EA Anamnesis: An Approach for Decision Making Analysis in Enterprise Architecture

Georgios Plataniotis, Public Research Centre Henri Tudor, Luxembourg & Radboud University Nijmegen, the Netherlands & EE-Team, Luxembourg*[1]

Sybren de Kinderen, University of Luxembourg, Luxembourg & EE-Team, Luxembourg

Henderik A. Proper, Public Research Centre Henri Tudor, Luxembourg & Radboud University Nijmegen, the Netherlands & EE-Team, Luxembourg

ABSTRACT

Enterprise Architecture (EA) modeling languages can express the business-to-IT-stack for an organization, showing how changes in the IT landscape impact business aspects and vice versa. Yet EA languages provide only the final architectural design, not the rationale behind this design.

In earlier work, we presented the EA Anamnesis approach for EA rationalization. We discussed how EA Anamnesis forms a complement to current EA modeling languages, showing for example design alternatives, EA artifact selection criteria and the decision making strategy that was used.

In this paper, we extend EA Anamnesis with a capability for organizational learning. In particular, we present an integration of two viewpoints presented in earlier work: (1) an ex-ante decision making viewpoint for rationalizing EA during decision making, which for example captures a decision and its anticipated consequences, and (2) an ex-post decision making viewpoint, which for example captures the unanticipated decision consequences, and possible adjustments in criteria.

We use a fictitious, yet realistic, case study to illustrate our approach.

Keywords: Enterprise Architecture, Design Rationale, Organizational learning, Decision Design Graph

1. INTRODUCTION

Enterprise Architecture (EA) languages are considered as an instrument for expressing an enterprise holistically (Lankhorst, 2005), linking perspectives on an organization that are usually considered in isolation. In doing so, one can consider the organization-wide impact of a change

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

(Lankhorst, 2005; Op't Land, Proper, Waage, Cloo and Steghuis, 2008), expressing its complete business-to-IT-stack (Aier and Winter, 2009). For example, for a newly introduced IT application, EA modeling languages can be used to consider implications on business processes, human resources, organizational goals, and more.

While EA modeling languages can be used to express the holistic design of an organization, they do not express the design decisions behind the resulting models. 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). This means that, without design rationale, one leaves implicit design criteria, reasons for a design, and design alternatives.

As a result of lacking rationalization architects are unable to justify their designs (Tang, Jin and Han, 2005). Furthermore new designs are constructed in an ad hoc manner without taking into consideration constraints implied by past design decisions (Tang et al., 2007). Here, constraints refer to boundaries implied by the design. These boundaries can be of a business or of a technical nature, such as the choice for a programming language implied by choosing a particular application environment.

Moreover, a survey amongst enterprise architecture practitioners (Plataniotis, 2013c) provides indications of the usefulness of design rationalization for motivating design decisions, and for architectural maintenance. At the same time, however, the survey shows that practitioners often forego the use of a structured template/approach when rationalizing an architecture, relying instead on ad hoc information capturing in tools such as Microsoft Office.

In earlier work (Plataniotis, de Kinderen and Proper, 2012; Plataniotis, de Kinderen and Proper, 2013a; Plataniotis, de Kinderen and Proper, 2013b), we introduce the EA Anamnesis approach for architectural rationalization. EA Anamnesis derives from the ancient greek word ανάμνησις (/ ænæm"ni:sIs/), which denotes memory and repair of forgetfulness. Our earlier work addresses two separate viewpoints for architectural rationalization: ex-ante and ex-post. On the one hand, (Plataniotis et al., 2013a) addresses ex-ante decision rationalization. In so doing, it focuses on concepts relevant during or just after a decision making process, such as decision criteria and the anticipated consequences of a decision. On the other hand, (Plataniotis et al., 2013b) focuses on a reflection on past decisions. This ex-post viewpoint focuses on concepts relevant some time after the decision making process, such as unanticipated decision consequences and a possible subsequent change in the importance score of criteria.

In this paper, we focus on relating the ex-ante and ex-post viewpoints of EA Anamnesis. We do so because these viewpoints naturally complement each other. For one, by a comparison of ex-post unanticipated consequences to ex-ante decision criteria and anticipated consequences, we can inform future decision making. In so doing, we essentially provide a good first step in using EA Anamnesis for "organizational learning" (cf. Conklin, 1996) in the area of decision making.

As such, the main contribution of this paper is twofold: (1) to integrate two viewpoints that we considered previously in isolation, (2) to show how this integration can be useful for learning from past decision making. For now, we focus our approach on a single decision maker: the enterprise architect.

At this point, we would like to note that existing rationalization approaches in software architecture, such as (IEEE, 2005; Kruchten, 2004; Tang et al., 2007), do not consider business issues, such as decisions related to business processes. Furthermore, while the EA language ArchiMate 2.0 (The Open Group, 2012) has a motivational layer, it lacks concepts important for rationalization such as considered alternatives, decision criteria, etc. As such, ArchiMate 2.0 is not a suitable language for architectural rationalization. We discuss this in further detail in Section 2.

This paper is structured as follows. Section 2 presents the related work in design rationale, decision strategies and challenges, Section 3 introduces our central artifact, a metamodel for capturing decision making. Section 4 illustrates the usefulness of EA Anamnesis with an insurance industry example. Finally Section 5 concludes.

2. RELATED WORK

In this section, we position our work against the existing body of knowledge. This section reports on the stream of existing work in (1) design rationale as well as (2) decision making strategies, and (3) factors that influence decision making.

2.1 Design rationale

Concerning **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, as pointed out by Shipman III and McCall (Shipman and McCall, 1997), argumentation-based approaches are in practice not suitable for capturing design rationales. Mainly this is because argumentation-based approaches require extensive documentation (Shipman and McCall, 1997). Furthermore, argumentation-based approaches lack formality, and thus are not amenable to the computer-based support that we aim for with EA Anamnesis. Also 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 artifacts (such as elements of architectural languages).

Moreover, **goal-oriented requirements engineering approaches** (Elahi and Yu, 2012; Horkoff and Yu, 2012; Liaskos, 2011) propose mechanisms for decision analysis and prioritization of requirements. However, requirements engineering approaches deal with the problem space of an architecture. Despite the fact that some concepts from goal-oriented modeling can be used to describe design rationales, goal-oriented concepts are more generic. For one, a goal can denote a high level, strategic, goal (for example, "make more profit") pertaining to the problem space. However, a goal can also denote very specific, attribute-level, criteria pertaining to the solution-space, such as the criterion "have good usability" for a software application.

Differently, in the words of (Savolainen, 1999), design rationale approaches such as EA Anamnesis focus on the "solution space". The solution space comes after the translation of high-level goals into more specific ones (which (Savolainen, 1999) refers to as the requirements space), and before the specific design (the design space). Since we position our work in the solution space, we opt to borrow concepts from existing rationalization approaches. This is because they provide a set of concepts that is specifically tailored to our purposes (such as criteria, alternatives, and artifact), as opposed to the more generic problem-oriented concepts from *goal-oriented* approaches.

Furthermore, since its second version the **Archimate EA modeling language** has a motivation extension and an implementation and migration extension. The motivation extension is used to model the reasons behind architectural changes, but lacks concepts common to existing rationalization approaches. For example, the motivational extension does not capture design alternatives, the used decision making strategy, or unanticipated consequences of decisions. Furthermore the Implementation and Migration Extension deals with the project management and the planning of enterprise architecture changes and as such, is not well suited for architectural rationalization.

Finally, there exist **design rationale approaches for Software Architecture** (Jansen and Bosch, 2005; Kruchten, 2004; Savolainen, 1999; Tang et al., 2007; Tyree and Akerman, 2005). These approaches are template based or model based. Akin to argumentation-based rationalization approaches, template based approaches (Savolainen, 1999; Tyree and Akerman, 2005) describe in textual format elements of

Architectural Decisions such as Rationale, Issue, Implications and etc. Differently, model based approaches (Jansen and Bosch, 2005; Kruchten, 2004; Tang et al., 2007) provide a formal metamodel of decision rationalization concepts, thus enabling computer-processable rationalization.

However rationalization approaches from software architecture focus on software issues (such as code documentation). Yet these issues are different from those in Enterprise Architecture (Coggins and Speigel, 2007). For one, as part of their responsibilities enterprise architects concern themselves with the business-to-IT-stack (Aier and Winter, 2009). Here, for example, they analyze impacts of changes in IT infrastructure to business processes and vice versa. Thus enterprise architects deal with different, cross-organizational, issues compared to software architects, who deal mostly with software concerns.

