘scalability’ of the application as well as its ‘cost’.

Figure 6.5 provides a graphical overview of the decision making process for D03. Here we can also see the role of EA principle ‘EA is built on scalable components’ for the elicitation of non-functional requirements. NFR01 ‘scalability’ was selected in order to comply with the given principle. This is signified by the relationship ‘complies with’ between the NFR01 and the principle. As such, we captured the motivational reason behind the selection of NFR01.

Table 6.2: EA design decision 03 details

| | |
|---|---|
| **Decision:** | use application A |
| **Functional requirement:** | find IT solution for storing and processing budget |
| **EA perspective:** | application layer |
| **Alternatives:** | AL01: COTS application<br>AL02: use application A |
| **Non-functional requirements:** | scalability, minimize cost |

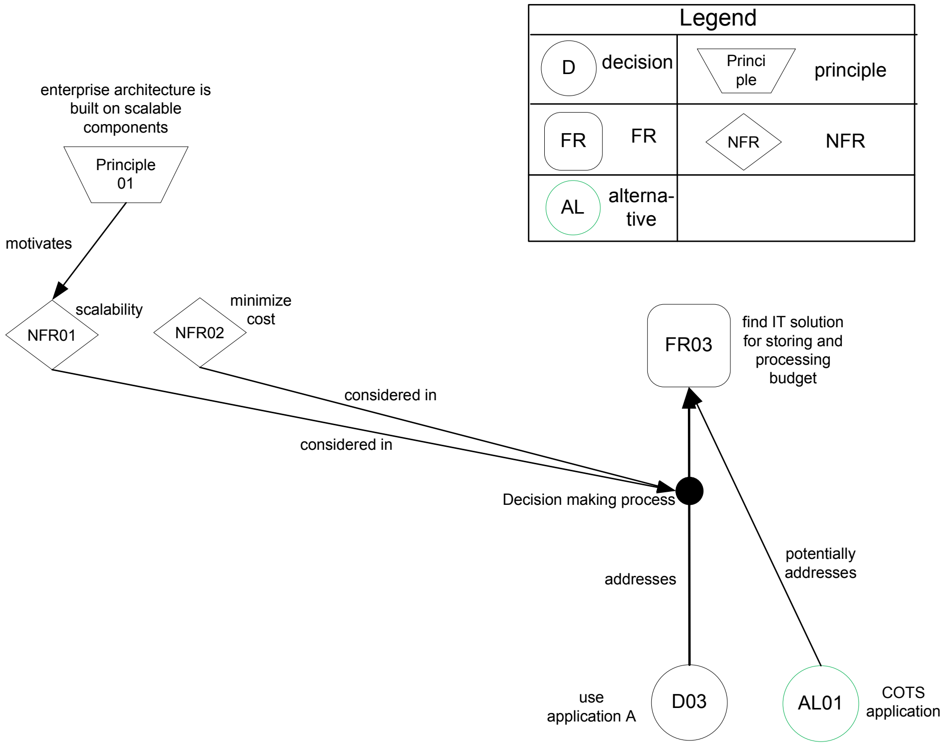


Figure 6.5: Decision making process of D03

## 6.4.3 Replaying decision making processes

Our analysis continues by zooming into the decision making process for design decision D03 'upgrade Application A'. We applied the notion of decision making processes and strategies in order to capture the reasoning of the IT architect for D03. We now replay that process.

Figure 6.6 provides the visualization of the decision making process for EA decision 'Upgrade Application A' (D03). As we discussed above, D03 addressed FR03 'find IT solution for storing and processing budget'. An alternative that potentially could address FR03 was AL01 'COTS Application'.

So, what was the reason that stakeholders chose the upgrade of the existing financial application (D03) instead of AL01 'COTS Application'? By interviewing the stakeholders we understood and captured the context which influenced their decision making: during the execution of the enterprise transformation another high level decision from the

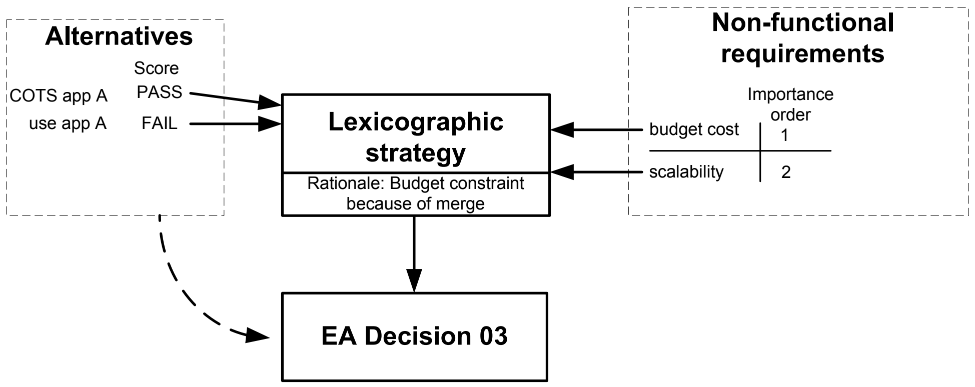


Figure 6.6: Decision making strategies of D03

Luxembourgish government had to be applied in the organization. The government decided that LuxRTO had to be merged with another national Research and Technology Organization. This imposed the need for serious changes in the organizational structure, since some departments of LuxRTO had overlapping roles with departments of the other organization. Moreover, new business models should be defined based on the exchange of research expertise of research groups.

The upcoming merge of the organization posed some serious design challenges on the involved stakeholders of the budget estimation business process. Their initial plan was to acquire 'COTS application A' that would be possible to also support the future needs of the organization. This plan was in accordance with the architecture principle 01 of LuxRTO 'EA is built on scalable components'. At the same time, it was not clear how the financial departments and business processes would be merged. Therefore, the risk of wasting budget for significant business and IT development was high.

Consequently, despite the fact that the initial plan of the stakeholders was the acquisition of the 'COTS Application', the upcoming merge led to a minimization of budget spending. This led in turn stakeholders to the decision of upgrading the in-house applications.

By using EA Anamnesis we captured the aforementioned situation: Two non-functional requirements were considered in the decision making process for the application selection, NFR01 'scalability' which complied with Principle 01 and NFR02 'minimize cost'. On the one hand, 'COTS application A' was satisfying NFR01, but not NFR02. On the other hand, 'Upgrade application A' was not satisfying NFR01, but was satisfying NFR02. In the specific decision making context stakeholders decided that they should prioritize and decide based on NFR02 'minimize cost', Therefore they rejected AL01 'COTS Application'. The decision making strategy for this kind of prioritization is a 'lexicographic' strategy.

Without rationalization, the above reasons behind the architecture designs of Figures 6.2 and 6.3 would have remained implicit. Stakeholders or even newcomers to the enterprise who want to analyze or provide justification for past decisions to management or other stakeholders, can use EA Anamnesis in order to understand the role of principles and requirements to the decision making process.

For example, if a newcomer is asked about the alignment of the 'Upgrade application A' with the principle 'EA is built on scalable components', he will be able to justify that this application is not aligned with the given principle since his predecessors had to compromise with a low budget solution. Another example would be the explication of the negative observed impact of diverging spreadsheets as a result of the introduction of spreadsheet template. As a result, this negative observed impact can be anticipated for future similar decisions.

## 6.5 Evaluation

As stated in Section 6.1, the design rationale was captured together with the involved stakeholders. This Section presents our reflections on the quality characteristics that were discussed in Section 1.4.3. Furthermore, in Section 6.6, we enlist some lessons learned by the application of EA Anamnesis to LuxRTO.

We enlist our findings with regard to the quality characteristics:

- *Domain appropriateness:*

  This quality characteristic deals with the ability of EA Anamnesis to express design rationale in an EA context. Practitioners were able to capture design rationale, such as the cross-domain implications of their decision making, the design problem that initiated the design process and the reasoning behind their design decisions. Moreover, stakeholders were able to have insight on the decision making of other stakeholders on the enterprise, as well as to the consequences of their design decisions.

- *Participant language knowledge appropriateness:*

  This quality characteristic deals with practitioner's level of understanding of EA Anamnesis terminology. We started by demonstrating EA Anamnesis and its concepts to practitioners. This was done by giving concrete descriptions per concepts and by illustrating the approach using the ArchiSurance example of Section 4.3. Practitioners understood the idea of EA Anamnesis and they could reflect on its usefulness for design rationalization. By doing so, we ensured that practitioners had a proper understanding of the terminology before the actual capturing of design rationale. Afterwards, the capturing process we also observed how EA Anamnesis could support the transfer of knowledge between practitioners of various architecture domains. Practitioners were able to understand, based on the terminology of EA Anamnesis, the design rationale of another domain.

- *Knowledge externalizability appropriateness:*

  This quality characteristic deals with the level that the knowledge of practitioners was reflected by EA Anamnesis. By demonstrating EA Anamnesis to practitioners, we educated and made them understand the notion of decision making strategies. Implicitly, they were using decision making processes. However, the stakeholders did so *without being aware of that.* The awareness of different types of decision making strategies enabled them to better structure and analyze the decision problem. This means that they were able to explicitly describe how they decided (e.g. in terms of applied decision making strategy) for a certain decision problem and what evaluation parameters they used.

## 6.6 Lessons learned

- *Lesson 1: Stakeholders use simple selection processes, or decide without examining alternatives.*

  EA Anamnesis is designed to cover a variety of decision making strategies, compensatory or non compensatory. Our findings from this case study is that stakeholders use simple techniques to eliminate alternatives from their choice set. For example, in Section 6.1, we have seen that the the decision 'upgrade Application A' was selected based on its cost without further trade-off analysis w.r.t. other qualities. Even more, sometimes stakeholders addressed a functional requirement without examining alternative choices. The main reason for not considering alternatives is that experienced stakeholders make decisions based on previous experiences from similar cases.

  Compensatory strategies like MAUT were not used for any of the captured design decisions in the LuxRTO case study. We argue that this finding actually promotes the applicability of EA Anamnesis in practice, since it is easier in terms of capturing effort for the designer, to capture the underlying decision making strategies.

- *Lesson 2: Decision making can be ongoing, with unaddressed functional requirements*

  As can be observed from FR06 'extend the application with business logic rules' (Figure 6.4) some functional requirements were not resolved. Reasons such as lack of resources (budget, time), sometimes prevent designers from addressing the requirements.

  This is an important finding for design rationalization, since it shows that decision making is an ongoing activity. The awareness of unaddressed functional requirements gives the ability of better justification of EA designs. For example, by capturing unaddressed functional requirements a stakeholder of the RTO organization can justify a lacking usability of the budget forecast business process due to a lack of business logic control mechanisms in the application layer. This information could not be provided by the EA design.

## 6.7 Conclusion

In this chapter, we presented the application of EA Anamnesis to a real world EA transformation. By conducting case study research, we evaluated the capability of EA Anamnesis to capture and represent adequately design rationale. EA Anamnesis captures sufficiently design rationales for EA. Furthermore, during the application of EA Anamnesis, some important lessons were learned from this case.

CHAPTER 7

# Applying EA Anamnesis in a Greek e-Government Organization

In this chapter we apply EA Anamnesis to a Greek e-government organization and we evaluate the ability of EA Anamnesis to capture and represent design rationale. We used EA Anamnesis in order to capture the design rationale behind the incorporation of a centralized monitoring system for pension salaries in Greece. Our case study shows that design rationale was sufficiently captured by EA Anamnesis. Moreover, we found that it raises awareness of problematic situations.

The chapter is structured as follows. Section 7.1 presents the objectives and the case study protocol that we followed. Section 7.2 introduces the case study context and Section 7.3 describes the enterprise transformation. Section 7.4 presents the captured design rationale information based on EA Anamnesis and Section 7.6 discusses the lessons learned. Finally, Section 7.7 concludes.

## 7.1 Objectives and case setup

We followed the same methodology as presented in Chapter 6. Our main objective is to review to what extent EA Anamnesis is able to capture design rationale in the context of a *real life* enterprise transformation. In this case, we study one particular transformation: the incorporation of a centralized monitoring system for pension salaries in Greece. We organized interviews with key stakeholders that participated in the decision making during the transformation. More specifically, we interviewed the project manager of the system, the software developer and an IT architect. Moreover, the stakeholders provided us with sample data files, documentation, architecture diagrams and presentations. We used ArchiMate to model the various states of the enterprise and we ensured after having demonstrated the EA models to the relevant stakeholders that the EA models were representative.

We started our case study by presenting the EA Anamnesis to the stakeholders. We explained the goals and challenges of our case study, and we illustrated our conceptual framework using the example case of Section 4.3.1. This example case helped the stakeholders to understand EA Anamnesis.

After the presentation of EA Anamnesis, we conducted a collaborative modeling exercise with three stakeholders. The goal of this exercise was to see to what extent EA Anamnesis was able to capture the design rationale of this transformation. Furthermore we also identified the perception of stakeholders regarding the concepts of EA Anamnesis.

Similarly, with the previous case study (Chapter 6), our goal was to evaluate EA Anamnesis in terms of its *domain appropriateness*, *participant language knowledge appropriateness* and *knowledge externalizability appropriateness.*

## 7.2 A fragmented social security landscape

The financial crisis that started in the late 2000 forced the Greek government to implement, in a very short period of time, major structural reforms. One of the most important reforms was the establishment of the national register of pensioners and pension salaries.

In the past few years social insurance policies were developed in a fragmented way (OECD 2002) through the establishment of social insurance institutions per different profession category. For example, doctors have their own social security institution, engineers a different one etcetera. Greece had by the end of 2002 170 different social security institutions (OECD 2002). In the following years, Greek governments initiated a series of mergers. As a result, the number of the institutions was significantly reduced. However, the number of these institutions is still high compared to the average number of institutions in other EU member states. Furthermore, the merging was not executed deep in the organizational structure of the institutions. As a result there are cases where these merged social security institutions have departments with overlapping activities and information systems.

Due to the high number of social security institutions and the lack of a standardized process for the pension salary calculation and payment in each of these institutions, there was a lack of *central monitoring* of the aggregate amount of budget that was spent nationally on pensions. On the one hand, the government was not able to make projections regarding the spending on pensions. On the other hand, there were rare cases where citizens were cheating the system with a variety of ways (receiving double allowances etcetera).

The situation was getting even more problematic when someone had worked in two or more different types of professions during their career. For example, someone that had worked 10 years as a professional driver and the rest of their career as an employee in a company had to wait for more than 2 years to have an accurate estimation and award of their pension salary. This was due to the fact that different social security institutions had to exchange, in paper, the social security information for this person and then make a common decision regarding the amount of pension that each institution had to pay. Moreover, the pension payment was fragmented among the institutions. Each institution was sending separate payment notices per pensioner to the bank.

To address these issues, the Greek government established a pension payment and report centralized system. The planning, design and operation of this project was assigned to the Social Security e-government center of the ministry of Labor (e-gov center). The agency had to deliver in a short period of time a system that would provide a unified report of the pension salaries spent by the Greek government.

## 7.3 Enterprise Transformation

We now provide the description of the enterprise transformation by means of EA models. In our analysis, we used the ArchiMate modeling language. For economy of space we do not provide the EA models for every social security institution. We use instead the abbreviation 'SII01' to refer to the first social security institution of our study and 'SSI...n' to signify that we have more than one institutions in our analysis. By doing so, we reduce the complexity of the EA models. Moreover, for simplicity reasons, we do not include in our models architecture elements that have a supportive role in the current setup like network infrastructure, etcetera.

### 7.3.1 Baseline architecture

Figure 7.1 presents the EA model of the baseline architecture before the incorporation of the pension report system. Each of the institutions has, independent from the others, the business role 'Pension administrator'. This role is supported by the business process 'Pension calculation SSI' in every institution.

The institution calculates the amount of pension salary based on (1) the years that each citizen was insured in the specific institution and (2) the special legal regulations that are applied for each profession. During the lifetime of a citizen's pension, several salary calculations are performed. This is because the amount of salary has to be adapted to several factors like inflation, new regulations, etcetera.

