View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Radboud Repository

Radboud Repository

Radboud University Nijmegen

PDF hosted at the Radboud Repository of the Radboud University Nijmegen

The following full text is a preprint version which may differ from the publisher's version.

For additional information about this publication click this link.
http://hdl.handle.net/2066/143774

Please be advised that this information was generated on 2018-07-07 and may be subject to change.

Traceability and Modeling of Requirements in Enterprise Architecture from a Design Rationale Perspective

Georgios Plataniotis, Qin Ma, Erik Proper
Luxembourg Institute of Science and Technology, Luxembourg
Radboud University Nijmegen, Nijmegen, the Netherlands
EE-Team, Luxembourg, Luxembourg
{georgios.plataniotis, qin.ma, erik.proper}@list.lu

Sybren de Kinderen
University of Luxembourg, Luxembourg, Luxembourg
EE-Team, Luxembourg, Luxembourg
sybren.dekinderen@uni.lu

***Abstract*—Our work aims to rationalize Enterprise Architectures (EA) by providing the reasoning behind the designs, in terms of selection criteria, design alternatives and more. Its major contribution is a formal metamodel that captures the reasoning and the inter-relationships of design decisions.**

This paper extends our approach in order to provide an explicit bridging between the Problem space that is defined by the different requirements and the Solution space that is described by specific design decisions. In doing so, EA Anamnesis also supports traceability from specific design decisions to the given requirements.

***Keywords*—Enterprise Architecture, Design Rationale, Functional requirements, Nonfunctional requirements**

I. Introduction

In our earlier work [1], [2], we introduced the EA Anamnesis approach for architectural rationalization. EA Anamnesis captures decision characteristics, such as decision criteria and used decision making strategies, and shows the relation between Business-level and IT-level decisions. Furthermore, EA Anamnesis relies on a metamodel-based formal linkage between EA modeling languages (mainly ArchiMate) and the corresponding decision aspects to realize the connection between EA designs and EA rationale.

In this paper we enhance the EA Anamnesis metamodel to provide an explicit linkage with requirements engineering concepts. By doing so we enable backward / forward traceability from design decisions to the given requirements and vice versa, and the capability to identify if the architectural decisions are aligned to the essential enterprise architecture requirements. The refinement of the metamodel was based on existing approaches from the requirements engineering domain.

II. Requirements and EA Anamnesis

In this section we introduce briefly an insurance case study (Section II-A), and subsequently we use it to illustrate our metamodel (Section II-B). 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

A. Case study

ArchiSurance is an insurance company that sells insurance products using a direct-to-customer sales model and now wants to change its selling model to intermediary sales. John, an enterprise architect, is hired to execute the enterprise transformation. Due to space limitation we zoom in to an architectural scenario for the selection of an application system which would support the newly introduced intermediary business process. John had to select between Commercial Off-The-Shelf (COTS) Application A and COTS Application B. For a detailed description of the case study please refer to [1].

B. Metamodel

Figure 1 presents the enhanced metamodel. The metamodel is comprised by two parts: Problem space and Solution space. Problem space includes the Requirement concept and its subclasses Functional requirement and Nonfunctional requirement. The Requirement concept provides description of the given enterprise architectural problem. The solution space includes the existing concepts of EA Anamnesis and rationalizes enterprise architecture decisions.

As we can see, functional and nonfunctional requirements are members of both solution and problem space. The idea is that requirements are used as a bridge between the problem and the solution space and in order to cross this bridge we have to move from generic, high abstraction level requirements, to more refined ones. On the one hand the generic requirements describe how the enterprise architect formulates the given architectural problem, and on the other hand the more refined requirements provide the rationalization behind specific design decisions.

We now describe the problem space concept Requirement and its' relationships. Due to space limitations we leave out the solution space concepts of EA Anamnesis. For a detailed description of these concepts please refer to [1].

The Enterprise Engineering Team (EE-Team) is a collaboration between Public Research Centre Henri Tudor, Radboud University and HAN University of Applied Sciences (`www.ee-team.eu`)