2.2 Decision making strategies

Decision making strategies generally fall in two main categories: compensatory and noncompensatory (Einhorn, 1970; Payne, 1976; Rothrock and Yin, 2008; Svenson, 1979). We briefly explain these strategies with a car buying example. In this example, a customer selects a car based upon the criteria "color of car', "carbon emission', "size of car' and "gasoline consumption'.

In compensatory decision making (Einhorn, 1970), alternatives are evaluated exhaustively, taking all criteria and their trade-offs into consideration. Criteria with high values compensate for criteria with lower values. Finally, the alternative with the highest score is selected. For our car buying example, this implies that a customer considers all four criteria "color of car", "carbon emission', "size of car" and "gasoline consumption". For example: s/he can state that "color of car" is of high importance, and "carbon emission", "size of car" and "gasoline consumption" are of less. By doing so the customer then selects a car that best complies with all these criteria.

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 criteria, and they are time consuming (Einhorn, 1970).

Noncompensatory strategies (Einhorn, 1970), on the other hand, are consistent with the concept of bounded rationality. This means that the rationale to make a decision is limited by factors such as hard constraints, time constraints and the cognitive load of the decision maker. As such, noncompensatory strategies evaluate alternatives heuristically, using only a limited number of criteria and trade-offs.

Considering a noncompensatory decision strategy for our car buying example, let us now assume that the customer lives in the city and selects between two cars: a small and a big one. Now, "size of car" is a hard constraint for the customer due to the limited parking space available in the city. Therefore, regardless of the criteria "carbon emission", "color of car", and "gasoline consumption" s/he excludes the big car from her/his choice set.

The main characteristics of noncompensatory strategies are twofold: (1) they reduce the decision making effort, (2) they are not demanding regarding the information needed to make the decision. As such, it is a common practice for decision makers to use noncompensatory strategies in situations whose limitations affect the decision making process (Payne, Bettman and Johnson, 1993). Example situations include time stress or hard constraints. However, by definition decision makers do not take all criteria into account when using noncompensatory decision strategies.

Last but not least, in some cases the use of a combination of compensatory and noncompensatory decision strategies (a hybrid) is required (Elrod, Johnson and White, 2004; Jeffreys, 2004; Rothrock and Yin, 2008). For example, a decision maker starts his evaluation process by excluding alternatives that do not meet certain noncompensatory criteria, and only thereafter evaluates the remaining alternatives with a compensatory strategy.

2.3 Decision making strategies

Decision making, in particular the choice for a decision making strategy, is influenced by factors such as time, information completeness, cognitive load of the decision maker, and more (Alenljung and Persson, 2008).

We describe briefly those factors and how they influence the decision making process.

Time stress: One of the most common situations in decision making processes is time stress (Orasanu and Connolly, 1993). Decision makers under time pressure must take critical decisions. Usually these decisions are made last moment, in an adhoc manner, and without sufficient rationalization.

Ill-structured problems: A decision problem is often complex in terms of cause-effect relations, correlations and feedback loop between relevant factors (Orasanu and Connolly, 1993). As such, decision problems are difficult to understand, also in terms of the impacts and outcomes that they have.

Information incompleteness: meaning that in practice information can be ambiguous or even missing (Orasanu and Connolly, 1993).

Shifting, ill-defined, or competing goals: A decision maker can have conflicting goals during the decision making process. Decision maker should weight appropriately each of these goals in making a decision (Ruhe, 2003).

Action and feedback loops: Decision making contains a series of loops that the decision maker should deal with (Orasanu and Connolly, 1993). Early mistakes and poor information generate decisions that should be reexamined. For example violations in architectural design are not disclosed early enough and this implies that the decision making process should be repeated.

High stakes: Decision makers have also to cope with high risk decisions, especially when the problem they are called to solve is of high importance (Orasanu and Connolly, 1993).

Multiple player situations: When multiple stakeholders are involved in a decision making process the situation gets more complicated (Orasanu and Connolly, 1993). Stakeholders have different interests, goals and expectations from a specific decision. Another common issue is the lack of shared understanding between stakeholders for a particular problem. Multiplayer situations may result in delays in the decision making possibly leading to a revision of the decision, with high cost impact (Regnell, 2001).

Organizational goals and norms: Organizations operate under specific goals and norms (Orasanu and Connolly, 1993). Decision makers should make decisions in the context of these goals and norms and should avoid making decision based only on their personal preferences. Decision makers, regardless of their personal preferences, must evaluate alternatives with criteria that the organization sets.

Note that, for now, our approach does not provide rationalization support for decision strategies with multiple decision makers. Currently this paper focuses on single decision maker environments.

3. THE EA ANAMNESIS APPROACH

In this section we discuss the basic motivations behind architectural rationalization with EA Anamnesis. Thereafter we introduce an insurance case (in Section 3.2) which is used as a running case throughout out this paper. Subsequently, we use the insurance case to illustrate our integrated EA Anamnesis metamodel (section 3.3). 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 (The Open Group, 2012), which in turn is based on a real insurance company.

3.1 Organizational learning with EA Anamnesis

As stated in the introduction, this paper integrates two decision making meta models: one for ex-ante decision rationalization, and one for ex-post decision rationalization. Here, ex-ante and ex-post refer to different, but complementary, perspectives on decision making.

On the one hand, ex-ante decision making concepts rationalize decision making during or just after the decision making process. In so doing, ex-ante rationalization focuses on criteria deemed important during decision making, and captures *anticipated* consequences of decisions. On the other hand, rationalization of ex-post decision making provides a reflection of *unanticipated* consequences of a decision some time after the decision has been made.

We consider ex-ante and ex-post rationalization as complementary, and posit that a comparison of the two allows for *organizational learning* (Conklin, 1996). For one, we foresee that an unanticipated consequence of a decision (ex-post), when compared to decision criteria and anticipated decision consequences (ex-ante), results in having to re-prioritize criteria and in having to revisit anticipated consequences. Thus, we argue that the comparison of ex-ante to ex-post informs future decision making. This reflects the purpose of EA Anamnesis: to have an organizational memory that helps organizations to learn from decision making.

3.2 Illustrative case

ArchiSurance is an insurance company that sells car insurance products using a direct-to-customer sales model. It does so to reduce its operations and product costs.

Figure 1 presents the partial (business and application layers) ArchiSurance direct-to-customer sales model, modeled with the EA modeling language Archi-Mate. ArchiMate is an Open Group[2] standard EA modeling language (Lankhorst, 2005; The Open Group, 2012). Two business services support the sales model of ArchiSurance: "Car insurance registration service" and "Car insurance service". Here, in accordance with the ArchiMate specification (The Open Group, 2012), a business service refers to the functionality offered to the environment by a business process or collaboration. For example, in this way ArchiMate separates between the functionality "Car insurance registration service", and how this functionality is actually realized.

ArchiMate helps us understand the dependencies between different perspectives of an enterprise. For example, in Figure 1 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". Note here that an application service denotes the functionality offered by an application. Thus, akin to an ArchiMate business service, an application service denotes functionality *independently* of how this functionality is realized.

Although disintermediation reduces operational costs, it also increases the risk of incomplete or faulty risk profiles of customers (Cummins and Doherty, 2006). This so called "adverse selection of profiles" (cf. Cummins and Doherty, 2006) leads insurance companies to calculate unsuitable premiums or, even worse, to wrongfully issue insurances to customers.

To reduce adverse selection of profiles ArchiSurance decides 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 *John*, and external architect, is hired by ArchiSurance to help guide the change to an intermediary sales model.

[2] See http://www.opengroup.org/standards/ea

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 translation of the new business process "Customer profile registration" to EA artifacts in the application layer. The resulting ArchiMate model is depicted in Figure 2.