After the calculation of the pension salary, a salary invoice is issued and forwarded to the pensioner. Moreover, the pension salary information is forwarded through the business object 'Pensioner's payment data' to the business process 'Pension payment SSI' in order to execute the payment order of a pension salary through a banking system. It is worthwhile to mention again that due to the high number of social security institutions, there are cases that a pensioner receives pension salaries from more than one institution. This situation is depicted in the EA model by the multiple links between the citizen's role 'Receives pension salary' and the business services 'Pension salary invoice SSI 01', 'Pension payment SSI 01', 'Pension salary invoice SSI...n' and 'Pension payment SSI...n'. In other words, a pensioner, instead of receiving an aggregate pension salary, was still receiving separately parts of pension salaries from the different social security institutions.

On the application layer, each of these social security institutions has its own application services and systems that support the aforementioned business processes. The 'Pension calculation application SSI 01' incorporates the business logic and the legal regulations for the calculation of pension salaries. The pension salary applications are realized by the technology layer elements 'Application server SSI 01' and 'Database server SSI 01'.

As we can see from the EA model, the EAs of the social security institutions were fragmented. The Greek government did not have any centralized way to monitor and control the spending of pensions.

### 7.3.2 Target architecture

As discussed in Section 7.2, the Greek government assigned the responsibility of the national pension payment and report system to the e-government center. The main goal of this project was the calculation and report of the spending in pensions on a monthly basis, the enforcements of cut on the aggregate amount of salary per pensioner, and

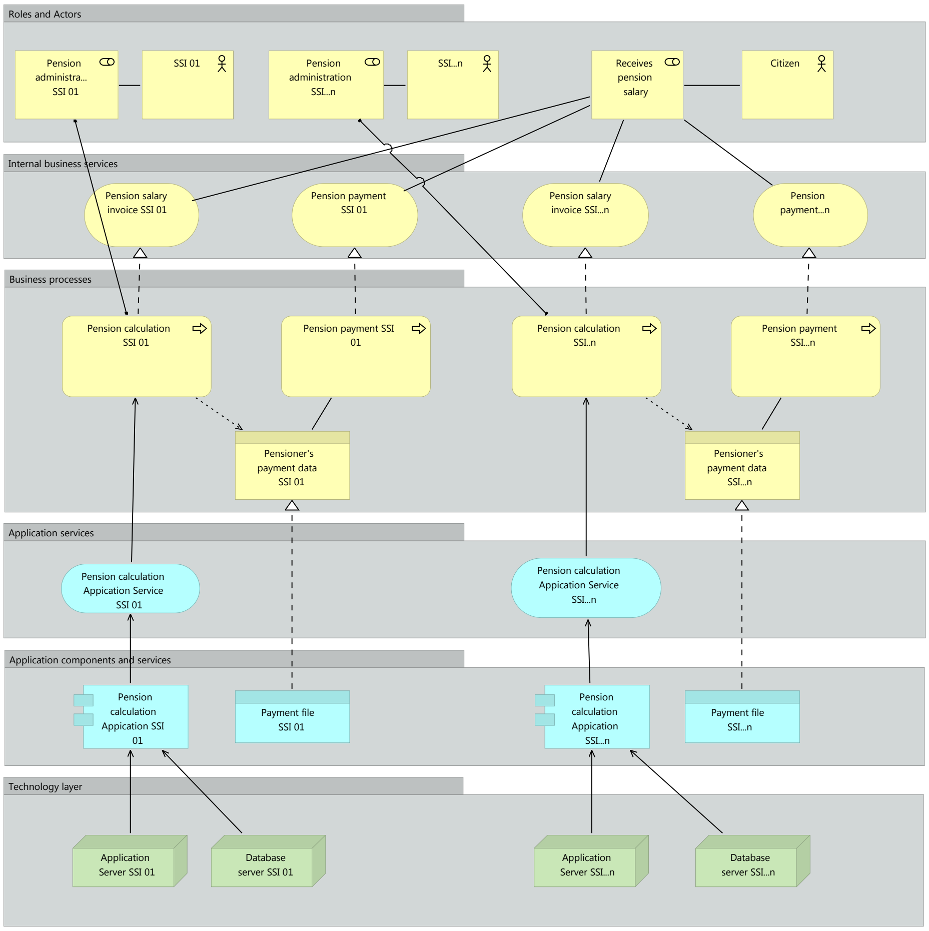

Roles and Actors
Pension administra... SSI 01
SSI 01
Pension administration SSI...n
SSI...n
Receives pension salary
Citizen
Internal business services
Pension salary invoice SSI 01
Pension payment SSI 01
Pension salary invoice SSI...n
Pension payment...n
Business processes
Pension calculation SSI 01
Pension payment SSI 01
Pension calculation SSI...n
Pension payment SSI...n
Pensioner's payment data SSI 01
Pensioner's payment data SSI...n
Application services
Pension calculation Appication Service SSI 01
Pension calculation Appication Service SSI...n
Application components and services
Pension calculation Appication SSI 01
Payment file SSI 01
Pension calculation Appication SSI...n
Payment file SSI...n
Technology layer
Application Server SSI 01
Database server SSI 01
Application Server SSI...n
Database server SSI...n

Figure 7.1: Baseline architecture

the apportionment and reporting of the pension salary to the pensioner's social security institution(s). As the first step, various social security institutions defined a common reference point for pensioners by using the national security number (i.e., a unique number per citizen) of each pensioner. Before the development of this project, each social security institution was using its own social numbers. As a result, there were cases where a person had several social numbers. After a period of six months the social security institutions adopted and migrated their records to the national social security number. By doing so, the e-gov center was able to collect data from the different institutions and make mappings across the different pensioners' records.

In the next part, we provide two alternative EA scenarios that were considered as solutions for the national pension payment and report registry. The first scenario was the 'Fully consolidated architecture' and the second was the 'Aggregation of social security institutions pension reports'. The two scenarios have commonalities only in the provision of the requested business services. Below, we provide description of the two alternative architectures.

### 7.3.3 Scenario 1 - Fully consolidated architecture

Figure 7.2 presents a candidate architecture scenario where the national payment report business service is provided by the unification of the individual business processes and information systems. The business process 'Unified pension calculation' which realizes the business service 'Unified pension salary invoice' would be created by establishing a common business process for the calculation of the pension salaries. Moreover, the 'Unified pension payments' business process would be created by the integration of the individual Pension payments' business processes of the social security institutions. Through this integration, the e-gov center would be able to provide as well the 'National payment report' business service to the government. In other words, the national pension payment and report project would be used as an opportunity for the unification of the individual business processes among the various social security institutions and this implies that the e-gov center would be the responsible authority not only for the reporting of pension salaries, but also for the calculation and payment of salaries.

On the information systems side, the aforementioned business processes would be supported by the corresponding IT elements. One of the biggest challenges in this transformation scenario was the migration of the individual pension calculation applications into a new 'Migrated Pension Application' that would incorporate a core business logic for the calculation of the pension salaries. In addition, it would take into account the different pension calculation specificities among the various social security institutions. The application architecture team should coordinate a migration procedure where the characteristics of the individual pension calculation application per institution would be taken into account. In parallel, the application architecture team should coordinate a data migration procedure in order to integrate the pensioners' data into a common database based on the national social security number. The e-gov center should also provide 'Application servers' that would host the 'Migrated pension application' and 'Database servers' for the migrated pensioners' data.

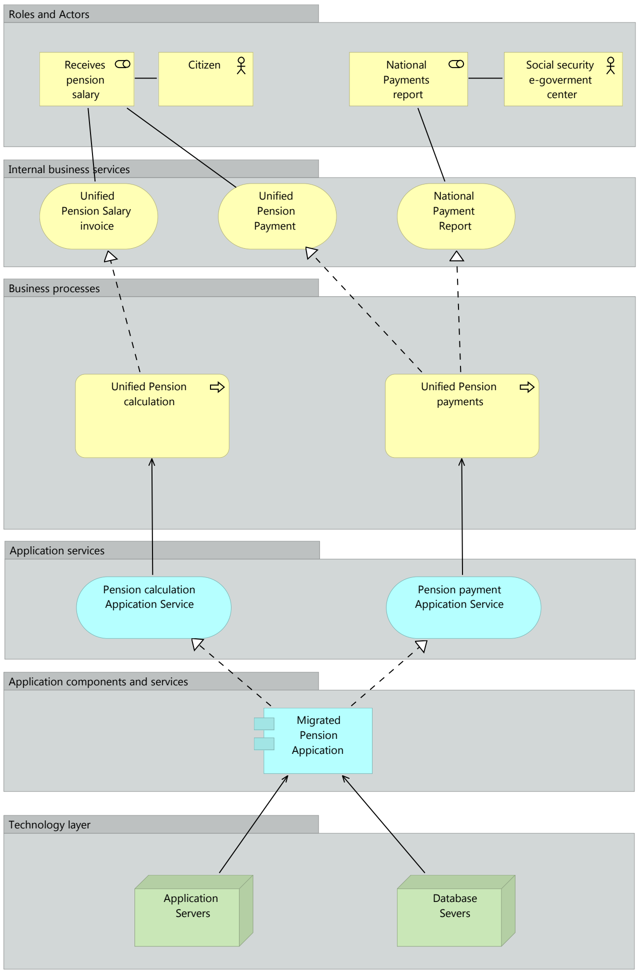

Roles and Actors
Receives pension salary
Citizen
National Payments report
Social security e-goverment center
Internal business services
Unified Pension Salary invoice
Unified Pension Payment
National Payment Report
Business processes
Unified Pension calculation
Unified Pension payments
Application services
Pension calculation Appication Service
Pension payment Appication Service
Application components and services
Migrated Pension Appication
Technology layer
Application Servers
Database Severs

Figure 7.2: Fully consolidated architecture

### 7.3.4 Scenario 2 - Aggregation of pension payments files

Figure 7.3 presents the alternative EA scenario which was actually selected by the architecture team. At first glance, we can see that the business services 'Unified pension salary invoice', 'Unified pension payment' and 'national payment report' are provided in a completely different way. The main difference is that the pension salary calculations are still kept under the authority of the individual social security institutions, while only the pension payments and national payment reports below the responsibility of the e-gov center.

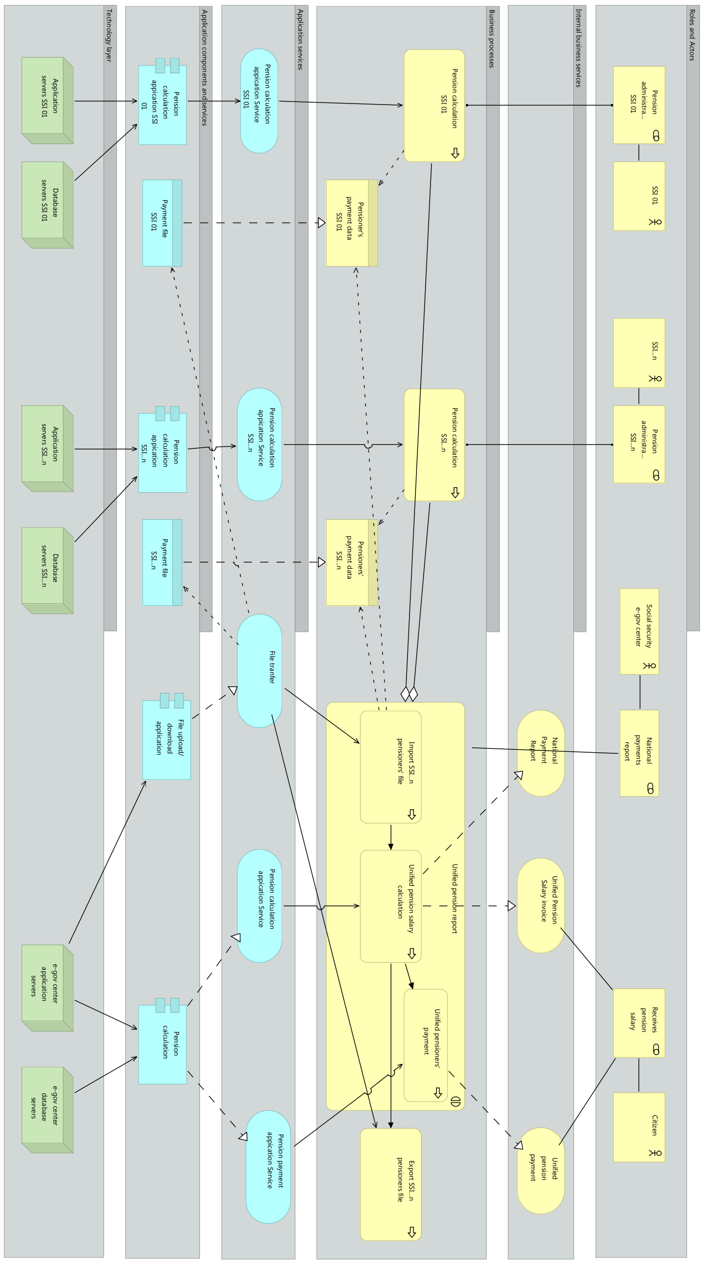

Technology layer
Application components and services
Application services
Business processes
Internal business services
Roles and Actors
Application servers SSI 01
Database servers SSI 01
Application servers SSI...n
Database servers SSI...n
e-gov center application servers
e-gov center database servers
Pension calculation application SSI 01
Payment file SSI 01
Pension calculation application SSI...n
Payment file SSI...n
File upload/ download application
Pension calculation
Pension calculation application Service SSI 01
Pension calculation application Service SSI...n
File tranfer
Pension calculation application Service
Pension payment application Service
Pension calculation SSI 01
Pensioner's payment data SSI 01
Pension calculation SSI...n
Pensioners' payment data SSI...n
Unified pension report
Import SSI...n pensioners file
Unified pension salary calculation
Unified pensioners' payment
Export SSI...n pensioners file
National Payment Report
Unified Pension Salary invoice
Unified pension payment
Pension administra... SSI 01
SSI 01
SSI...n
Pension administra... SSI...n
Social security e-gov center
National payments report
Receives pension salary
Citizen

Figure 7.3: National pension report by aggregation of pension payments files

More specifically, we can observe that the business process ‘Pension calculation’ is still maintained in every social security institution. Moreover, that each social institution has to provide a business object ‘Pension payment data’ to the e-gov center. Therefore, the social security institutions not only have to maintain their existing information systems (‘Pension calculation application’, ‘Application servers’ and ‘Database servers’), but they have also to send ‘Pensions payment data’ business objects to the e-gov center. In other words, this indicates that each institution has established next to the ‘Pension calculation’ business process a new task that sends on a monthly basis the pensioners’ payment data to the e-gov center. The business object ‘Pensioners’ payment data’ is realized by the use of the standardized data object ‘Payment file’. This means that the information between the social security institutions and the e-gov center is exchanged through standardized data files.

Furthermore, the e-gov center has established the business interaction ‘Unified pension report’ which acts as an aggegator of the ‘Pensioners’ payment data’ business objects. This business interaction consists of four different business processes. The first, ‘Import SSI...n pensioners’ file’ is the business process with the responsibility of collecting, on a monthly basis, the payments data files from the various social security institutions. As we mentioned before, there are cases where citizens have pension rights from more than one institution. Therefore, one of the most crucial tasks of this business process is the provision of a unified data file which has as reference key the pensioners’ social security number and which contains the information from the various social security institutions that correspond to this social security number. As we can see from the EA model, this exchange of information is done by using the ‘File transfer’ service of a specialized ‘File upload/download application’.

The corresponding business process is the ‘Unified pension salary calculation’. This business process is responsible for the calculation of the salary per pensioner by taking into account the new government measures regarding the maximum amount of pension salaries in the country. The business logic regarding the calculation of the pension salaries is realized by the ‘Pension calculation’ application. The aggregated pension salaries information is stored in the ‘e-gov center database servers’.