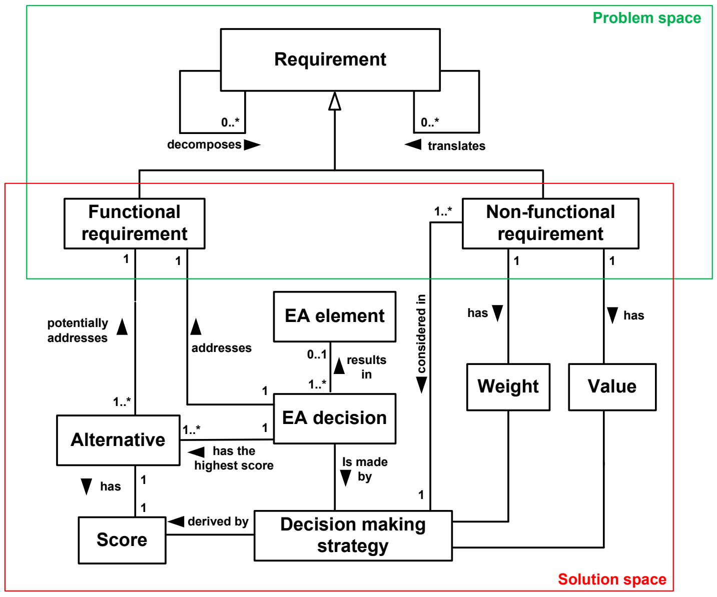

Fig. 1. Enhanced metamodel

Requirement: A requirement is defined as a statement of need, condition or capability that should be realized by a system [3], [4]. They are usually derived from goals and they should be compliant with the given enterprise architecture principles [4], [5]. Requirements can range from generic ones until very specific descriptions of system functions [3]. The refinement of generic requirements to specific ones varies depending the architectural layer of the enterprise. For example, the generic requirement 'Security' can be refined differently to business layer ('Task delegation') and technology layer ('Encryption') of the enterprise.

Requirement types:

a) Functional requirements: Functional requirements specify **what** the system should do or in other words a specific behavior that a system must have [3].

Example: For our architectural scenario John has to support the business process with an application system. The following functional requirement is captured through our approach: 'Find an appropriate application to interface with the intermediary'.

b) Nonfunctional requirements: Nonfunctional requirements specify the behavioral aspects of a system or in other words the quality criteria that determine **how** the system works.

Example: John selected COTS Application B based on the nonfunctional requirement 'Interoperability'.

Requirement relationships:

Based on our previous work [1] we have defined 2 different types of relationships between the requirement concept:

a) Decomposition: As we can see in Figure 1 the requirement can be decomposed further until we arrive to a desired abstraction level for the domain architecture. For our approach this means that we can decompose the problem until we arrive to design decisions that lead to a concrete EA element. The decomposition relationship is in line with 'decomposes into' relationship of Kruchten's ontology [6].

Example: The generic nonfunctional requirement 'IT Systems Adhere to Open Standards' which defines a general behavior of the enterprise architecture is decomposed in the specific nonfunctional requirement 'Interoperability' in the application layer. As we mentioned before 'Interoperability' was used to evaluate the application system alternatives.

b) Translation: Translation relationship describes how requirements that belong to different EA layers are translated in requirements of a different layer [7]. This relationship is critical for the domain of enterprise architecture since it provides to the enterprise architect a holistic view of the cross layer dependencies of the requirements.

Example: The functional requirement 'Establish business process intermediary' in the business layer of ArchiSurance was translated to the functional requirement 'Find an appropriate application to interface with the intermediary' in the application layer.

III. Conclusion

In this paper we presented a metamodel for capturing the the influence of functional and nonfunctional requirements on the decision making processes in enterprise architecture.

For future research we intend to extend further our approach in order to also cover the role of enterprise architecture principles during the decision making process.

Acknowledgments.

This work has been partially sponsored by the *Fonds National de la Recherche Luxembourg* (`www.fnr.lu`), via the PEARL programme.

References

[1] G. Plataniotis, S. de Kinderen, and H. A. Proper, "Ea anamnesis: An approach for decision making analysis in enterprise architecture," *International Journal of Information System Modeling and Design (IJISMD)*, to appear.

[2] ——, "Capturing decision making strategies in enterprise architecture – a viewpoint," in *Enterprise, Business-Process and Information Systems Modeling*, ser. Lecture Notes in Business Information Processing. Springer Berlin Heidelberg, 2013, vol. 147, pp. 339–353. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-38484-4_24

[3] K. Pohl, *Requirements engineering: fundamentals, principles, and techniques*. Springer Publishing Company, Incorporated, 2010.

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

[5] D. Greefhorst, H. Proper, and G. Plataniotis, "The dutch state of the practice of architecture principles," *Journal of Enterprise Architecture–November*, p. 21, 2013.

[6] P. Kruchten, P. Lago, and H. Vliet, "Building up and reasoning about architectural knowledge," in *Quality of Software Architectures*, ser. Lecture Notes in Computer Science, C. Hofmeister, I. Crnkovic, and R. Reussner, Eds. Springer Berlin Heidelberg, 2006, vol. 4214, pp. 43–58.

[7] G. Plataniotis, S. d. Kinderen, and H. A. Proper, "Relating decisions in enterprise architecture using decision design graphs," in *Proceedings of the 17th IEEE International Enterprise Distributed Object Computing Conference (EDOC)*, 2013.