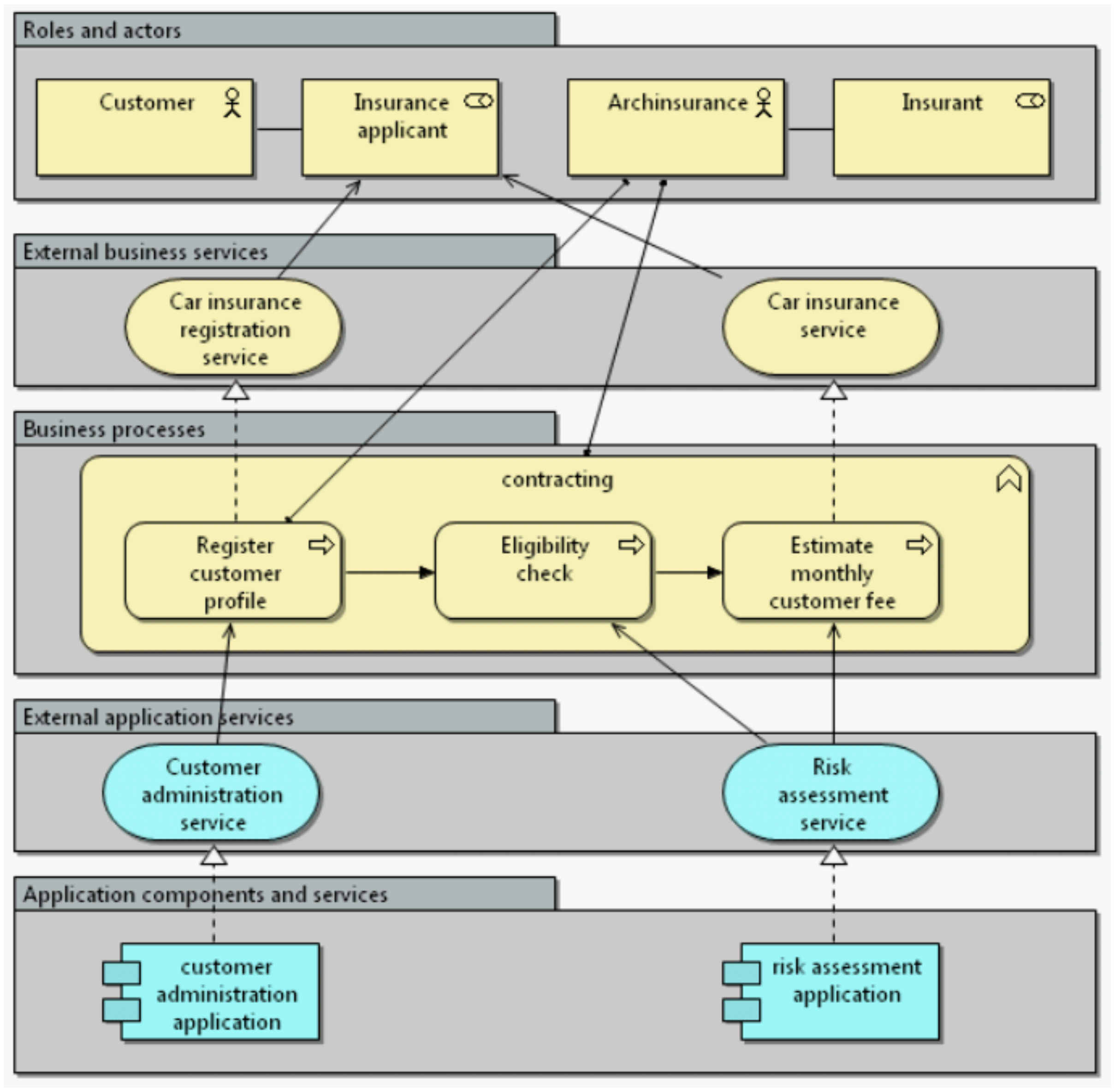
Roles and actors
Customer
Insurance applicant
Archinsurance
Insurant
External business services
Car insurance registration service
Car insurance service
Business processes
contracting
Register customer profile
Eligibility check
Estimate monthly customer fee
External application services
Customer administration service
Risk assessment service
Application components and services
customer administration application
risk assessment application

Figure 1. ArchiSurance direct-to-customer EA model

Here we see for example how a (new) business process "customer profile registration", owned by the insurance broker (cations "customer administration service intermediary" and "customer administration service ArchiSurance". The new business process "customer profile registration" is a collaboration between the insurance company and the intermediary (symbolized by the double semicircle).

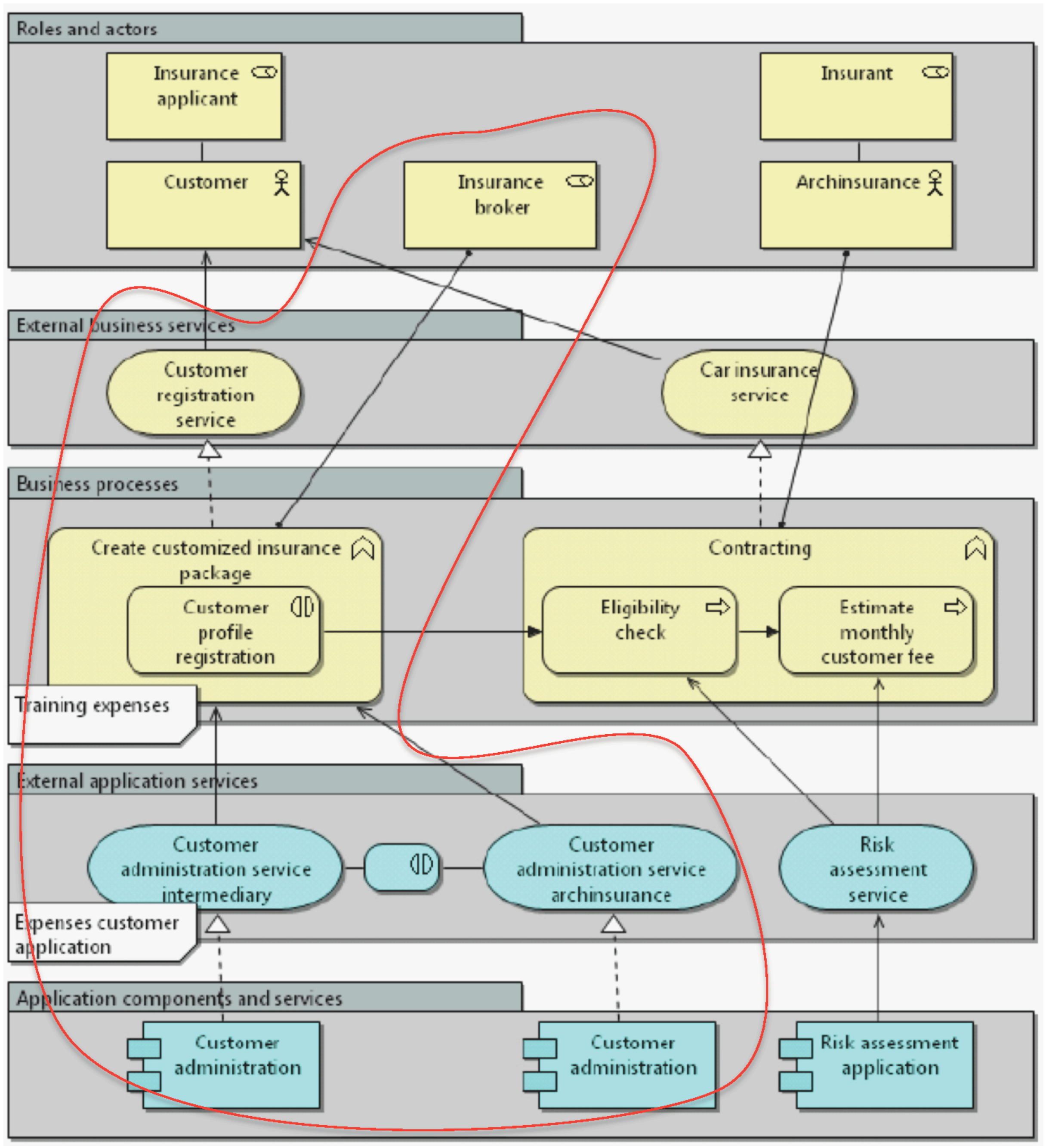
Roles and actors
Insurance applicant
Customer
Insurance broker
Insurant
Archinsurance
External business services
Customer registration service
Car insurance service
Business processes
Create customized insurance package
Customer profile registration
Contracting
Eligibility check
Estimate monthly customer fee
Training expenses
External application services
Customer administration service intermediary
Customer administration service archinsurance
Risk assessment service
Expenses customer application
Application components and services
Customer administration
Customer administration
Risk assessment application

Figure 2. ArchiSurance direct-to-customer EA model

3.3 The EA Anamnesis metamodel

Figure 3 presents the EA Anamnesis metamodel. This metamodel is an integration of two metamodels presented in earlier work: the EA decision making metamodel (Plataniotis et al., 2013a), and the decision relationships metamodel (Plataniotis et al., 2013b). With this integration, we allow for (1) contextualizing the decision making process of a single decision in terms of cross cutting/intertwining decision relationships, and (2) a comparison of decision outcomes to the original decision making process.