As a final step, the e-gov center runs the ‘Export SSI...n pensioners’ file’ business process where the social institutions are still in charge of their ‘Pension administration’ business role and they provide to e-gov center (via the ‘Import SSI...n pensioners’ file’ business process) information regarding the amount of their pension salaries spending. Moreover, the e-gov center applies some government measures which actually influence the total amount of pension salary per pensioner. Due to the fact that the pension calculation is scattered among the various social security institutions, the e-gov center through the business process ‘Export SSI...n pensioners’ file’ informs each social security institution about its actual spending on pension salaries. This is done by using another type of payment file. For simplicity reasons we leave out the architectural description of this type of data exchange.

Finally, the business process ‘Unified pensioners’ payment’ is responsible for executing the payment orders of the aggregated pension salaries to the pensioners’ bank.

## 7.4 Rationalizing the EA design

In the previous section we discussed two different EA scenarios for the national pension payment and report project. We also mentioned that the ‘National pension report by aggregation of pension payments files’ scenario was finally selected by the stakeholders team. By just observing the EA scenarios ‘Scenario 1: Fully consolidated architecture’ seems preferred in terms of simplicity and the number of EA elements. Contrariwise, in scenario 2, we can see that each social institution maintains individually the ‘Pension calculation business process’ which means that institutions spend a significant amount of budget on employees that are actually executing a quite similar task. Moreover, each of these institutions is maintaining their own information systems which implies additional cost for IT systems and their maintenance.

The examination of the EA models triggers questions regarding their rationalization. For example, what made the stakeholders team decide for a more complicated architecture, which factors played a role in the decision making process etcetera. We applied the EA Anamnesis to capture design rationale behind the EA models. In the context of this case study, we provide the rationalization for three critical design decisions which impacted the EA design. Similar to Section 6.4 we provide the rationalization information by means of a design rationalization graph and then we further zoom into the specific design decisions.

### 7.4.1 Design rationalization graph

The design rationalization graph of Figure 7.4 provides a holistic overview of the captured design rationales and their relationship. We start our evaluation by capturing the strategic goal SG01 ‘provide national pension report to government’. The strategic goal SG01 motivated’ the functional requirement FR01 ‘establish national report’. FR01 was addressed by design decision D01 ‘establish business interaction unified pension report’. To do so stakeholders executed a decision making process in which they rejected the alternative solution ‘unified pension payments’. The rationale behind this selection is provided

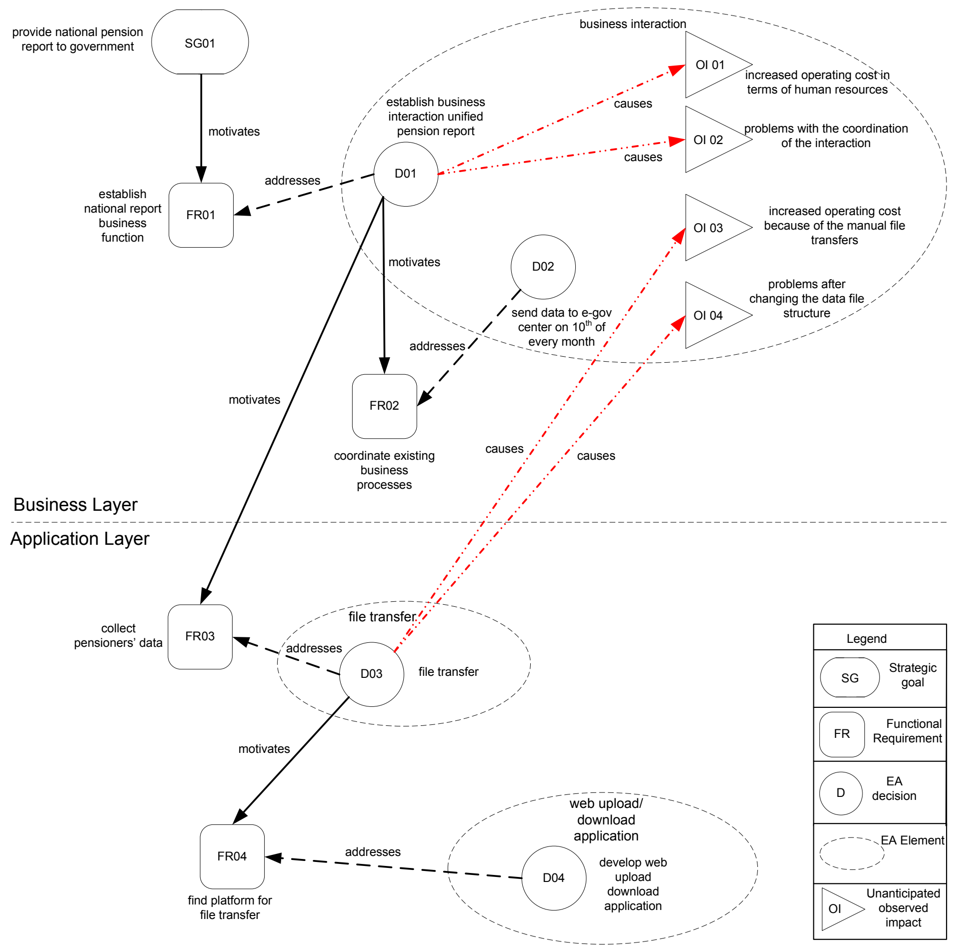

provide national pension report to government
SG01
motivates
establish national report business function
FR01
addresses
business interaction
establish business interaction unified pension report
D01
causes
OI 01
increased operating cost in terms of human resources
causes
OI 02
problems with the coordination of the interaction
OI 03
increased operating cost because of the manual file transfers
OI 04
problems after changing the data file structure
motivates
D02
send data to e-gov center on 10th of every month
addresses
FR02
coordinate existing business processes
motivates
causes
causes
Business Layer
Application Layer
collect pensioners' data
FR03
file transfer
addresses
D03
file transfer
motivates
FR04
find platform for file transfer
addresses
web upload/ download application
D04
develop web upload download application
Legend
SG
Strategic goal
FR
Functional Requirement
D
EA decision
EA Element
OI
Unanticipated observed impact

Figure 7.4: National pension payment design rationalization graph

in the decision making process analysis for D01 (Section 7.4.3) and visualized in Figure 7.5.

Our analysis continues with the elicitation of two functional requirements FR02 'coordinate existing business processes' and FR03 'collect pensioners' data'. Since stakeholders decided to start collecting pensioners' data from the various social security institutions, they had to find a way of doing that. They first had to decide on a business level for the frequency of the execution of the business interaction between the egov center and the various social security institutions. D01 motivated FR02 'coordinate existing business processes' which was addressed by decision D02 'send data to e-gov center at 10th of every month'. In other words social security institutions had to send to the e-gov center detailed pension reports at the 10th of every month.

Moreover, stakeholders had to support the exchange of the information by electronic means. FR03 'collect pensioners data ' was also motivated by D01. This is a cross perspective relationship between a decision on the business layer of the enterprise and a requirement on the application layer. Three alternative solutions were considered by stakeholders. The decision making process is visualized in Figure 7.7. FR03 was addressed by D03 'file transfer'. The exchange of information would be based on files.

Finally, D03 motivated FR04 'find platform for file transfer'. At this point, stakeholders had to find the appropriate means for the exchange of files. After another decision making process (depicted in Figure 7.9), they decided to develop a web application that would facilitate users to download and upload the data files. D04 'develop upload/download application' addressed FR04.

### 7.4.2 Unanticipated observed impacts of design decisions

Next, we analyze the unanticipated observed impacts of design decisions across the EA. For the national pension payment and report case we captured four observed impacts. Figure 7.4 presents these observed impacts.

#### Unanticipated observed impact OI01:

The first one OI01 'increased operating cost in terms of human resources' originates from design decision D01 'establish business interaction unified pension report'. The observed impact concerns the same EA perspective (architecture layer) as the design decision. The establishment of the business interaction among the various social security institutions induced an increased cost for the operation of this interaction from every institution. Each institution had to assign to its employees the task of processing, exporting and sending the pensioners' data files to the e-gov center. Due to the fragmented social security landscape of the EA the number of people doing for the same kind of work was higher than the number of the participating social security institutions.

#### Unanticipated observed impact OI02:

The second observed impact OI02 'problems with the coordination of the interaction' concerns the coordination of the individual business processes. Because of the large number of the institutions there are cases where the stakeholders forget to send the data files within the specified time limits. This affects the effectiveness of the business interaction because the e-gov center requires the input from all social security institutions within the specified time limit. If one of the input is missing, then the unified pension calculation can not be executed. It is a quite common case that certain employees of the e-gov center has to call the relevant employees of the institutions that are delayed to send as soon as possible the data files.

**Unanticipated observed impact OI03:**

The third observed impact OI03 'increased operating cost because of the manual file transfers' concerns again the productivity of the business interaction. As we can see from Figure 7.4, we have a cross-domain relationship between the design decision D03 'file transfer' (application layer) and the observed impact (business layer). Capturing this cross-domain consequence makes explicit that an IT decision can still introduce problems on the business side. The problem here is that the files are uploaded and downloaded manually on both sides, the social security institutions and the e-gov center. That means additional cost for the operation of the business interaction.

**Unanticipated observed impact OI04:**

Finally, OI04 'problems after changing the data file structure' concerns a problematic situation regarding the consistency of the structure of the data file that institutions are using to send information towards the e-gov center. More specifically, since the start of the national pension payment and report project there are cases that the e-gov center has to incorporate new directives regarding the unified calculation of the pension salaries. This sometimes implies that the national pension repository of the e-gov center has to be supplied with additional information from the institutions side and consequently the structure of the data file has to be changed. Due to the large number of social security institutions, there were problems with the adoption of the new data file structure. Some of them were sending the information with the new version and some with outdated versions. This produced again additional administrative cost for operation of the business interaction.

### 7.4.3 Analysis per design decision

In this Section we further analyze design decisions. We capture the alternatives and requirements that were considered during the decision making process and the decision making strategies that were used for the evaluation.

**D01: establish business interaction unified pension report**

We examine design decision D01 ‘establish business interaction unified pension report’ on the business layer of the enterprise. The decision details are summarized in Table 7.1 and visualized in Figure 7.5.

Stakeholders had to satisfy the functional requirement FR01 ‘establish national report business function’. Two alternatives were considered, the ‘establish business interaction unified pension report’, which is the executed decision D01 and the AL01 ‘unified pension payments’.

Stakeholders started their decision making process by eliciting the non-functional requirements. The organization was operating under high pressure from the government for immediate delivery of social security IT projects. The first non-functional requirement had to comply with the Principle P01 ‘fast implementation’. Non-functional requirement NFR01 ‘implementation time’ was elicited. Moreover, NFR02 ‘running cost’ and NFR03 ‘running efficiency’ were considered for the decision making process. Due to the fact that they had to deliver the project in a short time frame, NFR01 ‘implementation time’ was the non-functional requirement with the highest importance. The stakeholders captured the order of non-functional requirements by using a ‘Lexicographic’ non compensatory decision making strategy. Despite the fact that the quality characteristics of the alternative AL01 ‘unified pension payments’ were considered by stakeholders as better, D01 ‘establish business interaction unified pension report’ was selected. Figure 7.6 zooms further in the decision making process and presents the decision making strategy that was used.

Table 7.1: EA design decision 01 details

| | |
|---|---|
| **Decision:** | D01: establish business interaction unified pension report |
| **Functional requirement:** | FR01: establish national report business function |
| **EA perspective:** | business layer |
| **Alternatives:** | AL01: unified pension payments<br>AL02: establish business interaction unified pension report |
| **Non-functional requirements:** | NFR01: implementation time<br>NFR02: running cost<br>NFR03: running efficiency |

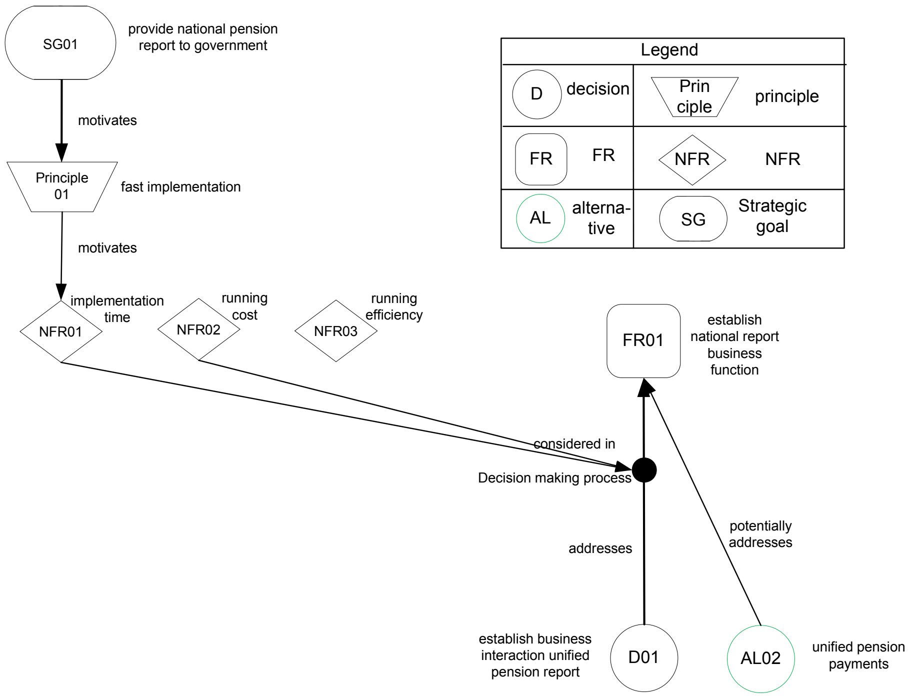


Figure 7.5: Decision making process of D01

## D03: file transfer

Table 7.2 summarizes and Figure 7.7 visualizes the decision making process of D03. The stakeholders had to select the IT service that would support the business collaboration

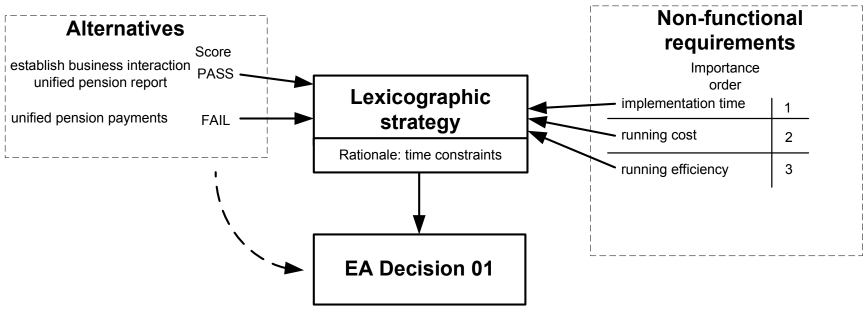


Figure 7.6: Decision making strategy of D01

Table 7.2: EA design decision 03 details

| | |
|---|---|
| **Decision:** | D03: file transfer |
| **Functional requirement:** | FR03: collect pensioners' data |
| **EA perspective:** | Application layer |
| **Alternatives:** | AL03: file transfer<br>AL04: shared database<br>AL05: remote procedure invocation |
| **Non-functional requirements:** | NFR01: implementation time<br>NFR04: automated data integration<br>NFR05: interoperability |

between the e-gov center and the social security institutions. This captured by FR03

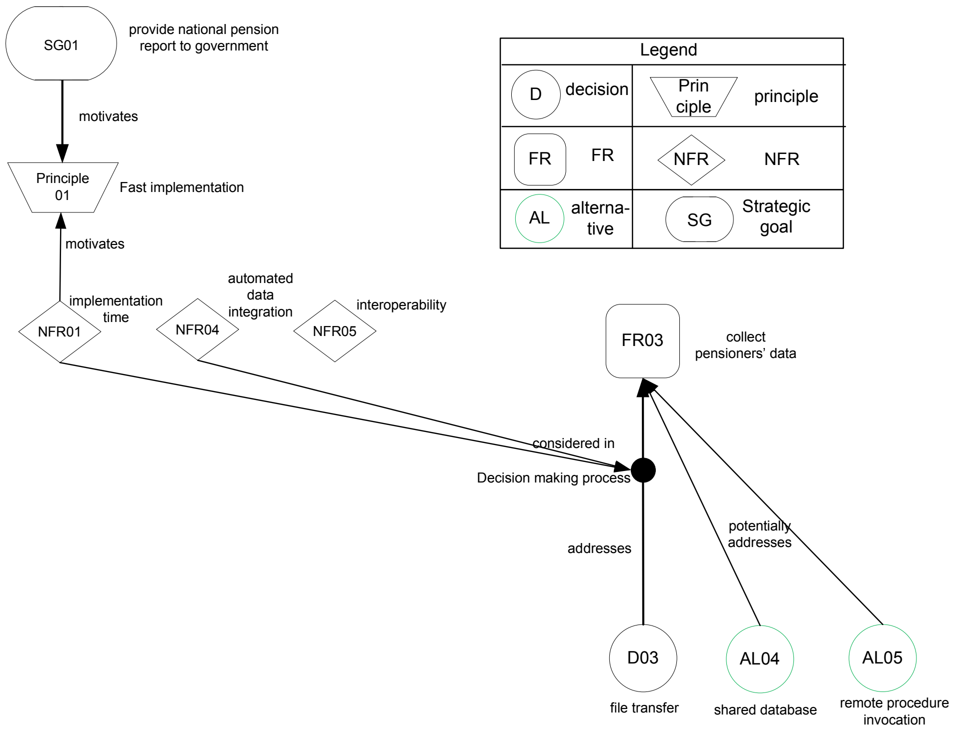


Figure 7.7: Decision making process of D03

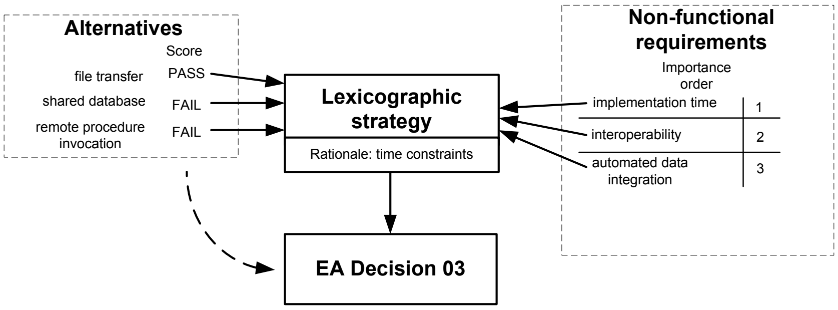


Figure 7.8: Decision making strategy of D03

'collect pensioners' data'. Three alternative solutions were considered and captured by EA Anamnesis: AL03 'file transfer' where the information from the institutions would be provided in the form of standardized data files, AL04 'shared database' where the application systems of the various institutions should be modified in order to have an common interface towards a centralized database, and AL05 'remote procedure invocation' where the application systems should be modified to exchange information directly without the intervention of a database. From a technical perspective two important qualities were captured: NFR04 'automated data integration' and NFR05 'interoperability'. However, the satisfaction of these non-functional requirements required a significant amount of development time, since the various application systems of social security institutions were developed with different technologies. Due to the fact that stakeholders had to deliver the solution in a short period of time they considered the non-functional requirement 'implementation time' (NFR01) as the most important for this decision making process. They used a lexicographic strategy to capture the prioritization of requirements. Figure 7.8 zooms further in the decision making process and presents the decision making strategies that was used.

### D04: develop upload/download application

Table 7.3 summarizes and Figure 7.9 visualizes the decision making process of D04. This decision concerns the selection of the appropriate means for the data file transfer between the social security institutions and the e-gov center. This is captured through FR04 'find platform for file transfer'. Two alternatives were considered : AL06 'file upload/download application', which means that the e-gov center should develop a web application that would facilitate users to download and upload the data files and AL07 'ftp server' where users would upload/download files through an FTP service. The architecture team elicited

Table 7.3: EA design decision 04 details

| | |
|---|---|
| **Decision:** | D04: develop upload/download application |
| **Functional requirement:** | FR04: find platform for file transfer |
| **EA perspective:** | Application layer |
| **Alternatives:** | AL06: develop upload/download application<br>AL07: ftp server |
| **Non-functional requirements:** | NFR06: users administration capability<br>NFR07: user auditing<br>NFR08: file size capability |

three different non-functional requirements. These requirements were potentially addressing technical interests of the IT stakeholders. As such, there were not motivated by any strategic goal or principle.

These non-functional requirements were of different importance. In collaboration with the

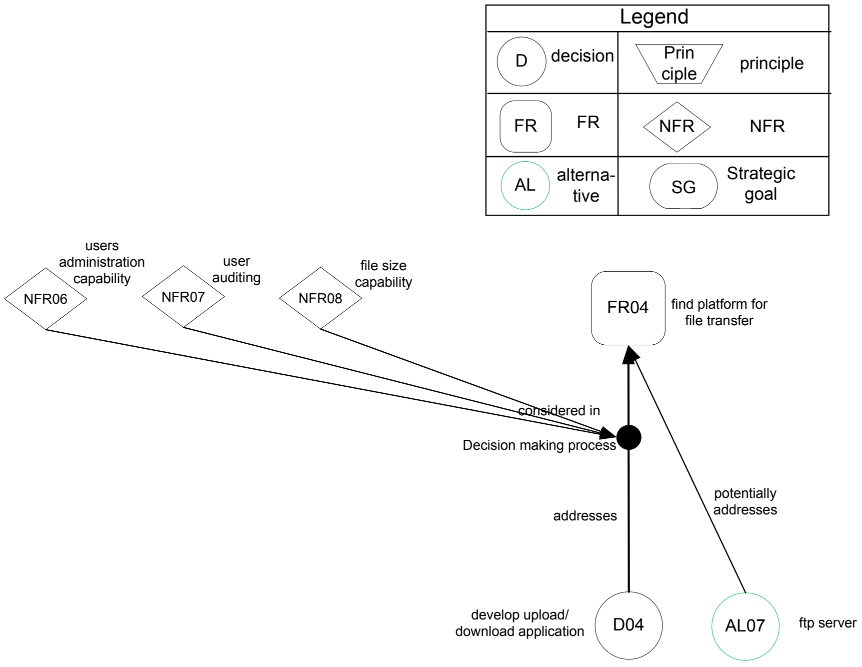


Figure 7.9: Decision making process of D04

stakeholders we captured this variation of importance with the use of a MAUT decision making strategy. We used a 0 to 10 scale to capture the importance and the value of the non-functional requirements.

NFR06 ‘users administration capability’ determines how capable each alternative solution is to manage the various user accounts of the different institutions and their access rights. In terms of its importance, stakeholders assigned 7/10. The value for NFR06 of the alternative ‘develop upload/download application’ was 8/10 whereas of ‘ftp server’ was 5/10. This is because the ‘ftp server’ had an autonomous account management system and it was not possible for users to manage their accounts.

NFR07 ‘user auditing’ indicates the capability of a solution to keep track of the activity of users. For example, an overview of what was uploaded, when and from which user. Due to the criticality of the information this non-functional requirement was consider of higher importance. Stakeholders assigned 9/10. The value of NFR07 for the alternative ‘develop upload/download application’ was 9/10 whereas for ‘ftp server’ was 5/10. The alternative ‘ftp server’ had a lack of such a mechanism’.

Finally, NFR08 ‘file size capability’ had to do with the capability of each solution to handle large file sizes. In terms of importance stakeholders assigned 6 out of 10. The value of NFR08 for the alternative ‘develop upload/download application’ was 5/10 whereas for ‘ftp server’ was 10/10. Due to some restrictions in the development environment, it was not possible for the application to support uploads/downloads of large file sizes.

Based on Equation 3.2 we calculated the score of alternatives. The alternative ‘develop upload/download application’ had the higher score and it was selected by the IT stakeholders. Table 7.4 summarizes the importance and values of non-functional requirements and scores of the considered alternatives.

Figure 7.10 zooms further in the decision making process and presents the decision making strategies that were used.

Table 7.4: D04 compensatory MAUT strategy

| **Alternatives** | **NFR06** | **NFR07** | **NFR08** | **score** |
|---|---|---|---|---|
| develop upload/download application | 7x8 | 9x9 | 6x5 | 167 |
| ftp server | 7x5 | 9x5 | 6x10 | 140 |

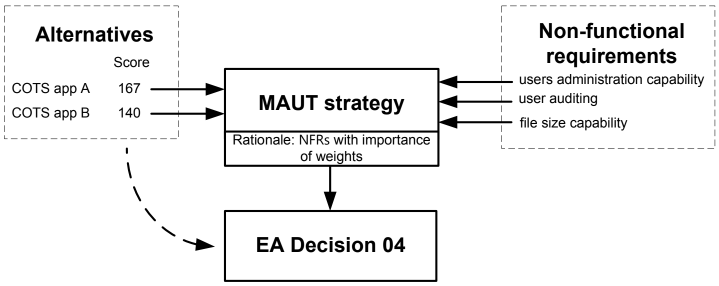


Figure 7.10: Decision making strategy of D04

## 7.5 Evaluation

Similarly with Section 6.5, this Section presents our reflections based on the the key quality characteristics of Section 1.4.3. Moreover, we present some additional lesson learned from the application of EA Anamnesis to the e-gov organization.

We enlist our findings with regard to the quality characteristics:

- *Domain appropriateness:*

  This quality characteristic deals with the ability of EA Anamnesis to express design rationale in an EA context. During the modeling exercise, practitioners were able to express the various design rationale, namely the design problem that initiated the design process, the justification by means of decision making strategies for specific design decisions and the cross-domain implications of their decision making. Moreover, practitioners understood the decision making of their colleagues from other domains of the organization. Moreover, they realized the relationship of their decision making with other decisions and unanticipated observed impacts in the organization.

- *Participant language knowledge appropriateness:*

  This quality characteristic deals with practitioner's level of understanding of EA Anamnesis terminology. Before the initiation of the design rationale modeling exercise, we demonstrated and explained to practitioners the concepts of EA Anamnesis. As such, we ensured that practitioners had a proper understanding of them. Thereafter, they confirmed that they were able to understand and use the concepts for expressing the design rationale of the case study. Finally, we exposed the captured design rationale information of one domain to stakeholders of other domains and confirmed that the captured information was understood by other stakeholders.

- *Knowledge externalizability appropriateness:*

  This quality characteristic deals with the level that the knowledge of practitioners was reflected by EA Anamnesis. Practitioners were using implicitly different ways of decision making. EA Anamnesis helped them to externalize the various evaluation criteria of their decision making and the various decision making strategies that they used. Furthermore, as we will see below in Section 7.6, EA Anamnesis made explicit and raised awareness regarding some problematic situations in the enterprise.

## 7.6 Lessons learned

- *Lesson 1: The capturing effort of EA Anamnesis can be reduced by selectively capturing design decisions based on their impact*

  An important critique against rationale management systems, like EA Anamnesis, is that they require an effort for the capturing and maintenance of the design rationale information (Lee 1997). During our case study, we observed that some design decisions had a high impact on the EA, in terms of changes or observed unanticipated consequences from the architectural design. We came up with design decisions which did not play a significant role in our analysis.

  Based on this observation, we argue that the capturing effort of EA Anamnesis can be significantly reduced by doing a selective capture of the most critical design decisions. However, the definition of criticality is subjective and it depends on the viewpoint of stakeholders of the EA. For EA Anamnesis, a criticality parameter can be the number of functional requirements and observed impacts that originate from a design decision. Of course, the ideal situation would be to capture every design decision and its relationships. At the same time, we believe that capturing high impact decisions would potentially be useful in an organization for the adoption of such an approach.

- *Lesson 2: EA Anamnesis raises awareness of problematic situations in the enterprise*

  The national pension salary payment and report system is considered as a quite successful project especially if we take into account how quickly it was implemented. It provides the government with the requested results and the operation of the business collaboration between the e-gov center and the social security institutions has been normalized. However, our analysis showed that there are many of malfunctions in the EA of this project. During our study we observed that some malfunctions that actually increased the operation costs of the business collaboration were not considered as open issues for further improvement. Most of the problems were disregarded, since the project was providing the requested results and the key stakeholders were preoccupied with the operational support in the current architecture context. In other words, there was no time to think for an improvement in the EA. EA Anamnesis helped stakeholders to realize and rethink about these problematic situations.

## 7.7 Conclusion

This chapter presented the application of EA Anamnesis to an EA transformation in an e-government organization in Greece. Through this case study we confirmed that EA Anamnesis can indeed capture and express design rationale in an EA context and that practitioners can understand and express their decision making based on the EA Anamnesis concepts. Finally, we learned that EA Anamnesis raises awareness regarding problematic situations in the enterprise and that there is the possibility to reduce the capturing effort by selectively capturing design rationale.

# Part IV

# Closing

# CHAPTER 8

# Conclusions and Further Research

In this chapter we present the summary and the general conclusions that arise from our research. We first start by providing an overview of EA Anamnesis where we describe the main contributions and their implications for research and practice. Thereafter, we will revisit the research questions and we will explain how they were addressed. Finally, we provide some closing remarks.

## 8.1 Recapitulation of EA Anamnesis

In this thesis we presented EA Anamnesis, a conceptual framework for EA design rationalization. While design rationalization was already investigated in other domains, the domain of EA was still unexplored. To the best of our knowledge EA Anamnesis is the first design rationale conceptual framework for the domain of EA.

EA Anamnesis complements EA modeling languages by capturing the underlying rationalization information of EA designs. For the development of EA Anamnesis, we followed a research design which comprises three methodologies: the design science research paradigm by Peffers et al. (2007), which indicated the necessary steps for the identification of the objectives, motivation, development and evaluation of EA Anamnesis, the work by Gregor and Jones (2007), which provides guidelines regarding the structural characteristics of EA Anamnesis and the work by Krogstie (2002) which provided quality characteristics that guided us during the identification of objectives and evaluation of EA Anamnesis.

As such, our design artifact was developed by taking into account the specificities of EA. We used the notion of EA perspectives in order to categorize design rationales and reveal their cross-domain relationships. Moreover, we borrowed concepts such as goal, principle and requirement from goal modeling techniques in order to capture the design problem formulation. Furthermore, based on multicriteria decision analysis theories, we captured how the given requirements were balanced during the decision making process for the evaluation of alternatives.

Our survey study (Section 2.3) showed that the concepts of EA Anamnesis are considered as useful by practitioners for the maintenance and justification of EAs. Moreover, through our case studies (Chapters 6 and 7), we evaluated EA Anamnesis based on the key quality characteristics and we showed that EA Anamnesis helps practitioners on the analysis and understanding of EA designs. Throughout the evaluations, EA Anamnesis was able to capture and represent adequately the decision making of practitioners. Despite the fact that practitioners did not have any previous experience in the use of decision making techniques, after a certain level of familiarization with our conceptual framework, they were able to recognize which decision making strategy they used for their evaluations.

Moreover, we gained insights regarding the current status of architecture decision making. In many cases, practitioners did not actually evaluate alternatives, but rather decided based on previous experiences from similar cases. Another interesting finding was that the capturing of design rationale through EA Anamnesis raised the awareness of stakeholders with regard to problematic situations in the enterprise.