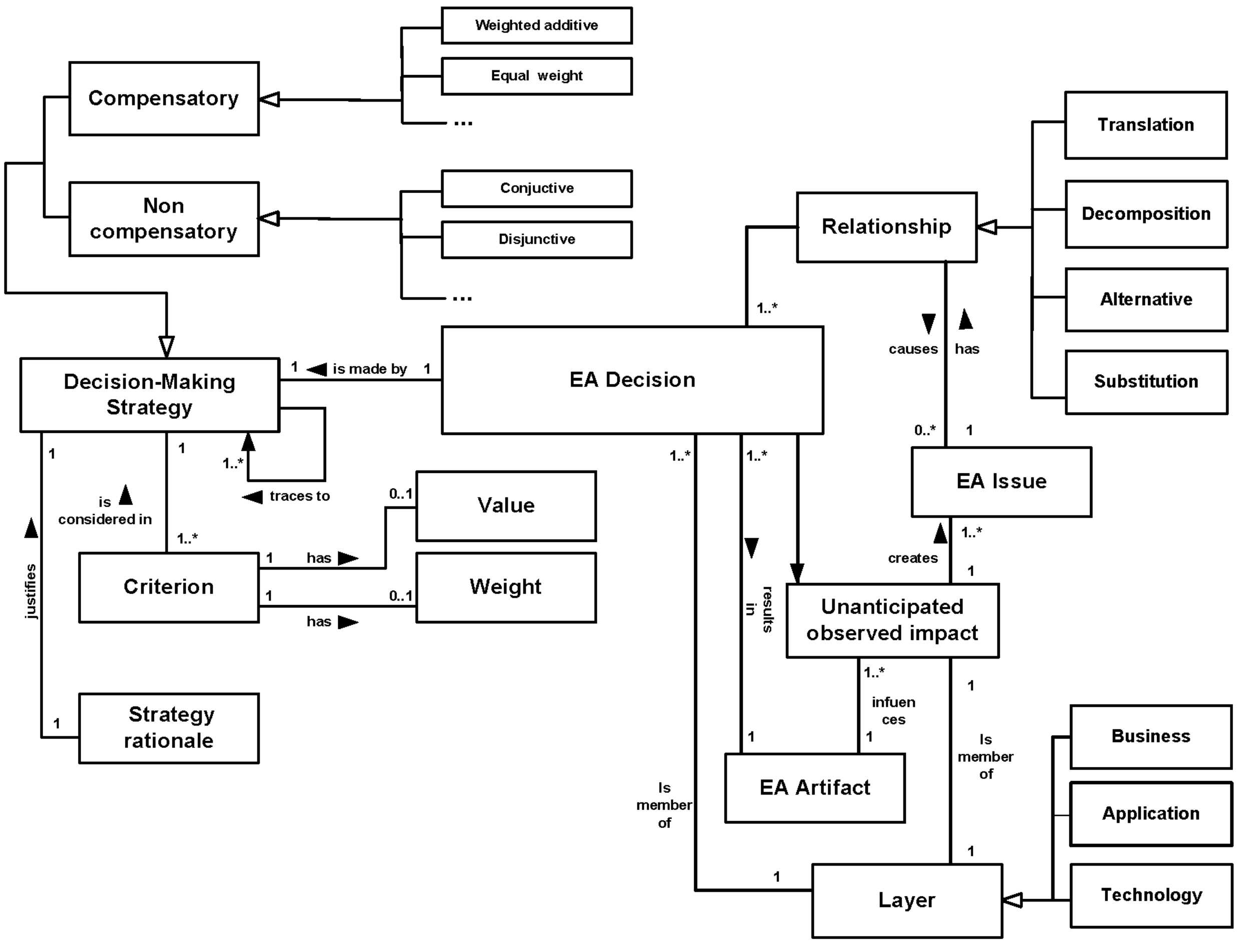

Figure 3. ArchiSurance direct-to-customer EA model

For comprehension purposes the concepts of our metamodel will be introduced in 3 sections: decision properties (Section 3.3.1), decision making process concepts (Section 3.3.2) and decision relationships (Section 3.3.3). Furthermore, we use the ArchiSurance case for illustration purposes.

3.3.1 Decision properties

EA decision: An EA decision represents the actual design decision that was made after an evaluation process. (Proper and Op't Land, 2010; Orasanu and Connolly, 1993).

Example: John makes the EA decision "make customer profile registration via intermediary".

EA issue: Similar to the concept of an issue from (Tyree and Akerman, 2005), an EA issue represents the architectural design problem that enterprise architects have to address.

Example: The EA issue "create an appropriate application service to support new business process" resulting from the EA decision "introduce a new business process for customer profile registration".

EA artifact: An EA artifact (similar to concept of an architecture element (Tang et al., 2007)) is either the direct result produced from a set of executed EA decisions, or a representation of this result. For now, we use an EA artifact to refer to architectural representations. Specifically, we use it as a bridging concept towards the EA modeling language ArchiMate, whereby an EA artifact allows us to link EA decisions to concepts from ArchiMate.

Example: The EA artifact "Customer profile registration" in the ArchiMate model in Figure 2 is linked to, amongst others, the EA Decision "Make customer profile registration via intermediary".

Layer: In line with the ArchiMate language (The Open Group, 2012), an enterprise is specified in three layers: *Business, Application and Technology*. Using these three layers, we express an enterprise *holistically*, showing not only applications and physical IT infrastructure (expressed through the application and technology layers), but also how an enterprise's IT impacts/is impacted by an enterprise's products and services and its business strategy and processes.

Example: The EA decision "make customer profile registration via intermediary" is a part of the business layer of ArchiSurance.

Unanticipated observed impact: This concept signifies an *unanticipated* consequence of an already made decision to an EA artifact. This opposes to anticipated consequences, as indicated by relationships such as translation or decomposition.

In current everyday practice, architects model *anticipated* consequences using what-if-scenarios (Lankhorst, 2009). Unfortunately, not every possible impact of made EA decisions can be predicted. This is especially true for enterprise architecture, where one considers impacts across the enterprise rather than in one specific (e.g. technical) part. The outcome of EA decisions can be observed during an ex-post analysis of the architecture (Baron and Hershey, 1988; Proper and Op't Land, 2010). Some of the consequences of EA decisions are revealed during the implementation phase, or during the maintenance of the existing architecture design. These unanticipated consequences are captured exactly by the concept of an unanticipated observed impact.

For us the main usefulness of capturing unanticipated observed impacts is that they can be used by architects to avoid decisions with negative consequences in future designs of the architecture.

Example: The EA decision "Acquisition of COTS application B" has an unanticipated observed impact "Degraded user experience in the application use". This observed impact captures an unanticipated, ex-post, side effect of acquiring COTS application B, due to unfamiliarity of users with the new user interface that COTS application B introduces.

3.3.2 Decision making process concepts

The decision making process concepts of our metamodel focus on capturing (1) decision making strategies that were used during the architectural design process for a specific EA decision, (2) the rationale behind this specific decision strategy choice, and (3) available alternatives and criteria. Below we provide the description of these concepts.

Decision-Making Strategy: This concept captures the decision making strategy used by the enterprise architect to (1) evaluate the alternatives, and make the actual EA decision. As we mentioned in Section 2.2, decision strategies are characterized as compensatory, noncompensatory, or as a hybrid of these two.

In our metamodel, we specify this as follows:

Compensatory strategy

- *Weighted additive (WADD):* In WADD strategies the criteria which evaluate the alternatives have different weights. The score of each alternative is computed by multiplying each criterion by its weight and then by taking the sum of these values. The alternative with the highest score is chosen by the decision maker (Rothrock and Yin, 2008).

- *Equal weight*: The score of each alternative is calculated by the same way as WADD strategies. The difference is that criteria have the same weight (Rothrock and Yin, 2008).

Noncompensatory strategy

- *Conjunctive*: In conjunctive strategies, alternatives that fail to comply with one or more criteria, are immediately excluded from the decision maker's choice set (Rothrock and Yin, 2008).
- *Disjunctive:* In this strategy an alternative is selected if it complies with a criterion, irrespective on its values on other criteria (Rothrock and Yin, 2008).