## 8.2 Answering the individual research questions

- *RQ1: Which design rationale concepts can be used for the rationalization of EA designs?*

  As we stated in Section 1.2, this question can be broken down into the following two subquestions:

  - *RQ2: How to make explicit the underlying reasoning behind design decisions?*

    To answer this question we used concepts from multi-criteria decision making literature, such as 'decision making strategy' in order to capture in a structured way the reasoning behind the selection of specific design decisions. By doing so, we were able to capture which requirements were considered during the evaluation process and how the decision maker balanced amongst them. Our evaluation case studies showed that the decision making processes of practitioners were captured and reflected adequately through this conceptualization.

  - *RQ3: How to capture and represent the design problem and its role in the decision making process?*

    This research question was answered in two parts. The first part of the answer was provided in our generic metamodel. The generic metamodel is comprised by the problem and the solution space. In the problem space we used the notion of requirement as a key concept for the formulation of the design problem. Moreover, we used requirement as a bridging concept between the problem and solution space. By doing so, we were able to capture how the given requirements triggered the execution of new design decisions, how design decisions motivate new requirements and how requirements were evaluated during the decision making process. The second part of the answer is provided in the EA Anamnesis metamodel where we account concepts which are widely used in the domain of EA (Goal, principle) for the formulation of the design problem.

- *RQ4: How to make explicit the cross-domain relationships of design rationales?*

  In order to answer this question, the structured way that EA follows to define and view enterprises in EA perspectives was used. EA perspectives allowed us to categorize design rationales according to their architectural domain and to reveal their cross-domain interrelationships by connecting a rationale element in one domain to another rationale element in another domain. Practitioners, during case study evaluations, were able to identify how the design problem was formulated across the enterprise and cross-domain implications of their design decisions.

## 8.3 Implications for research and practice

Our research has implications both on research as well as practice. With reference to research we contribute to the domain of EA by introducing a conceptual framework for the rationalization of EA designs. EA Anamnesis, is the first design rationale approach developed for the domain of EA. By doing so, we also manifest the need for additional research work towards the direction of EA design rationalization. EA Anamnesis has already inspired EA research scholars to further work on the development of relevant approaches which are based on EA Anamnesis (Zimmermann et al. 2016, Martakis 2015).

Moreover, as stated in Section 1.5, parts of our design artifact contribute as well to the domain of design rationale in general. More specifically, EA Anamnesis makes use of multi-criteria decision analysis techniques for the formalization of decision making processes and subsequently for the provision of justification behind design decisions. Such a contribution can be useful for other design rationale approaches, where the decision making process involves multiple evaluation criteria. Our formalization provides insight in how the decision maker balanced among given criteria and as such it provides a better insight in the evaluation process.

Another important implication for research is the bridging between the domains of goal modeling and design rationale. Despite the fact that the two domains have overlapping research areas, we sometimes had the feeling that both communities work independently without taking into account the findings of each other. For example, while there are approaches for the prioritization of requirements in the domain of goal modeling, we rarely see such techniques to be applied in the domain of design rationale. In EA Anamnesis, we tried to bring together the two domains by capturing how the requirements were balanced during the decision making process.

For practice, EA Anamnesis serves as a conceptual framework which complements existing EA modeling languages. Organizations can use EA Anamnesis for the creation of design rationale knowledge repositories where the justifications, motivations and possible problematic situations of decision making are stored. By doing so, EA practitioners, especially newcomers, can quickly catch up and understand the as-is EA design and they can better maintain and extend the EA by taking into account possible problematic situations and vulnerabilities of the architecture.

## 8.4 Proposal for future research

EA Anamnesis is a design artifact which ex-post captures EA design rationale. During the development of EA Anamnesis we identified some research topics that would potentially improve and extend our research work. Below we present the additional research directions that we consider important to explore further.

- *Concrete syntax of EA Anamnesis*

  As we have seen in Section 1.3, EA Anamnesis is a conceptual framework which comprises an abstract syntax that rationalizes EA designs. Furthermore, we provided an concrete syntax which was used for the demonstrator of EA Anamnesis. The abstract syntax was used during the demonstration of EA Anamnesis with the software tool and during the evaluation of EA Anamnesis in our case studies. This concrete syntax is considered a 'byproduct' of our research.

  However, towards the development of a full-fledged modeling language we should focus further on the definition of a concrete syntax and its corresponding visualization which should take into account important issues, such as the understandability of notations to end users, incorporating as much as possible design elements proven to be effective already etcetera (Moody 2009, van der Linden et al. 2016a;b).

- *A-priori decision making support*

  The main goal of EA Anamnesis is to *ex-post* (after the design decisions have been made) rationalize EA designs. In that way it plays the role of a *descriptive* rationale management system (RMS) (Burge and Brown 1998). Descriptive approaches are designed to capture the thinking process of designers without intervening in this process. Their main focus is on organizing the design rationale after the design decisions have been made. They are mostly used for design teaching and maintenance activities.

  During the evaluation of our design artifact, we observed that practitioners became aware and got familiarized with various decision making strategies for the capturing of the reasoning of their decisions. We argue that our conceptualization of decision making processes can act as a basis for the *a-priori* (before the execution of the decision making) provision of procedural decisional guidance. By doing so, EA Anamnesis, in addition to its *descriptive* role, can play the *prescriptive* role of an RMS (Burge and Brown 1998). Prescriptive approaches focus on intervening in the activities of designers. Though this intervention, they aim to improve the decision making process and reasoning of designers and in turn make the design more concrete and persistent.

- *Collaborative decision making support*

  During the EA design process, various stakeholders with different individual stakes, from business as well as IT, have to collaborate to come to the final EA design. As mentioned above, EA Anamnesis captures ex-post the outcomes of this collaborative decision making process. EA Anamnesis could be extended in order to support collaborative decision making processes by taking into account the individual concerns of stakeholders. A first step towards this direction has been already done in the work of (Jugel et al. 2015). The authors used EA Anamnesis metamodel as a basis and extended it in order to support multi-perspective and collaborative decision-making processes.

- *Exploration of different decision models*

  As we mentioned in Section 3.1, during the exploration of decision analysis literature, we identified relevant models such as AHP and OWA that could potentially provide a better insight on the reasoning of decision makers. However, we decided to use relative simple decision schemers which would facilitate the uptake of EA Anamnesis in practice. We believe, that it would be beneficial as a research direction to explore different decision models and investigate through case study research if these models can better reflect the complexity of decision making in EA.

- *Investigate the semantic closeness between the concepts of goal, principle, requirement*

  The development of our problem space viewpoint was done by taking into account the EA practice and by also considering the existing state of the art for the domain of EA. In the case of problem space analysis we took into account the motivational extension of ArchiMate and more specifically the ARMOR language (Quartel et al. 2009). ARMOR comprises a quite significant number of concepts.

  However, according to (Engelsman and Wieringa 2014; 2012) the large amount of concepts combined with their sometimes ambiguous definition, introduces difficulties in the usability of the motivational extension by EA practitioners. The authors identified that three concepts were well understood: ‘stakeholder’, ‘requirement’ and ‘goal’. Interestingly, the large amount of ambiguously defined concepts also hints that the motivational extension is at odds with one of the key design principles behind the ArchiMate language: ‘the language should be as compact is possible’ (Lankhorst et al. 2010).

  Another interesting finding was that practitioners had difficulties to distinguish among the concepts of requirement and goal due to their semantic closeness (Engelsman and Wieringa 2014). We argue that the same ambiguity may exist as well for the distinction between the concepts ‘requirement’ and ‘principle’. Unfortunately the concept ‘principle’ was not included in the aforementioned study, but we suspect that the semantic closeness between principle and requirement, which are defined as high level normative properties, leads as well to ambiguities. As an example of such an ambiguity the readers can have a look at the principle 'interoperability', found in an EA specification document (Deighton 2014). Interoperability can be also considered as a non-functional requirement for the selection of an IT system.

We believe that these findings should be taken into account in the next iterations of development of EA Anamnesis. An idea for the manifestation of this semantic closeness is to use for the concepts goal, principle, requirements of our metamodel UML *stereotyping* techniques. Stereotyping is used to provide a lightweight distinction amongst concepts (Rumbaugh et al. 2004). For example, we can use 'requirement' as a unique concept and then stereotype it into the concepts 'principle' and 'goal'. To understand the different semantic interpretations of these concepts we can use existing data showing category structures (van der Linden and Proper 2014), and semantic features (van der Linden and van Zee 2014). Furthermore, we can elicit the conceptual understanding they have of these concepts, (van der Linden et al. 2012) to show which people think alike (van der Linden and Hoppenbrouwers 2012).

- *Return of capturing effort*

  Another important challenge is to investigate the return of capturing effort of EA Anamnesis. EA Anamnesis assists architects to better understand existing EA designs, but the effort of capturing this information might be a discouraging factor. The return of capturing effort should be more than satisfactory as a prerequisite for using EA Anamnesis. To do this, effective ways of capturing design decisions during the design process should be investigated and integrated into EA Anamnesis. Such a way was identified during our case study evaluations. The capturing effort could be significantly reduced by capturing design decisions in a selective manner based on the impact and unanticipated consequences of decisions in the EA.

## 8.5 Closing remarks

In this thesis we introduced EA Anamnesis, a conceptual framework for the rationalization of EA designs. EA Anamnesis is the first design rationale approach that was developed by taking into account key characteristics of EA. We believe that one of the key contributions of our work is that it manifests and justifies the need for additional research work in the area of EA design rationalization.

Moreover, some parts of EA Anamnesis contribute to the research domain of design rationale in general. Our decision making processes conceptualization (based on MCDA) can be generically used in environments where the decision making involves consideration and trade-off analysis among multiple evaluation criteria. Last but not least, in this work we tried to bring together the domains of goal modeling and design rationale by considering requirements as entities that can be used for the design problem formulation and as evaluation criteria during the decision making process.

EA practitioners can benefit from EA Anamnesis, since the provision of design rationalization information can help them to realize how the design problem was formulated across the enterprise, how the design problem was addressed by means of design decisions and their rationalization, and what were the unanticipated consequences of these decisions across the architecture.

# Bibliography

P. Avgeriou, J. Grundy, J. G. Hall, P. Lago, and I. Mistrík. *Relating software requirements and architectures*. Springer Science & Business Media, 2011.

W. Banks. Linguistic variables: Clear thinking with fuzzy logic. *Mode of access http://www. phaedsys. com/principals/bytecraft/bytecraftdata/Linguistic Variables. pdf*, 2008.

J. Baron and J. C. Hershey. Outcome bias in decision evaluation. *Journal of personality and social psychology*, 54(4):569, 1988. doi: dx.doi.org/10.1037/0022-3514.54.4.569.

J. Burge and D. C. Brown. Design rationale types and tools. Technical report, Worcester Polytechnic University, 1998.

J. Cummins and N. Doherty. The economics of insurance intermediaries. *Journal of Risk and Insurance*, 73(3):359–396, 2006. doi: 10.1111/j.1539-6975.2006.00180.x.

R. C. de Boer, R. Farenhorst, P. Lago, H. van Vliet, V. Clerc, and A. Jansen. *Architectural Knowledge: Getting to the Core*, pages 197–214. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007. ISBN 978-3-540-77619-2. doi: 10.1007/978-3-540-77619-2_12.

D. Deighton. Enterprise architecture principles. Technical report, IT Services, University of Birmingham, 2014.

J. L. Dietz. *What is Enterprise Ontology?*, pages 7–13. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006. ISBN 978-3-540-33149-0. doi: 10.1007/3-540-33149-2_2.

A. H. Dutoit, R. McCall, I. Mistrík, and B. Paech. *Rationale Management in Software Engineering*, chapter Rationale Management in Software Engineering: Concepts and Techniques, pages 1–48. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006. ISBN 978-3-540-30998-7. doi: 10.1007/978-3-540-30998-7_1.

H. Einhorn. The use of nonlinear, noncompensatory models in decision making. *Psychological bulletin*, 73(3):221–230, 1970. doi: dx.doi.org/10.1037/h0028695.

T. Elrod, R. Johnson, and J. White. A new integrated model of noncompensatory and compensatory decision strategies. *Organizational Behavior and Human Decision Processes*, 95(1):1–19, 2004. doi: dx.doi.org/10.1016/j.obhdp.2004.06.002.

W. Engelsman and R. Wieringa. Goal-oriented requirements engineering and enterprise architecture: Two case studies and some lessons learned. In *Requirements Engineering: Foundation for Software Quality*, pages 306–320. Springer, 2012.

W. Engelsman and R. Wieringa. *Understandability of Goal-Oriented Requirements Engineering Concepts for Enterprise Architects*, pages 105–119. Springer International Publishing, Cham, 2014. ISBN 978-3-319-07881-6. doi: 10.1007/978-3-319-07881-6_8.

H. Faller. *Organizational Subcultures and Enterprise Architecture Effectiveness: an Explanatory Theory.* PhD thesis, EE Series, Radboud Universiteit Nijmegen, 2016.

J. Figueira, S. Greco, and M. Ehrgott. *Multiple criteria decision analysis: state of the art surveys*, volume 78. Springer Science & Business Media, 2005.

C. Fischer, R. Winter, and F. Wortmann. Design theory. *Business & Information Systems Engineering*, 2(6):387–390, 2010. doi: 10.1007/s12599-010-0128-2.

N. E. Fuchs, U. Schwertel, and R. Schwitter. Attempto Controlled English – Not Just Another Logic Specification Language. *Lecture Notes in Computer Science*, 1559:1–20, 1999. doi: 10.1007/3-540-48958-4_1.

S. Ghanavati, D. Amyot, and L. Peyton. Compliance analysis based on a goal-oriented requirement language evaluation methodology. In *17th IEEE International Requirements Engineering Conference*, pages 133–142, 2009. doi: 10.1109/RE.2009.42.

J. Gordijn and H. Akkermans. Value based requirements engineering: Exploring innovative e-commerce ideas. *Requirements Engineering Journal*, 8(2):114–134, 2003. doi: 10.1007/s00766-003-0169-x.

D. Greefhorst and H. A. Proper. *Architecture Principles: The Cornerstones of Enterprise Architecture.* Springer Publishing Company, Incorporated, 1st edition, 2011. ISBN 9783642202780.

D. Greefhorst, H. A. Proper, and G. Plataniotis. The dutch state of the practice of architecture principles. *Journal of Enterprise Architecture*, 4:20–25, 2013.

S. Gregor and D. Jones. The anatomy of a design theory. *Journal of the Association for Information Systems*, 8(5):312–335, 2007.

S. O. Hansson. Decision making under great uncertainty. *Philosophy of the social sciences*, 26(3):369–386, 1996. doi: 10.1177/004839319602600304.

J. C. Henderson and N. Venkatraman. Strategic alignment: Leveraging information technology for transforming organizations. *IBM systems journal*, 32(1):4–16, 1993. doi: 10.1147/sj.382.0472.

A. R. Hevner, S. T. March, J. Park, and S. Ram. Design science in information systems research. *MIS Q.*, 28(1):75–105, Mar. 2004. ISSN 0276-7783.

F. S. Hillier. *Introduction to operations research.* Tata McGraw-Hill Education, 1995.

J. Hoogervorst. Enterprise architecture: Enabling integration, agility and change. *International Journal of Cooperative Information Systems*, 13(03):213–233, 2004. doi: dx.doi.org/10.1142/S021884300400095X.

C.-L. Hwang and K. Yoon. *Multiple attribute decision making: methods and applications a state-of-the-art survey*, volume 186. Springer Science & Business Media, 2012.

IEEE. Systems and software engineering – architecture description. *ISO/IEC/IEEE 42010:2011(E) (Revision of ISO/IEC 42010:2007 and IEEE Std 1471-2000)*, pages 1 –46, 1 2011. doi: 10.1109/IEEESTD.2011.6129467.

K. Isabels and D. G. Uthra. An application of linguistic variables in assignment problem with fuzzy costs. *International Journal of Computational Engineering Research*, pages 1065–1069, 2012.