Noncompensatory strategies are appropriate for evaluating alternatives in situations of information incompleteness and absence of numerical data. For example, the "car color" criterion can not be quantified. Alternatives which do not comply with this criterion can be eliminated from the choice set because they do not comply with this feature.

We should also mention that there is no restriction in the use of additional decision strategies. We include a set of common decision strategies, but we also denote in the metamodel that more decision strategies can be supported.

"Traces to" relationship: In line with Section 2.2, a hybrid decision strategy is supported by our metamodel. The relationship "traces to" signifies the combination of two or more decision strategies during the decision making process.

Example: John rejects "acquisition of COTS app C" because it exceeds the budget set on beforehand by top management. Thus, here John employed a conjunctive non compensatory decision making strategy.

Criterion: A criterion is an attribute that, to a larger or lesser extent, should be satisfied during the decision making process (Elrod et al., 2004). Criteria are used for both compensatory or noncompensatory decision making strategies. For example, if a disjunctive strategy was used, a decision is taken on compliance with one or more criteria. Furthermore, the concepts **value** and **weight** of a criterion are included in our viewpoint. The value concept represents the value that the decision maker assigns to this criterion during the evaluation process, which indicates how well an artifact performs on a particular criterion. The weight concept represents the importance of this criterion, and is typically used in WADD strategies.

Example: After discarding "acquisition of COTS app C", John considers 3 quality criteria his evaluation, "usability", "interoperability" and "scalability". "Interoperability" was considered as the most important, with a weight of 10. Furthermore, "COTS application C" has the value "7" (out of 10) on the criterion "interoperability", whereas "COTS application B" has the value "3" out of 10. This difference in value reflects that "COTS application C" performs better than "COTS application B"on the criterion "interoperability".

Strategy rationale: In a decision making process, the architect not only has to choose amongst some alternatives (actual decision making process), but has also to select the decision strategy that satisfies his current evaluation needs. Actually, this concept represents the rationale for the decision strategy that was selected for the evaluation process. This is what is referred as metadecision making, decision making about the decision process itself (Mintzberg, 1976).

As we discussed in Section 2.3, different factors affect the decision making process and decision makers should adjust their decision making strategy accordingly. The concept of a strategy rationale enables a decision maker to justify the reasons for his metadecision. We argue that time stress can be another important factor to capture a strategy rationale. By capturing that a decision was made under time stress, we reason that decision makers can protect themselves. For example a decision maker can justify that a "bad" decision was made due to time constraints on the decision making process

Example: The strategy rationale "time stress" to select a non compensatory (heuristic) decision strategy. "Time stress" is a strategy rationale, or metadecision, since it concerns decision making about the decision making process itself, independently of specific decision criteria such as "usability".

3.3.3 EA Decision relationships

The role of relationship concepts is to make the different types of relationships between EA decisions explicit. Based on ontologies for software architecture design decisions (Kruchten, 2006; Tyree and Akerman, 2005), we define four types of relationships:

Translation relationship: Translation relationships illustrate relationships between decisions/EA issues that belong to different layers or different EA artifacts. In accordance with the metamodel (Figure 3) a translation relationship relates decisions with decisions, and decisions with issues. The translation relationship bases on the generic "is related to" relationship from (Kruchten, 2004), only we adapt it to the basic terminology of Architects. One of the key tasks for architect is, by communication, to translate the requirements that new EA artifacts impose (EA issue) to decisions that will support these requirements by means of another EA artifact (Op't Land and Proper, 2007; Strano and Rehmani, 2007).

Example: The EA decision "make customer profile registration via intermediary" translates to the issue "find an appropriate application service". Subsequently this issue translates to a second EA decision "acquisition of COTS application B".

Decomposition relationship: The Decomposition relationship is in line with "Comprises (Is Made of, Decomposes into)" of Kruchten's ontology (Kruchten, Lago and Vliet, 2006). Decomposition relationships signify how generic EA decisions decompose into more detailed design decisions.

Example: The EA decision "acquisition of COTS application B" has a decomposition relationship with EA decision "Application interface type 1". This is to indicate that choosing application B also implies the more detailed choice for a particular type of user interface.

Alternative relationship: The alternative relationship type (Kruchten et al., 2006), illustrates the EA decisions that were rejected (alternatives) in order to address a specific EA issue.

Example: Rejected EA decisions "COTS application A", "COTS application C" and "Upgrade existing application (inhouse)" have an alternative relationship with EA issue "find an appropriate application to interface with the intermediary". This signifies that these decisions were the alternatives for this issue.

Substitution relationship: A substitution relationship explicates how one EA decision repairs the negative outcome of another EA Decision.

Example: The EA decision "Acquisition of COTS application B" has a negative unanticipated observed impact on the business process "Customer profile registration". This is because it leads users to make mistakes, as we observed the concept "unanticipated observed impact". As such, it is repaired by the EA decision "Application interface 2".

4. USING EA ANAMNESIS FOR ORGANIZATIONAL LEARNING

We now illustrate how EA Anamnesis approach can support enterprise architects to improve their decision making capabilities based on decision making processes and their outcomes from past cases. Section 4.1 shows how we use the EA Anamnesis metamodel to capture EA decisions, exemplified by our running ArchiSurance case. Subsequently Section 4.2 introduces a procedural model for using the EA anamnesis metamodel for organizational learning, using the archisurance case for illustation purposes.

4.1 Capturing a decision making process

For the purposes of this paper we assume that John, the enterprise architect for ArchiSurance, is a single decision maker that is capable of identifying specific alternatives and criteria. Furthermore, we assume that he has full information to evaluate them.

Recall (from section 3.2) that John supports ArchiSurance in changing to selling car insurance via intermediaries. To start the decision making process based on the requirements of selling via intermediaries, John defines the criteria that the new application should satisfy (the criteria for application selection are grounded in (Jadhav and Sonar 2009).

For our illustrative example, John considers that the most important criteria are “usability”, “interoperability” and “scalability”. Based on these criteria he identifies 4 alternatives to choose from, 3 alternative Commercial Off-The-Shelf (COTS) applications and 1 alternative to upgrade the existing application in house.

Let us also assume that John receives a constraining budget limitation of €10000 for the acquisition of new IT systems.

John is now faced with a hybrid decision strategy: on the one hand, he wants to carefully evaluate the four alternatives on the criteria “usability”, “ interoperability” and “scalability” (via a compensatory strategy), but on the other hand he has to account for the hard constraint “budget limitation” (via a non-compensatory strategy).

At this point John uses the EA Anamnesis approach to capture and justify his strategy selection, as well as the alternatives and criteria of his decision problem. For the noncompensatory part, John wants to discard all alternatives that fail to meet the cost criterion. Because of this hard constraint, he chooses a conjunctive noncompensatory strategy (for an explanation, see Section 3.3.2) to exclude from his choice set alternatives that exceed the maximum value of this criterion.

Table 1 summarizes the score of each alternative. “COTS application C” is eliminated from the choice set because it fails to meet the maximum cost requirement.

Table 1. EA decision 13 noncompensatory conjunctive strategy

Alternatives	Cost	score
COTS A	€9000	1
COTS B	€8000	1
COTS C	€12000	0
Upgrade app	€5000	1

As discussed in section 3.3.2, conjunctive noncompensatory strategies evaluate alternatives using a threshold level on one or more criteria. In this example the conjunctive criterion is “cost”. The alternatives “COTS A”, “COTS B” and “Upgrade application” comply with this criterion (Table 1) and will be evaluated further in the next step of the decision making process. “COTS C” cost exceeds the maximum limit and is eliminated from the choice set. For noncompensatory strategies, alternative scores are boolean data types, they either comply or not with some criteria. The scores of the alternatives are also captured by our metamodel.

For the compensatory part, John evaluates the three remaining alternatives based on the value and the weight of each criterion. “Scalability” is the most important factor because, according to John, the application should be able to support changes in the business processes of ArchiSurance, for example, to support the addition of extra intermediaries.

Given the fact that the criteria that evaluate alternatives have different weights, John selects the use of a weighted additive compensatory strategy. Here, John captures again his decision strategy as well as the weights and the values of the compensatory criteria. The score of each alternative is calculated by multiplying the value of each criterion by its weight, and then by summing up these values. Here, the weights range from 1 – not important to 10 – important.