A. Jadhav and R. Sonar. Evaluating and selecting software packages: A review. *Information and software technology*, 51(3):555–563, 2009. doi: 10.1016/j.infsof.2008.09.003.

A. Jansen and J. Bosch. Software architecture as a set of architectural design decisions. In *Software Architecture, 2005. WICSA 2005. 5th Working IEEE/IFIP Conference on*, pages 109–120. IEEE, 2005.

A. P. Jarczyk, P. Löffler, and F. M. Shipman III. Design rationale for software engineering: a survey. In *Proceedings of the Twenty-Fifth Hawaii International Conference on System Sciences*, volume 2, pages 577–586. IEEE, 1992.

I. Jeffreys. The use of compensatory and non-compensatory multi-criteria analysis for small-scale forestry. *Small-scale Forestry*, 3(1):99–117, 2004. doi: 10.1007/s11842-004-0007-0.

D. Jugel, C. M. Schweda, and A. Zimmermann. *Advanced Information Systems Engineering Workshops: CAiSE 2015 International Workshops, Stockholm, Sweden, June 8-9, 2015, Proceedings*, chapter Modeling Decisions for Collaborative Enterprise Architecture Engineering, pages 351–362. Springer International Publishing, Cham, 2015. ISBN 978-3-319-19243-7. doi: 10.1007/978-3-319-19243-7_33.

Y.-G. Kim and G. C. Everest. Building an IS architecture: Collective wisdom from the field. *Information & Management*, 26(1):1–11, 1994. doi: 10.1016/0378-7206(94)90002-7.

J. Kozielecki. *Psychological decision theory*, volume 24. Springer Science & Business Media, 1982.

J. Krogstie. A Semiotic Approach to Quality in Requirements Specifications. In L. Kecheng, R. J. Clarke, P. B. Andersen, R. K. Stamper, and E.-S. Abou-Zeid, editors, *Proceedings of the IFIP TC8 / WG8.1 Working Conference on Organizational Semiotics: Evolving a Science of Information Systems*, pages 231–250, Deventer, The Netherlands, 2002. Kluwer. ISBN 1-402-07189-2.

J. Krogstie, O. I. Lindland, and G. Sindre. Defining quality aspects for conceptual models. In E. D. Falkenberg, W. Hesse, and A. Olivé, editors, *Information System Concepts: Towards a consolidation of views – Proceedings of the third IFIP WG8.1 conference (ISCO–3)*, pages 216–231, Marburg, Germany, March 1995. Chapman & Hall/IFIP WG8.1, London, United Kingdom.

P. Kruchten. An ontology of architectural design decisions in software intensive systems. In *2nd Groningen Workshop on Software Variability*, pages 54–61, 2004.

W. Kunz and H. W. Rittel. *Issues as elements of information systems*, volume 131. Institute of Urban and Regional Development, University of California, Berkeley, California, 1970.

M. Lankhorst. *Enterprise Architecture at Work: Modelling, Communication and Analysis.* Springer, 3rd edition, 2013. ISBN 9783642296505.

M. M. Lankhorst, H. A. Proper, and H. Jonkers. The anatomy of the ArchiMate language. *International Journal of Information System Modeling and Design (IJISMD)*, 1(1):1–32, 2010. doi: 10.4018/jismd.2010092301.

A. Lapouchnian. Goal-oriented requirements engineering: An overview of the current research. Technical report, University of Toronto, Canada, 2005.

J. Lee. Extending the potts and bruns model for recording design rationale. In *Software Engineering, 1991. Proceedings., 13th International Conference on*, pages 114–125. IEEE, 1991.

J. Lee. Design rationale systems: understanding the issues. *IEEE intelligent systems*, (3):78–85, 1997. doi: 10.1109/64.592267.

J. Lee and K.-Y. Lai. What's in design rationale? *Human–Computer Interaction*, 6(3-4): 251–280, 1991. doi: 0.1207/s15327051hci0603\&4_3.

P. Loucopoulos and V. Karakostas. *System Requirements Engineering.* McGraw-Hill, Inc., New York, USA, 1995. ISBN 0077078438.

J. Luftman. Assessing Business-IT alignment maturity. *Strategies for information technology governance*, 4:99, 2004. doi: 10.4018/978-1-59140-140-7.ch004.

I. Malavolta, P. Lago, H. Muccini, P. Pelliccione, and A. Tang. What industry needs from architectural languages: A survey. *Software Engineering, IEEE Transactions on*, 39(6): 869–891, 2013. ISSN 0098-5589. doi: 10.1109/TSE.2012.74.

A. Martakis. Framework for enterprise uncertainty-driven decision-making: FEUD. Master's thesis, University of Twente, 2015.

D. L. Moody. The "Physics" of Notations: Toward a Scientific Basis for Constructing Visual Notations in Software Engineering. *IEEE Transactions on Software Engineering Software Engineering*, 35(6):756–779, 2009. doi: 10.1109/TSE.2009.67.

D. Nadler, M. S. Gerstein, and R. B. Shaw. *Organizational architecture: Designs for changing organizations.* 192. Jossey-Bass Inc Pub, 1992.

B. Nuseibeh. Weaving together requirements and architectures. *Computer*, 34(3):115–119, 2001. doi: 10.1109/2.910904.

OECD. The greek social security system. Technical report, Ministry of Labour and Social Security, General Secretariat for Social Security, 2002.

OMG. OMG Unified Modeling Language (OMG UML), Infrastructure, V2.1.2. Technical report, The Object Management Group, November 2007. URL `http://www.omg.org/spec/UML/2.1.2/Infrastructure/PDF`.

M. Op 't Land, H. Proper, M. Waage, J. Cloo, and C. Steghuis. *Enterprise architecture: creating value by informed governance*. Springer, 2008.

J. H. Panchal, M. G. Fernández, C. J. Paredis, J. K. Allen, and F. Mistree. A modular decision-centric approach for reusable design processes. *Concurrent Engineering*, 17(1): 5–19, 2009. doi: 10.1177/1063293X09102251.

M. M. Parker and R. J. Benson. Enterprisewide information management: state-of-the-art strategic planning. *Information System Management*, 6(3):14–23, 1989.

J. Payne. Task complexity and contingent processing in decision making: An information search and protocol analysis. *Organizational behavior and human performance*, 16(2): 366–387, 1976. doi: 10.1016/0030-5073(76)90022-2.

J. Payne, J. Bettman, and E. Johnson. *The adaptive decision maker*. Cambridge University Press, 1993.

K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee. A design science research methodology for information systems research. *Journal of management information systems*, 24(3):45–77, 2007. doi: 10.2753/MIS0742-1222240302.

K. Pohl. *Requirements Engineering: Fundamentals, Principles, and Techniques*. Springer Publishing Company, Incorporated, 1st edition, 2010. ISBN 3642125778, 9783642125775.

H. A. Proper and M. Op 't Land. Lines in the water. In F. Harmsen, H. A. Proper, F. Schalkwijk, J. Barjis, and S. Overbeek, editors, *Practice-Driven Research on Enterprise Transformation*, volume 69 of *Lecture Notes in Business Information Processing*, pages 193–216. Springer Berlin Heidelberg, 2010. ISBN 978-3-642-16769-0. doi: 10.1007/978-3-642-16770-6_9.

D. Quartel, W. Engelsman, H. Jonkers, and M. Van Sinderen. A goal-oriented requirements modelling language for enterprise architecture. In *Enterprise Distributed Object Computing Conference, 2009. EDOC'09. IEEE International*, pages 3–13. IEEE, 2009.

J. W. Ross, P. Weill, and D. Robertson. *Enterprise architecture as strategy: Creating a foundation for business execution*. Harvard Business Press, 2006. ISBN 978-1-4221-4817-4.

L. Rothrock and J. Yin. Integrating compensatory and noncompensatory decision-making strategies in dynamic task environments. *Decision Modeling and Behavior in Complex and Uncertain Environments*, pages 125–141, 2008. doi: 10.1007/978-0-387-77131-1_6.

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

P. Runeson and M. Host. Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering*, 14(2):131–164, 2009. ISSN 1382-3256. doi: 10.1007/s10664-008-9102-8.

T. L. Saaty. What is the analytic hierarchy process? In *Mathematical models for decision support*, pages 109–121. Springer, 1988.

J. Savolainen. Tools for design rationale documentation in the development of a product family. In *Proceedings of the 1st Working IFIP Conference on Software Architecture, San Antonio, Texas*, 1999.

R. Schwitter. *Controlled Natural Languages.* Centre for Language Technology, Macquary University, Sydney, New South Wales, Australia, 2004.

P. S. Seligmann, G. M. Wijers, and H. G. Sol. Analyzing the Structure of I. S. Methodologies, an alternative approach. In R. Maes, editor, *Proceedings of the First Dutch Conference on Information Systems (1989)*, 1989.

F. M. Shipman and R. J. McCall. Integrating different perspectives on design rationale: Supporting the emergence of design rationale from design communication. *Artificial Intelligence for Engineering, Design, Analysis and Manufacturing*, 11(02):141–154, 1997. doi: dx.doi.org/10.1017/S089006040000192X.

O. Svenson. Process descriptions of decision making. *Organizational behavior and human performance*, 23(1):86–112, 1979. doi: 10.1016/0030-5073(79)90048-5.

A. Tang, M. A. Babar, I. Gorton, and J. Han. A survey of architecture design rationale. *Journal of Systems and Software*, 79(12):1792 – 1804, 2006. ISSN 0164-1212. doi: 10.1016/j.jss.2006.04.029.

A. Tang, Y. Jin, and J. Han. A rationale-based architecture model for design traceability and reasoning. *Journal of Systems and Software*, 80(6):918 – 934, 2007. ISSN 0164-1212. doi: 10.1016/j.jss.2006.08.040.

A. Tang, P. Liang, V. Clerc, and H. van Vliet. Supporting co-evolving architectural requirements and design through traceability and reasoning. Technical report, 2011.

A. H. M. ter Hofstede and H. A. Proper. How to Formalize It? Formalization Principles for Information Systems Development Methods. *Information and Software Technology*, 40(10):519–540, October 1998. doi: dx.doi.org/10.1016/S0950-5849(98)00078-0.

The Open Group. *TOGAF Version 9.1.* Van Haren Publishing, 2011.

The Open Group. *ArchiMate 2.0 Specification.* Van Haren Publishing, 2012.

S. E. Toulmin. *The uses of argument.* Cambridge University Press, 2003.

J. Tyree and A. Akerman. Architecture decisions: Demystifying architecture. *Software, IEEE*, 22(2):19–27, 2005. doi: doi.ieeecomputersociety.org/10.1109/MS.2005.27.

D. van der Linden and S. Hoppenbrouwers. Challenges of identifying communities with shared semantics in enterprise modeling. In *IFIP Working Conference on The Practice of Enterprise Modeling*, pages 160–171. Springer, 2012.

D. van der Linden and H. A. Proper. Category structure of language types common to conceptual modeling languages. In *Enterprise, Business-Process and Information Systems Modeling*, pages 317–331. Springer, 2014.

D. van der Linden and M. van Zee. On the semantic feature structure of modeling concepts: an empirical study. In *Business Informatics (CBI), 2014 IEEE 16th Conference on*, volume 2, pages 158–165. IEEE, 2014.

D. van der Linden, S. Hoppenbrouwers, A. Lartseva, and W. Molnar. Beyond terminologies: Using psychometrics to validate shared ontologies. *Applied Ontology*, 7(4): 471–487, 2012.

D. van der Linden, I. Hadar, and A. Zamansky. Towards a marketplace of visual elements for notation design. In *Requirements Engineering Conference (RE), 2016 IEEE 24th International*, pages 353–358. IEEE, 2016a.

D. van der Linden, A. Zamansky, and I. Hadar. *How Cognitively Effective is a Visual Notation? On the Inherent Difficulty of Operationalizing the Physics of Notations*, pages 448–462. Springer International Publishing, Cham, 2016b. ISBN 978-3-319-39429-9. doi: 10.1007/978-3-319-39429-9_28.

A. van Lamsweerde. Goal-oriented requirements engineering: A guided tour. In *Requirements Engineering, 2001. Proceedings. Fifth IEEE International Symposium on*, pages 249–262. IEEE, 2001.

W. F. Van Raaij, G. M. van Veldhoven, and K.-E. Wärneryd. *Handbook of economic psychology.* Springer Science & Business Media, 2013.

M. van Steenbergen and S. Brinkkemper. Modeling the contribution of enterprise architecture practice to the achievement of business goals. In *Information Systems Development*, pages 609–618. Springer, 2009.

J. van't Wout, M. Waage, H. Hartman, M. Stahlecker, and A. Hofman. *The integrated architecture framework explained: why, what, how.* Springer Science & Business Media, 2010.

M. Whelton, G. Ballard, and I. Tommelein. Application of design rationale systems to project definition–establishing a research project. In *9th Conference of the International Group for Lean Construction (IGLC), Singapore, August*, volume 10, 2001.

W. L. Winston and J. B. Goldberg. *Operations research: applications and algorithms*, volume 3. Duxbury press Boston, 2004.

R. Winter. Design science research in Europe. *European Journal of Information Systems*, 17(5):470–475, 2008. doi: 10.1057/ejis.2008.44.

R. R. Yager. On ordered weighted averaging aggregation operators in multicriteria decisionmaking. *IEEE Transactions on systems, Man, and Cybernetics*, 18(1):183–190, 1988.

E. S. Yu. Towards modelling and reasoning support for early-phase requirements engineering. In *Requirements Engineering, 1997., Proceedings of the Third IEEE International Symposium on*, pages 226–235. IEEE, 1997.

J. A. Zachman. A framework for information systems architecture. *IBM systems journal*, 26(3):276–292, 1987. doi: 10.1147/sj.263.0276.

L. A. Zadeh. The concept of a linguistic variable and its application to approximate reasoning. *Information sciences*, 8(3):199–249, 1975.

A. Zimmermann, D. Jugel, K. Sandkuhl, R. Schmidt, C. Schweda, and M. Möhring. Architectural decision management for digital transformation of products and services. *Complex Systems Informatics and Modeling Quarterly*, (6):31–53, 2016. doi: 10.7250/csimq.2016-6.03.

# Summary

Enterprise transformations impose socio-technical changes to organizations. Enterprise architecture (EA) is acknowledged as a steering instrument that assists stakeholders in the process of an enterprise transformation. Amongst others, the practice of EA is supported by modeling languages which describe an enterprise holistically. By doing so, they show an enterprise's business products and services, and how these are realized by IT infrastructure and applications.

However, EA modeling languages lack the capability to capture the design rationale that led to specific architectural designs in the context of an enterprise transformation. This lack of transparency regarding design decisions can cause design integrity issues when architects have to maintain or change the current EA design. Due to this lack of insight into the rationale, new designs are constructed in an *ad-hoc* manner, without taking into account considerations and constraints implied by past design decisions.

This thesis addresses this lack of design rationale support for the domain of EA, by introducing EA Anamnesis, a conceptual framework for EA design rationalization. EA Anamnesis complements existing EA modeling languages with design rationalization information. This is realized by capturing rationale such as the formulation of the design problem across the EA, how the problem was addressed by specific design decisions, the reasoning behind the selection of those decisions and their possible unanticipated consequences, and by linking that rationale with elements of the EA design.

EA Anamnesis is developed in an iterative process, following the design science research paradigm. We first start with the problem identification and motivation. We explore the domains of EA and design rationale and then we identify their main characteristics. Based on that, we identify a set of key design rationale concepts which can be used as a basis

for the development of our conceptual framework. Through a survey study, we present this set of concepts to a group of EA practitioners. Practitioners consider the proposed concepts as useful and they confirm the need and usefulness of a design rationale approach for EA.

Thereafter, we proceed with our design artifact. EA Anamnesis provides analysis both on the problem and solution spaces of the EA design. For the problem space analysis, we use techniques from the domain of goal modeling in order to capture how the design problem is formulated with given goals, EA principles and requirements, terms which are widely used in the domain of EA. In the solution space, we use operation research theories, more specifically Multi-Criteria Decision Analysis (MCDA), in order to conceptualize the reasoning of the architects during the decision making process. MCDA allows us to approach the decision making process of architects as a decision making problem that involves multiple evaluation criteria amongst which decision makers have to balance. Furthermore, we use the structural approach of EA in order to categorize the various design rationale of the EA design and to make explicit their possible cross-domain relationships. For example, how a business decision triggers an IT requirement or vice versa.

The practical validity of EA Anamnesis is assessed through its application in two real-world case studies, one in a research and technology organization in Luxembourg and the other in a Greek e-government organization. The evaluation indicates that EA Anamnesis captures adequately the design rationale of the two cases and that practitioners are able to recognize how the design problem was formulated and solved by the appropriate design decisions. In addition to to the real world case study evaluation, we also illustrate EA Anamnesis through a fictitious case study from the insurance sector. This case study was used during the intermediary steps of our conceptual framework development for the illustration of the individual design artifact chunks. The feedback received through these illustrations enabled us to extend and improve EA Anamnesis. Last but not least, we use a prototype tool implementation to evaluate the implementability of our conceptual framework and for further extensions and improvements.

EA Anamnesis has implications both in practice and research. In practice, organizations can use EA Anamnesis for the creation of design rationale knowledge repositories where the justifications, motivational reasons and possible problematic situations of decision making can be stored. By doing so, EA practitioners, especially newcomers, can quickly catch up and understand the as-is EA design and can better maintain and extend the EA by taking into account possible problematic situations and vulnerabilities of the architecture.

Regarding the research implications, EA Anamnesis makes a contribution to the field of EA by manifesting the need for design rationale support in EA and by presenting an approach especially designed for this purpose. Moreover, we make contribution to the domain of design rationale by presenting a conceptualization of decision making processes based on MCDA for the provision of the reasoning behind design decisions. Last but not least, we make explicit the intertwining between problem and solution space by capturing how requirements trigger the execution of new design decisions, how the requirements are evaluated during the decision making process and how design decisions or their possible unanticipated observed impacts motivate the elicitation of new requirements.

Through our research we made a first attempt to support the domain of EA with design rationalization information. At the time of writing, EA Anamnesis has inspired other researchers to work towards this direction and has been used as basis for the development of relevant extensions. However, more work has to be done. Possible directions include extensions for a-priori and collaborative decision making support, reconsideration of the problem space part of the conceptual framework due to the semantic closeness of its concepts and the investigation of the return of capturing effort of EA Anamnesis.

# Samenvatting

Ondernemingstransformaties leggen socio-technologische veranderingen aan organisaties op. Ondernemingsarchitectuur, meer bekend als Enterprise Architecture (EA), wordt erkend als een sturingsinstrument dat belanghebbenden ondersteunt in het proces van een ondernemingstransformatie. De praktijk van EA wordt onder andere ondersteund door modelleertalen die een organisatie holistisch beschrijven. Dergelijke modellen maken de producten en diensten van een organisatie expliciet, en laten zien hoe deze gerealiseerd worden door IT infrastructuur en applicaties.

Modelleertalen voor EA hebben niet de capaciteit om ontwerpmotivaties die in de context van een ondernemingstransformatie tot een specifiek architectuurontwerp hebben geleid vast te leggen. Dit gebrek aan transparantie van de ontwerpbesluiten kan problemen met de integriteit van het ontwerp als gevolg hebben. Door dit gebrek aan inzicht in de motivaties worden nieuwe ontwerpen op een ad-hoc manier gebouwd, zonder overwegingen en beperkingen die impliciet in oudere ontwerpmotivaties liggen in overweging te nemen.

Deze dissertatie richt zich op dit gebrek aan steun voor ontwerpmotivering in het EA-domain door EA Anamnesis te introduceren. EA Anamnesis is een conceptueel raamwerk voor ontwerpmotivering van EA-besluiten. Het complementeert bestaande EA-modelleertalen met informatie over ontwerpmotivering. Dit wordt gedaan door motivaties zoals formulering van het ontwerpprobleem over het EA, hoe het probleem aangekaart is door specifieke ontwerpbesluiten, de motivatie achter de keuze voor die besluiten, en hun mogelijke onverwachte consequenties vast te leggen, en deze informatie te verbinden met elementen van het EA-ontwerp.

EA Anamnesis is ontwikkeld in een iteratief proces, volgens het "design science" paradigma. We beginnen met de identificatie van het probleem en onze motivering. We verkennen

de domeinen van EA en ontwerpmotivatie om hun voornaamste karakteristieken vast te identificeren. Op basis hiervan identificeren we een verzameling van belangrijkste ontwerpmotivatieconcepten die gebruikt kunnen worden als een basis voor de ontwikkeling van ons conceptueel raamwerk. Door middel van een questionnaire leggen we deze concepten voor aan een groep EA-beoefenaars. Beoefenaars vinden de voorgestelde concepten nuttig, en bevestigen de behoefte aan, en nut van een ontwerpmotivatiemethode voor EA.

Hierna gaan we verder met ons ontwerpartefact. EA Anamnesis levert analysemogelijkheden voor zowel de probleem- als oplossingskanten van het EA-ontwerp. Voor de analyse aan de probleemkant gebruiken we technieken uit het domein van doelmodellering om vast te leggen hoe het ontwerpprobleem geformuleerd is met doelen, EA-principes en benodigdheden, termen die een brede acceptatie binnen het domein van EA genieten. De analyse aan de oplossingskant wordt ondersteund door theorie uit "operations research," namelijk Multi-Criteria Decision Analysis (MCDA), om de redenering van de architecten tijdens het besluitvormingsproces vast te leggen. Met MCDA kunnen we het besluitvormingsproces van architecten als een besluitvormingprobleem behandelen waarin er meerdere evaluatiecriteria zijn waar besluitnemers een balans tussen moeten vinden. We gebruiken ook de structurele aanpak van EA om de diverse ontwerpmotivaties van een EA-ontwerp vast te leggen, en mogelijke verbindingen door meerdere domeinen heen vast te leggen. Bijvoorbeeld, hoe een besluit aan zakelijke kant een behoefte aan de informatie-technologiekant oplegt.

We stellen de praktische bruikbaarheid en validiteit van EA Anamnesis vast door het toe te passen in twee case studies uit de echte wereld: één in een organisatie voor onderzoek en technologie in Luxemburg, en één in een Griekse e-overheid organisatie. Uit deze evaluatie blijkt dat EA Anamnesis adequaat de ontwerpmotivaties van beide case studies vastlegt, en dat beoefenaars kunnen herkennen hoe het ontwerpprobleem was geformuleerd en opgelost door de relevante ontwerpbesluiten. We illustreren EA Anamnesis verder door een fictieve case study uit het verzekeringsdomein. Deze case study is ook gebruikt tijdens het ontwikkelingsproces van ons conceptueel raamwerk om individuele delen van ontwerpartefacten te illustreren. Met de FEEDBACK die we over deze illustraties ontvangen hebben is EA Anamnesis verder uitgebreid en verbeterd. Ten laatste, gebruiken de implementatie van een softwareprototype om de implementeerbaarheid van ons conceptueel raamwerk te evalueren, en als input te dienen voor verdere uitbreidingen en verbeteringen.

EA Anamnesis heeft zowel voor de praktijk als onderzoek nut. In de praktijk kunnen beoefenaars van EA Anamnesis gebruik maken om databases van ontwerpmotivatiekennis te maken. Hierin kunnen rechtvaardigingen, motivaties en mogelijk problematische situaties van besluitvorming vastgelegd worden. Hierdoor kunnen EA-beoefenaars, in het bijzonder nieuwkomers snel hun kennis van het EA-ontwerp bijhalen en makkelijker de

EA onderhouden en uitbreiden door mogelijke problematische situaties en zwakheden van de architectuur in acht te nemen.

Wat onderzoek betreft, levert EA Anamnesis een toevoeging aan het gebied van EA door de benodigdheid van ontwerpmotivatie in EA expliciet te maken, en hier een methode voor te geven. Bovendien leveren we een bijdrage aan het domein van ontwerpmotivatie door een conceptualisatie te leveren van besluitvormingsprocessen gebaseerd op MCDA voor het vastleggen van redenering achter ontwerpbesluiten. Ten laatste maken we duidelijk hoe probleem- en oplossingskanten verwoven zijn door vast te leggen hoe behoeftes de uitvoering van nieuwe ontwerpbesluiten tot gevolg hebben, hoe deze behoeftes worden geëvalueerd tijdens het besluitmakingsproces, en hoe ontwerpbesluiten of hun mogelijk onverwachte geobserveerde impact de ontlokking van nieuwe behoeftes motiveert.

Middels ons onderzoek hebben we een eerste stap gezet om het domein van EA te ondersteunen met informatie over ontwerpmotivatie. Op het moment van schrijven heeft EA Anamnesis al andere onderzoekers geïnspireerd om in deze richting te werken, en is ons werk als een basis voor de ontwikkeling van relevante uitbreidingen. Er blijft echter nog werk te doen. Mogelijke richtingen voor toekomstig onderzoek zijn uitbreidingen voor de ondersteuning van a-priori en collaboratieve besluitvorming, en een verdere analyse van de semantieke aspecten van de probleemkant van ons conceptueel raamwerk, en een onderzoek naar het rendement van de inspanning benodigd om EA Anamnesis toe te passen.

# Περίληψη

Οι επιχειρησιακοί μετασχηματισμοί επιφέρουν κοινωνικο-τεχνικές αλλαγές στους οργανισμούς. Η επιχειρησιακή αρχιτεκτονική (enterprise architecture, EA) αναγνωρίζεται ως το μέσο το οποίο βοηθά τα ενδιαφερόμενα μέρη κατά τη διάρκεια της διαδικασίας του επιχειρησιακού μετασχηματισμού. Μεταξύ άλλων, η πρακτική της επιχειρησιακής αρχιτεκτονικής υποστηρίζεται από γλώσσες μοντελοποίησης οι οποίες περιγράφουν τον οργανισμό με ολιστικό τρόπο. Με αυτόν τον τρόπο, περιγράφουν τα επιχειρησιακά προϊόντα και προσφερόμενες υπηρεσίες ενός οργανισμού, και το πώς αυτά πραγματοποιούνται μέσω της υποδομής πληροφοριακών συστημάτων και εφαρμογών.

Ωστόσο, οι γλώσσες μοντελοποίησης επιχειρησιακής αρχιτεκτονικής δεν έχουν τη δυνατότητα σύλληψης της αιτιολόγησης (design rationale) των αποφάσεων σχεδίασης που οδήγησαν σε συγκριμένες αρχιτεκτονικές σχεδιάσεις στα πλαίσια ενός επιχειρησιακού μετασχηματισμού. Η έλλειψη διαφάνειας στην αιτιολόγηση των αποφάσεων σχεδίασης μπορεί να προκαλέσει θέματα ακεραιότητας σχεδιασμού όταν οι σχεδιαστές αρχιτεκτονικής πρέπει να συντηρήσουν ή να μεταβάλουν την ισχύουσα σχεδίαση της υπηρεσιακής αρχιτεκτονικής. Εξαιτίας της έλλειψης επίγνωσης της αιτιολόγησης σχεδίασης, οι νέες σχεδιάσεις γίνονται με ad-hoc τρόπο, χωρίς να λαμβάνονται υπόψη οι παραδοχές και περιορισμοί παλαιότερων αποφάσεων σχεδίασης.

Αυτή η διατριβή αντιμετωπίζει την έλλειψη της αιτιολόγησης σχεδίασης σε περιβάλλοντα επιχειρησιακής αρχιτεκτονικής, παρουσιάζοντας το EA Anamnesis, ένα εννοιολογικό πλαίσιο (conceptual framework) για τον εξορθολογισμό σχεδίασης της υπηρεσιακής αρχιτεκτονικής. Το EA Anamnesis συμπληρώνει τις υπάρχουσες γλώσσες μοντελοποίησης υπηρεσιακής αρχιτεκτονικής με πληροφορία αιτιολόγησης της σχεδίασης. Αυτό πραγματοποιείται μέσω της σύλληψης αιτιολογήσεων σχεδιασμού, όπως η διατύπωση του σχεδιαστικού προβλήματος της

επιχειρησιακής αρχιτεκτονικής, πως το πρόβλημα επιλύθηκε μέσω συγκεκριμένων αποφάσεων σχεδίασης, την αιτιολόγηση πίσω από την επιλογή αυτών των αποφάσεων και τις πιθανές απροσδόκητες επιπτώσεις τους, και με τη σύνδεση των αιτιολογήσεων σχεδίασης με στοιχεία της σχεδίασης της υπηρεσιακής αρχιτεκτονικής.

Το EA Anamnesis, έχει αναπτυχθεί μέσω μια επαναληπτικής διαδικασίας, η οποία ακολουθεί το παράδειγμα της έρευνας της επιστήμης σχεδιασμού (design science research). Αρχικά ξεκινάμε με την αναγνώριση του ερευνητικού προβλήματος και με τα κίνητρα της ερευνάς μας. Διερευνούμε τους τομείς της επιχειρησιακής αρχιτεκτονικής και αιτιολόγησης σχεδιασμού και εντοπίζουμε τα κύρια χαρακτηριστικά τους. Βάσει αυτού, εντοπίζουμε μια ομάδα βασικών εννοιών (concepts) που μπορούν να χρησιμοποιηθούν σαν τη βάση για την ανάπτυξη του εννοιολογικού μας πλαισίου. Μέσω μιας ερευνητικής επισκόπησης (survey), παρουσιάζουμε αυτή την ομάδα βασικών εννοιών σε μια ομάδα επαγγελματιών επιχειρησιακής αρχιτεκτονικής. Οι επαγγελματίες θεωρούν ότι οι προτεινόμενες έννοιες είναι χρήσιμες και επιβεβαιώνουν τη αναγκαιότητα και τη χρησιμότητα μιας προσέγγισης αιτιολογήσης σχεδιασμού για επιχειρησιακές αρχιτεκτονικές.

Έπειτα, προχωρούμε με το σχεδιαστικό μας δημιούργημα (design artifact). Το EA Anamnesis είναι βασισμένο σε τεχνικές επιχειρησιακής έρευνας και πιο συγκεκριμένα σε πολυκριτηριακή ανάλυση αποφάσεων (MCDA) για την εννοιοποίηση (conceptualization) της αιτιολόγησης των σχεδιαστών αρχιτεκτονικής κατά τη διάρκεια της διαδικασίας λήψεων αποφάσεων. Η πολυκριτηριακή ανάλυση αποφάσεων μας επιτρέπει να προσεγγίσουμε τη διαδικασία λήψης απόφασης των σχεδιαστών αρχιτεκτονικής σαν ένα πρόβλημα λήψης απόφασης που περιλαμβάνει πολλαπλά κριτήρια αξιολόγησης μεταξύ των οποίων τα όργανα λήψης αποφάσεων θα πρέπει να βρουν τη κατάλληλη ισορροπία. Για την ανάλυση στο χώρο προβλήματος χρησιμοποιούμε τεχνικές από το πεδίο μοντελοποίησης στόχων (goal modeling) ώστε να συλλάβουμε τον τρόπο με τον οποίο το σχεδιαστικό πρόβλημα διατυπώνεται βάση συγκεκριμένων στόχων, αρχών σχεδίασης επιχειρησιακής αρχιτεκτονικής (EA principles) και των απαιτήσεων (requirements), ορολογίας η οποία ευρέως χρησιμοποιείται στο πεδίο της υπηρεσιακής αρχιτεκτονικής. Επιπροσθέτως, χρησιμοποιούμε τη διαρθρωτική προσέγγιση που παρέχει η επιχειρησιακή αρχιτεκτονική ώστε να κατηγοριοποιήσουμε τις διάφορες αιτιολογήσεις αποφάσεων του σχεδιασμού επιχειρησιακής αρχιτεκτονικής και να κάνουμε ρητές τις πιθανές διατομεακές (cross-domain) σχέσεις τους.