Table 2 shows us: (1) the criteria. "Scalability", the most important criterion for John has a weight of 10, while "usability" and "interoperability" have weights 2 and 5 respectively, (2) the score on a particular criterion for each alternative. For example: the alternative "COTS B" scores 9 on "scalability", whereas "Upgrade app" scores 4. (3) the total score of each alternative. For example: "COTS B" receives the highest score and as such, is selected by John.

Table 2. EA decision 13 compensatory weighted additive strategy

Alternatives	**usability**	**interoperability**	**scalability**	**score**
COTS A	7x2	7x5	7x10	119
COTS B	8x2	3x5	9x10	121
Upgrade app	9x2	5x5	4x10	83

Figure 4 depicts the captured decision making processes based on our approach

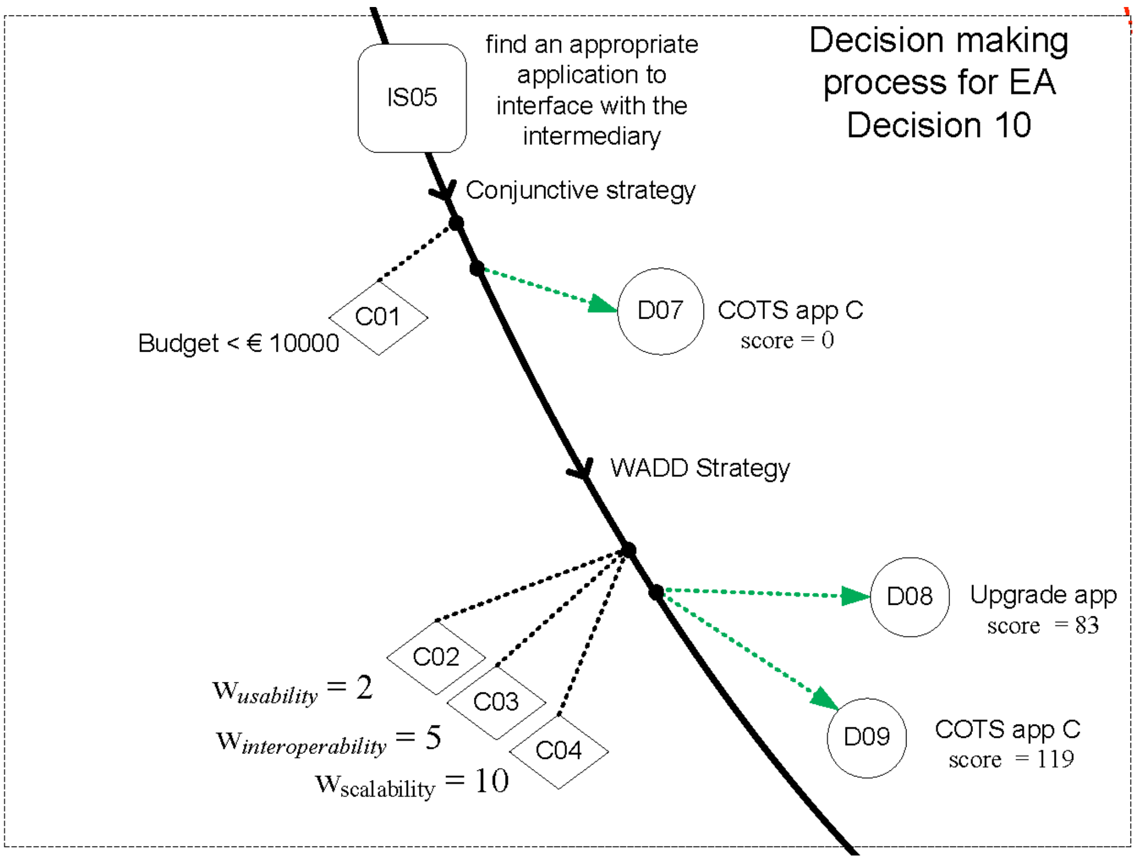

Figure 4. Decision making strategies for EA Decision 10

4.2 Using EA Anamnesis for organizational learning

In this section, we introduce a procedural model (Figure 5) for using EA anamnesis for organizational learning. We illustrate its steps by means of the archisurance case and we show how the integrated metamodel of ex-ante and ex-post perspectives assists architects in learning from past decision making. The steps that the enterprise architect follows in order to get support from our approach are derived from this procedural model. To visualize the metamodel for ArchiSurance we rely on Decision Design Graphs (DDGs), a concrete syntax for the EA Anamnesis metamodel that we introduced in [8]. Finally, a textualized table (Table 3) provides summarized information for EA decisions derived from our metamodel. Here, Table 3 as a "tooltip" for the DDGs.

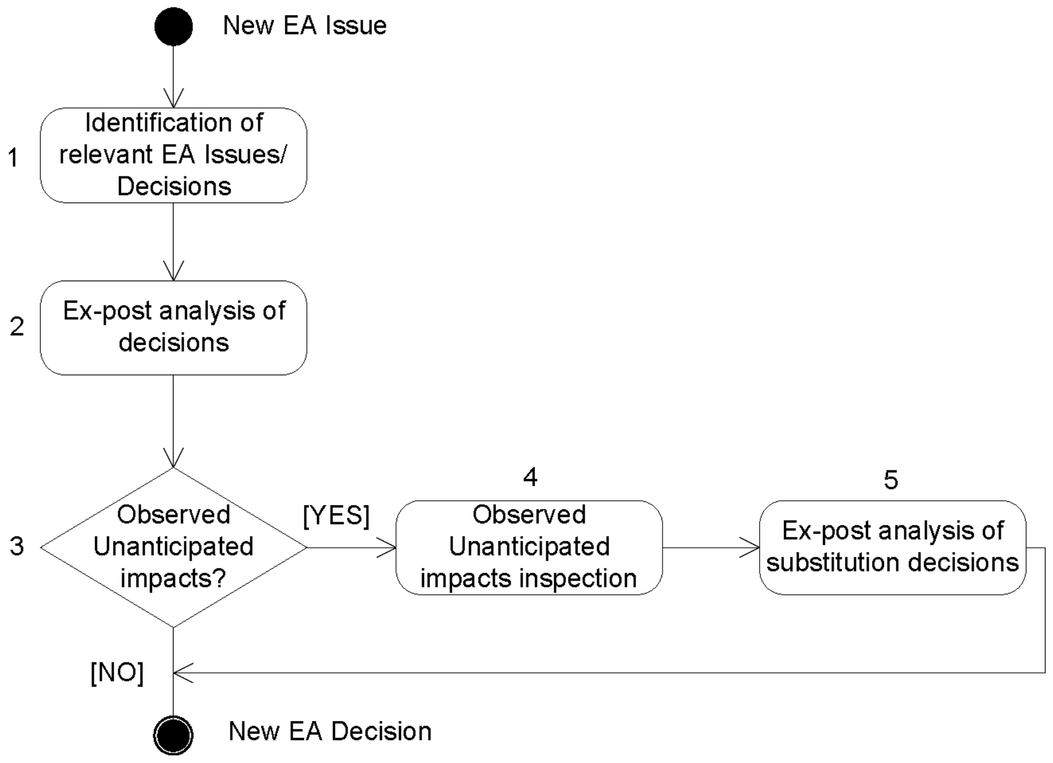
New EA Issue
1
Identification of relevant EA Issues/ Decisions
2
Ex-post analysis of decisions
3
Observed Unanticipated impacts?
[YES]
4
Observed Unanticipated impacts inspection
5
Ex-post analysis of substitution decisions
[NO]
New EA Decision

Figure 5. Organizational learning procedural model

Table 3. EA decision 10 summary

	Acquisition of COTS application B
EA issue:	EA Issue 05: Create an appropriate application service to support new business process
Decision Maker:	John
Layer:	Application
Relationships:	D05: introduce application service A D13: introduce user interface 3
Alternatives:	COTS application A COTS application C Upgrade existing application
Criteria:	usability, interoperability, scalability, cost
Observed Impact:	Observed Impact 01: Reduced performance of customer registration service business process

Note here that Figure 6 depicts the rationalization for the final intermediary design depicted in Figure 2.

Case: We now move two years forward to further illustrate how EA anamnesis supports future decision making. The customer profiles of ArchiSurance's car insurance are better calculated by using insurance intermediaries. As a result, ArchiSurance's management decides to also rely on intermediaries for the remainder of its insurance products. Bob is made responsible for translating the use of intermediaries into an appropriate enterprise architecture design. He uses ArchiMate to design the "to be" architecture.