Η πρακτική εγκυρότητα του EA Anamnesis αξιολογείται μέσω της εφαρμογής του υπό πραγματικές συνθήκες σε δυο μελέτες περίπτωσης (case studies), μία σε ένα ερευνητικό και τεχνολογικό οργανισμό στο Λουξεμβούργο και μία σε ένα Ελληνικό οργανισμό ηλεκτρονικής διακυβέρνησης. Η αξιολόγηση υποδεικνύει ότι το EA Anamnesis συλλαμβάνει επαρκώς τις αιτιολογήσεις σχεδίασης των δύο μελετών περίπτωσης και ότι οι επαγγελματίες επιχειρησιακής αρχιτεκτονικής μπορούν να αναγνωρίσουν πως το πρόβλημα σχεδίασης διατυπώνεται και επιλύεται μέσω κατάλληλων αποφάσεων σχεδίασης. Επιπροσθέτως των πραγματικών μελετών περίπτωσης, επιδεικνύουμε το EA Anamnesis μέσω μίας πλασματικής μελέτης περίπτωσης από τον ασφαλιστικό τομέα. Αυτή η μελέτη περίπτωσης χρησιμοποιήθηκε κατά τη διάρκεια ανάπτυξης του εννοιολογικού μας πλαισίου για την επίδειξη των μεμονωμένων μερών του σχεδιαστικού δημιουργήματος. Η συγκριμένη επίδειξη μας επέτρεψε να λάβουμε αναπληροφόρηση (feedback) ώστε να επεκτείνουμε και να βελτιώσουμε το EA Anamnesis. Τέλος, χρησιμοποιούμε μία πρωτότυπη υλοποίηση (prototype) ώστε να αξιολογήσουμε τη δυνατότητα υλοποίησης του εννοιολογικού μας πλαισίου και για περαιτέρω επεκτάσεις και βελτιώσεις.

Το EA Anamnesis έχει συνέπειες αμφότερα στην πρακτική και στην έρευνα επιχειρησιακής αρχιτεκτονικής. Στην πρακτική, οι οργανισμοί μπορούν να χρησιμοποιήσουν το EA Anamnesis για την δημιουργία δεξαμενών γνώσης αιτιολόγησης σχεδιασμού, στις οποίες μπορούν να αποθηκεύονται οι αιτιολογήσεις, κίνητρα και ενδεχόμενες επιπτώσεις των αποφάσεων σχεδίασης. Με αυτόν τον τρόπο, οι επαγγελματίες επιχειρησιακής αρχιτεκτονικής, και ιδιαιτέρως οι νεοφερμένοι στον οργανισμό, μπορούν γρήγορα να προσεγγίσουν και να καταλάβουν την υπάρχουσα επιχειρησιακή αρχιτεκτονική και μπορούν καλύτερα να συντηρήσουν και να επεκτείνουν την αρχιτεκτονική λαμβάνοντας υπόψη πιθανές προβληματικές καταστάσεις και ευπάθειες της υπάρχουσας αρχιτεκτονικής.

Σε ότι αφορά τις ερευνητικές συνέπειες, το EA Anamnesis συνεισφέρει στο πεδίο της επιχειρησιακής αρχιτεκτονικής φανερώνοντας την ανάγκη για την υποστήριξη της επιχειρησιακής αρχιτεκτονικής με προσεγγίσεις αιτιολόγησης σχεδίασης και με την παρουσίαση μια προσέγγισης ειδικά σχεδιασμένης για αυτό τον σκοπό. Επιπλέον, συνεισφέρουμε στο ερευνητικό πεδίο αιτιολόγησης σχεδίασης, παρουσιάζοντας μία εννοιοποίηση διαδικασίας λήψης αποφάσεων η οποία είναι βασισμένη σε MCDA για την παροχή της αιτιολόγησης πίσω από αποφάσεις σχεδίασης. Τέλος, κάνουμε ρητή τη συνύφανση (intertwining) μεταξύ των χώρων προβλήματος και λύσης, συλλαμβάνοντας τον τρόπο με τον οποίο οι απαιτήσεις ενεργοποιούν την εκτέλεση νέων αποφάσεων σχεδίασης, τον τρόπο με τον οποίο οι απαιτήσεις αξιολογούνται κατά τη διάρκεια της διαδικασίας απόφασης και τον τρόπο με τον οποίο οι αποφάσεις σχεδίασης ή οι πιθανές τους απρόβλεπτες επιπτώσεις προκαλούν την εκμαίευση (elicitation) νέων απαιτήσεων.

Με την έρευνα μας κάναμε μία πρώτη απόπειρα να υποστηρίξουμε τον τομέα της υπηρεσιακής αρχιτεκτονικής με πληροφορία αιτιολόγησης σχεδιασμού. Κατά τη σύνταξη της παρούσας διατριβής, το EA Anamnesis έχει ήδη εμπνεύσει άλλους ερευνητές να εργαστούν προς αυτή την κατεύθυνση και έχει ήδη χρησιμοποιηθεί σαν βάση για την ανάπτυξη σχετικών επεκτάσεων. Παρόλα αυτά απαιτείται περισσότερη εργασία. Πιθανές κατευθύνσεις συμπεριλαμβάνουν επεκτάσεις για a-priori και συνεργατική υποστήριξη αποφάσεων, αναθεώρηση του εννοιολογικού πλαισίου σε ότι αφορά την ανάλυση προβλήματος εξαιτίας της σημασιολογικής εγγύτητας των εννοιών του και την έρευνα που αφόρα την απόδοση του EA Anamnesis σε σχέση με τις προσπάθεια που απαιτείται για τη σύλληψη αιτιολογήσεων σχεδιασμού.

# Curriculum Vitae

Georgios Plataniotis is a PhD candidate in Information Science at Radboud University Nijmegen, the Netherlands. During his doctoral studies he worked with prof. Hend*erik* Proper at the Luxembourg Institute of Science and Technology (LIST), Luxembourg. His research work was concerned with the rationalization of enterprise architectures and it was carried out in the context of the research project ACET (Architecture Coordination of Enterprise Transformations), a collaboration between LIST and University of St. Gallen, Switzerland. Georgios is currently working as a scientific officer at the e-Government Center for Social Security (IDIKA), Greece. He is involved the European Commission project EESSI (Electronic Exchange of Social Security Information) where he is analyzing (both in business and technical levels) the Greek social security landscape with regard to the exchange and coordination of social security information between Greece and other EU member states.

# The Enterprise Engineering Network

## Background

The Enterprise Engineering Network (EE Network, `www.ee-network.eu`) is a research and training network targeting PhD candidates and research fellows. Next to the supervision of PhD candidates and research fellows, the main activities of the network involve:

- Research seminars;
- Events targeting interaction with practitioners;
- Events targeting interaction with M.Sc. students;
- Development of a joint curriculum for EE Network researchers and associated courses;
- Co-organisation of scientific events.

The hosts of the network are also concerned with formulating and conducting joint research projects. Yet, the EE Network itself focuses on the actual training activities.

The history of the EE Network, and its direct predecessors, can be traced back to 2001. It is currently hosted at five locations:

1. **Headquarters:** IT for Innovation Services department of the Luxembourg Institute of Science and Technology, Belval, Luxembourg;
2. Model Based System Development department of the Institute for Computing and Information Sciences of Radboud University, Nijmegen, the Netherlands;

3. HAN University of Applied Science, Arnhem, the Netherlands;
4. Information Systems Architecture group of Utrecht University of Applied Science, Utrecht, the Netherlands;
5. Individual and Collective Reasoning and Model Driven Engineering groups of University of Luxembourg, Luxembourg, Luxembourg.

To enable a practical operation of the training activities, in particular in for the research seminars, the EE Network has a traditional geographical focus on the Rhine-Scheldt-Meuse-Moselle basin, which includes the Low Countries (Belgium, Netherlands and Luxembourg), the Rhineland in Germany, as well as Lorraine in France.

## Finished dissertations

Dissertations produced in the EE Network, and its direct predecessors, include:

**2017-1** G. Plataniotis, *EA Anamnesis – A Conceptual Framework for Enterprise Architecture Rationalization*, Radboud University Nijmegen, Nijmegen, the Netherlands, April 4, 2017

**2016-2** R. Ettema, *Using triangulation in Lean Six Sigma to explain quality problems - An enterprise engineering perspective*, Radboud University Nijmegen, Nijmegen, the Netherlands, December 14, 2016

**2016-1** H. Faller, *Organizational Subcultures and Enterprise Architecture Effectiveness: an Explanatory Theory*, Radboud University Nijmegen, Nijmegen, the Netherlands, March 4, 2016

**2015-2** L.J. Pruijt, *Instruments to Evaluate and Improve IT Architecture Work*, University of Utrecht, Utrecht, the Netherlands, November 25, 2015.

**2015-1** D.J.T. van der Linden, *Personal semantics of meta/concepts in conceptual modeling languages*, Radboud University Nijmegen, Nijmegen, the Netherlands, February 13, 2015.

**2014-2** C. Feltus, *Aligning Access Rights to Governance Needs with the Responsibility MetaModel (ReMMo) in the Frame of Enterprise Architecture*, University of Namur, Namur, Belgium, March 11, 2014.

**2014-1** F. Tulinayo, *Combining System Dynamics with a Domain Modeling Method*, Radboud University Nijmegen, Nijmegen, the Netherlands, January 27, 2014.

**2013-2** C. Brandt, *An Enterprise Modeling Framework for Banks using Algebraic Graph Transformation*, Technische Universität Berlin, Berlin Germany, December 20, 2013.

**2013-1** R. Wagter, *Enterprise Coherence*, Radboud University Nijmegen, Nijmegen, the Netherlands, November 19, 2013.

**2012-2** A. Nakakawa, *A Collaborative Process for Enterprise Architecture Creation*, Radboud University Nijmegen, Nijmegen, the Netherlands, November 21, 2012.

**2012-1** D. Ssebuggwawo, *Analysis and Evaluation of Modelling Processes*, Radboud University Nijmegen, Nijmegen, the Netherlands, November 21, 2012.

**2009-2** S. Overbeek, *Bridging Supply and Demand for Knowledge Intensive Tasks*, Radboud University Nijmegen, Nijmegen, the Netherlands, April 24, 2009.

**2009-1** J. Nabukenya, *Improving the Quality of Organizational Policy Making using Collaboration Engineering*, Radboud University Nijmegen, Nijmegen, the Netherlands, March 3, 2009.

**2006-1** B. van Gils, *Aptness on the Web*, Radboud University Nijmegen, Nijmegen, the Netherlands, March 3, 2006.

**2004-1** S.J.B.A. Hoppenbrouwers, *Freezing Language – Conceptualisation Processes across ICT Supported Organizations*, Radboud University Nijmegen, Nijmegen, the Netherlands, December 10, 2004.

# Part V

# Appendices

# Survey Study

**Title:**

Questionnaire on the usefulness and current practice of design rationale concepts in EA.

**Investigator:**

This research is carried out by Georgios Plataniotis (e-mail: georgios@plataniotis.eu), CRP Henri Tudor, Luxembourg

This questionnaire is part of a larger research project that aims to provide insight into building better rationale for architectural decisions in EA. In our view, design rationale should be captured systematically by means of alternatives, design decisions, the architecture domain/layer to which the decision applies, evaluation criteria etcetera. In the following questionnaire, we present a set of basic concepts that could potentially be used for design rationale in EA and we aim to identify whether these concepts are considered to 1) help with the maintenance of an EA, 2) help to justify an EA, and 3) be currently actively documented in the participant's organization.

**Concepts:**

**Rationale:**

- Please indicate whether you feel that capturing rationale (reasoning behind design decisions) helps with the **maintenance** of the EA.

- ☐ Disagree
- ☐ Agree
- ☐ Strongly agree

- Please indicate whether you feel that capturing rationale (reasoning behind design decisions) helps with the **justification** of the EA.
  - ☐ Disagree
  - ☐ Agree
  - ☐ Strongly agree

- Please indicate whether the current practice in your organization is to **document** rationale (reasoning behind design decisions).
  - ☐ Disagree
  - ☐ Agree
  - ☐ Strongly agree

**Rejected alternatives:**

- Please indicate whether you feel that capturing rejected alternatives helps with the **maintenance** of the EA.
  - ☐ Disagree
  - ☐ Agree
  - ☐ Strongly agree

- Please indicate whether you feel that capturing rejected alternatives helps with the **justification** of the EA.
  - ☐ Disagree
  - ☐ Agree
  - ☐ Strongly agree

- Please indicate whether the current practice in your organization is to **document** rejected alternatives.
  - ☐ Disagree
  - ☐ Agree
  - ☐ Strongly agree

**EA layer:**

- Please indicate whether you feel that capturing the EA layer of design decisions helps with the **maintenance** of the EA.

  ☐ Disagree
  ☐ Agree
  ☐ Strongly agree

- Please indicate whether you feel that capturing the EA layer of design decisions helps with the **justification** of the EA.

  ☐ Disagree
  ☐ Agree
  ☐ Strongly agree

- Please indicate whether the current practice in your organization is to **document** the EA layer of design decisions.

  ☐ Disagree
  ☐ Agree
  ☐ Strongly agree

**Unanticipated observed impact:**

- Please indicate whether you feel that capturing the unanticipated observed impact of design decisions helps with the **maintenance** of the EA.

  ☐ Disagree
  ☐ Agree
  ☐ Strongly agree

- Please indicate whether you feel that capturing the unanticipated observed impact of design decisions helps with the **justification** of the EA.

  ☐ Disagree
  ☐ Agree
  ☐ Strongly agree

- Please indicate whether the current practice in your organization is to **document** the unanticipated observed impact of design decisions.

  ☐ Disagree
  ☐ Agree
  ☐ Strongly agree

**Design decision traceability:**

- Please indicate whether you feel that capturing the traceability of design decisions helps with the **maintenance** of the EA.

  ☐ Disagree
  ☐ Agree
  ☐ Strongly agree

- Please indicate whether you feel that capturing the traceability of design decisions helps with the **justification** of the EA.

  ☐ Disagree
  ☐ Agree
  ☐ Strongly agree

- Please indicate whether the current practice in your organization is to **document** the traceability of design decisions.

  ☐ Disagree
  ☐ Agree
  ☐ Strongly agree

**Documentation practice of design decisions:**

- Does your organization use a standardized template for documenting design decisions?

  ☐ Not aware
  ☐ Yes
  ☐ No

- If your organization does not use a standardized template, why not?

  ☐ Not useful
  ☐ No time / budget
  ☐ No suitable tool
  ☐ Other ______________________________

Enterprise transformations impose socio-technical changes to organizations. Enterprise architecture (EA) is acknowledged as a steering instrument that assists stakeholders in the process of an enterprise transformation. Amongst others, the practice of EA is supported by modeling languages which describe an enterprise holistically. By doing so, they show an enterprise's business products and services, and how these are realized by IT infrastructure and applications.

However, EA modeling languages lack the capability to capture the design rationale that led to specific architectural designs in the context of an enterprise transformation. This lack of transparency regarding design decisions can cause design integrity issues when architects have to maintain or change the current EA design. Due to this lack of insight into the rationale, new designs are constructed in an ad-hoc manner, without taking into account considerations and constraints implied by past design decisions.

This thesis addresses this lack of design rationale support for the domain of EA, by introducing EA Anamnesis, a conceptual framework for EA design rationalization. EA Anamnesis complements existing EA modeling languages with design rationalization information. This is realized by capturing rationale such as the formulation of the design problem across the EA, how the problem was addressed by specific design decisions, the reasoning behind the selection of those decisions and their possible unanticipated consequences, and by linking that rationale with elements of the EA design.

ISBN 978-94-6299-566-6