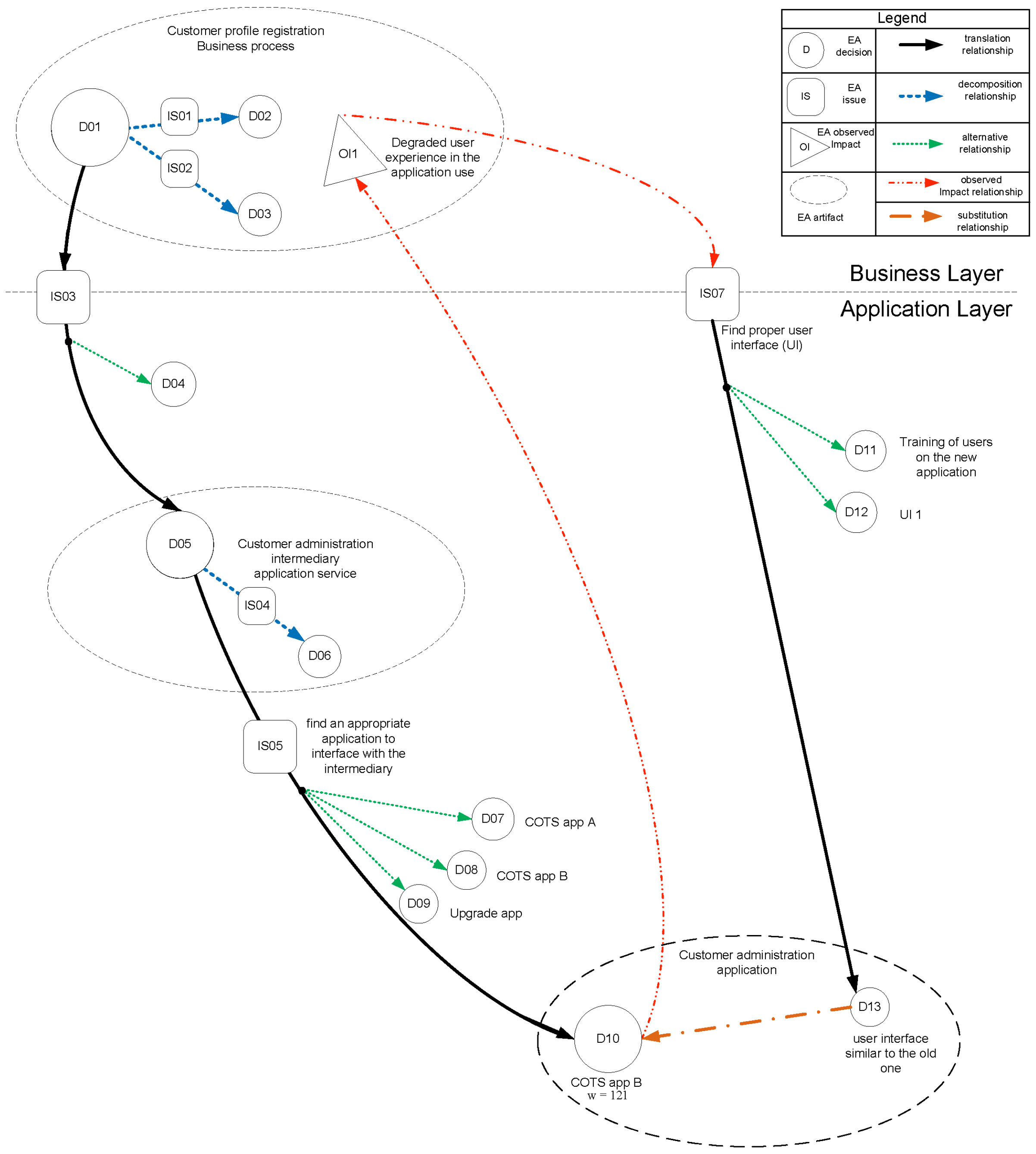
Legend
D
EA decision
translation relationship
IS
EA issue
decomposition relationship
OI
EA observed Impact
alternative relationship
observed Impact relationship
EA artifact
substitution relationship
Customer profile registration Business process
D01
IS01
D02
IS02
D03
OI1
Degraded user experience in the application use
Business Layer
Application Layer
IS03
IS07
Find proper user interface (UI)
D04
D11
Training of users on the new application
D12
UI 1
D05
Customer administration intermediary application service
IS04
D06
IS05
find an appropriate application to interface with the intermediary
D07
COTS app A
D08
COTS app B
D09
Upgrade app
Customer administration application
D13
D10
user interface similar to the old one
COTS app B
w = 121

Figure 6. EA decisions relationships visualization

Step 1: Identification of relevant EA issues/decisions. In this step, for those issues that the architect wants to address, we identify similar issues in the rationalization information stored in EA anamnesis.

Case: For the sake of example, we assume that Bob wants to know how his predecessor has supported the introduction of an intermediary with software applications. To this end, he can rely on rationale information regarding a similar issue captured by his predecessor. For example: In the EA ananmensis for ArchiSurance, Bob identifies the "Create an appropriate application service to support new business process" (issue 03), which addresses "Make customer profile registration via intermediary" (decision 01).

Step 2: Ex-post analysis of decisions. In this step we analyze decision details, without considering the unanticipated observed impacts.

Consider the issue "Create an appropriate application service to support new business process" (issue 03), as depicted in Figure 6 and for which we provide a "tooltip" in Table 3. For the issue "Create an appropriate application service to support new business process", Bob can amongst others see that the decision "Acquisition of COTS application B" was made, pertaining to the EA artifact "Customer administration application". Furthemore, Bob observes the alternatives "Acquistion of COTS application A", "Aquisition of COTS application C", and "Upgrade existing application", and used criteria, such as "usability" (in line with the information capturted in Section 4.1).

Step 3: Observed unanticipated impacts. In this step, we review if there are unanticipated observed impacts for the identified issues.

Step 4: Observed unanticipated impacts inspection. In this step we compare the unanticipated observed impact to decision details.

Case: Bob inspects "Customer administration application" from the decision "Acquisition of COTS application B" for Observed unanticipated impacts. He finds out that users had difficulties in using the new application, as signified by the unanticipated observed impact 01 "Degraded user experience in the application use" (OI1 in Figure 6). As such, Bob sees that EA Decision "Acquisition of COTS application B" (D10) had a negative observed impact on the business process "Customer profile registration". At this point, Bob can trace back to "Acquisition of COTS application B" (D10, as summarized in Table 3). From the inspection of the decision making analysis in Table 2, Bob realizes that the criterion usability, which is related with the observed unanticipated impact of user experience, was not considered as a criterion of high importance.

Step 5: Ex-post analysis of substitution decisions. In this step we analyze how the unanticipated observed impact was addressed.

Case: Using the DDG (Figure 6), Bob observes how John addressed the observed impact "Degraded user experience in the application use" (OI1). First Bob finds that OI1 leads to the new EA Issue "Find proper user interface" (IS07). Furthermore, from the DDG Bob notices that there are three possible alternatives for "Find proper user interface" (1) "Training of users on the new application" (EA decision 11), (2) "New User Interface 1" (EA decision 12) and, (3) "replace of existing application interface with an interface similar to the old one" (EA decision 13). Finally, Bob notices that his predecessor John decided for the third option (the executed EA decision 13) and rejected the other two (i.e., that John rejected EA decisions 11, 12).

In summary, from the analysis of past EA Decisions and their implications for the enterprise architecture, Bob can consider for his current decision making problem decision making strategies, alternatives and criteria as well as their relative importance from past decision making processes. Based on the outcomes of decisions, he can avoid decisions or problems that may come along. For example, if Bob has to address the issue of an appropriate application for the new intermediary business process, he will be aware of the possible implication “Degraded user experience in the application use”. To address this issue he can apply the same approach as his predecessor or he can inform the interested stakeholders regarding this concern.

5. CONCLUSIONS AND FUTURE WORK

In this paper we introduced a metamodel and corresponding visualization for capturing Enterprise Architecture (EA) decision making strategies decision making strategies and a procedural model for using the metamodel. Together, the integrated metamodel and produceral model allow for the comparison of a decision making process with observed outcomes of a decision. This comparison of (ex-ante) decision making with (ex-post) observed impact leads to a better understanding of existing architectures. As such, it provides a first step towards learning from architectural decision making.

For future research, first and foremost we intend to confront the illustrative examples of our approach to enterprise architecture practitioners. From our survey amongst enterprise architecture practitioners (Plataniotis, de Kinderen, van der Linden, Greefhorst and Proper, 2013), we already have indications about the perceived practical usefulness of architectural rationalization - particularly in terms of justification and maintenance. However we require case studies to further support those results.

In addition, we will look into how our approach deals with legacy artifacts, meaning artifacts available prior to the introduction of a rationalization approach (such as EA anamnesis). On the one hand, this is relevant as such legacy artifacts also imply constraints on the future design. On the other hand we do foresee a challenge in reverse engineering rationalization for such artifacts, as – even more so than with design rationalization of current artifacts – the relevant design knowledge may be incomplete or even completely missing.

Furthermore, we aim to provide procedural decisional guidance for using our approach inspired by Decision Support Systems (DSS) literature. While we feel that this paper taken a good first step towards capturing, representing and using a rationale, we deem further guidance especially necessary to scale up our approach for use in real-life domains, whereby many decisions are taken. Here, a good starting point is Silver (Silver, 1991), who introduces a typology of different types of DSS and their respective characteristics. For example: a DSS that provides support during (ex-ante) decision making has different characteristics than a DSS that (ex-post) reflects upon already captured information. This typology forms useful input for further structuring procedural guidance.

Last but not least, one of our major challenges is to investigate the return of capturing effort for our approach. Our design rationale assists architects to better understand existing EA designs, but the effort of capturing this information might be a dissuasive factor. To address this issue our research will focus on ways to decrease the capturing effort. One way of doing this is by evaluating the actual practical usefulness of the concepts of the decision making strategy viewpoint. For example we capture the strategy rationale for selecting a decision making strategy, but whether the effort for capturing this outweighs the received benefits remains to be seen. Furthermore, by further formalization, we foresee that in a future iteration we are better able to automatically compare factors relevant during decision making with those emerging some time after the decision has been made. For example: to express a delta in a criterion as an explicit result of an unanticipated consequence. This is opposed to the current manual procedure expressed in Figure 5.

REFERENCES

Aier, S., Winter, R. (2009). Virtual decoupling for it/business alignment -conceptual foundations, architecture design and implementation example. *Business & Information Systems Engineering 1*, 150–163

Alenljung, B., Persson, A. (2008). Portraying the practice of decision-making in requirements engineering: a case of large scale bespoke development. *Requirements engineering 13*, 257–279.

Baron, J., Hershey, J.C. (1988). Outcome bias in decision evaluation. *Journal of personality and social psychology* 54 (569).

Burge, J., Brown, D.C. (2000). Reasoning with design rationale. In: *Artificial Intelligence in Design*, (pp. 611–629). Springer.

Coggins, C., Speigel, J. (2007). The methodology for business transformation v1.5: A practical approach to segment architecture. *Journal of Enterprise Architecture.*

Conklin, J. (1996). Designing organizational memory: preserving intellectual assets in a knowledge economy. *Group Decision Support Systems 1*, 362.

Cummins, J., Doherty, N. (2006). The economics of insurance intermediaries. *Journal of Risk and Insurance 73*, 359–396.

Einhorn, H. (1970).The use of nonlinear, noncompensatory models in decision making. *Psychological bulletin 73*, 221–230

Elahi, G., Yu, E. (2012). Comparing alternatives for analyzing requirements trade-offs–in the absence of numerical data. *Information and Software Technology 54*, 517–530.

Elrod, T., Johnson, R., White, J. (2004). A new integrated model of noncompensatory and compensatory decision strategies. *Organizational Behavior and Human Decision Processes 95*, 1–19.

Horkoff, J., Yu, E. (2012). Comparison and evaluation of goal-oriented satisfaction analysis techniques. *Requirements Engineering*, 1–24.

Jadhav, A., Sonar, R. (2009). Evaluating and selecting software packages: A review. *Information and software technology 51*, 555–563.

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

Jeffreys, I. (2004). The use of compensatory and non-compensatory multi-criteria analysis for small-scale forestry. *Small-scale Forestry 3*, 99–117.

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

Kruchten, P., Lago, P., Vliet, H. (2006). Building up and reasoning about architectural knowledge. In Hofmeister, C., Crnkovic, I., Reussner, R., eds.: Quality of Software

Architectures. *Volume 4214 of Lecture Notes in Computer Science.* (pp. 43–58). Springer Berlin Heidelberg.

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

Lankhorst, M. (2009). Enterprise architecture at work: Modelling, communication and analysis. Springer.

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

Liaskos, S., McIlraith, S.A., Sohrabi, S., Mylopoulos, J. (2011). Representing and reasoning about preferences in requirements engineering. *Requirements Engineering 16*, 227–249.

Mintzberg, H., Raisinghani, D., Theoret, A. (1976). The structure of unstructured decision processes. *Administrative science quarterly,* 246–275.

Op't Land, M., Proper, H.A. (2007). Impact of principles on enterprise engineering. *ECIS 2007 Proceedings*.

Op't Land, M., Proper, E., Waage, M., Cloo, J., Steghuis, C. (2008). Enterprise architecture: creating value by informed governance. Springer.

Orasanu, J., Connolly, T. (1993). The reinvention of decision making. *Decision making in action: Models and methods*, 3–20.

Payne, J. (1976). Task complexity and contingent processing in decision making: An information search and protocol analysis. *Organizational behavior and human performance 16*, 366–387.

Payne, J., Bettman, J., Johnson, E. (1993). The adaptive decision maker. Cambridge University Press.

Plataniotis, G., de Kinderen, S., Proper, H.A. (2012) EA Anamnesis: Towards an Approach for Enterprise Architecture Rationalization. In Proceedings of the 2012 workshop on Domain-specific modeling. DSM '12, New York, NY, USA, ACM (2012) 27-32

Plataniotis, G., de Kinderen, S., Proper, H.A. (2013). Capturing decision making strategies in enterprise architecture - a viewpoint. In *Nurcan, S., Proper, H., Krogstie, J., Schmidt, R., Halpin, T., Bider, I., eds.: Enterprise, Business-Process and Information Systems Modeling. Volume 147 of Lecture Notes in Business Information Processing* (pp. 339–353). Springer Berlin Heidelberg.

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

Plataniotis, G., de Kinderen, S., van der Linden, D., Greefhorst, D., Proper, H.A. (2013b). An empirical evaluation of design decision concepts in enterprise architecture. In: *Proceedings of the 6th IFIP WG 8.1 working conference on the Practice of Enterprise Modeling* (PoEM 2013).

Proper, H.A., Op't Land, M. (2010). Lines in the water. In Harmsen, F., Proper, H.A., Schalkwijk, F., Barjis, J., Overbeek, S., eds. Practice-Driven Research on Enterprise Transformation. *Vol. 69 of Lecture Notes in Business Information Processing* (pp. 193–216). Springer Berlin Heidelberg.

Regnell, B., Paech, B., Aurum, A., Wohlin, C., Dutoit, A., och Dag, J. (2001). Requirements mean decisions!–research issues for understanding and supporting decision-making in requirements engineering. In: *First Swedish Conference on Software Engineering Research and Practise: Proceedings*, Citeseer.

Rothrock, L., Yin, J. (2008). Integrating compensatory and noncompensatory decision-making strategies in dynamic task environments. *Decision Modeling and Behavior in Complex and Uncertain Environments*, 125–141.

Ruhe, G. (2003). Software engineering decision support: methodology and applications. *Innovations in decision support systems 3*, 143–174.

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

Shipman, F.M., McCall, R.J. (1997). Integrating different perspectives on design rationale: Supporting the emergence of design rationale from design communication. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*: AI EDAM, 141.

Silver, M.S. (1991). Decisional guidance for computer-based decision support. *MIS Quarterly*, 105–122.

Strano, C., Rehmani, Q. (2007). The role of the enterprise architect. *Information Systems and e-Business Management 5*, 379–396.

Svenson, O. (1979). Process descriptions of decision making. *Organizational behavior and human performance 23*, 86–112.

Tang, A., Jin, Y., Han, J. (2007). A rationale-based architecture model for design traceability and reasoning. *Journal of Systems and Software 80*, 918–934.

The Open Group (2012). ArchiMate 2.0 Specification. Van Haren Publishing.

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

Tyree, J., Akerman, A. (2005). Architecture decisions: Demystifying architecture. *Software, IEEE 22*, 19–27.