Conceptual Models for Cross-cutting Aspects in Enterprise Architecture Modeling

Sabine Buckl, Florian Matthes, Christian M. Schweda
Chair for Informatics 19
Technische Universität München
E-mail: {buckls,matthes,schweda}@in.tum.de

Abstract—**The benefit of enterprise architecture (EA) management is directly coupled to the underlying conceptualization of the enterprise. This conceptualization should reflect the goals pursued by the EA management endeavor and focus on the areas of interest of the involved stakeholders. Whereas this statements often goes as a matter of course, an enterprise willing to develop such a goal-adequate conceptualization, finds itself confronted with a plethora of different approaches typically proposing a one-size-fits-it-all model, which neglects the subject of enterprise-specificity. These models are typically described in an object-oriented manner utilizing concepts like *class, property*, and *relationship*. Specific aspects relevant in the area of EA management, such as temporality, property-dependency, and cross-cutting aspects are more often than not neglected in this approaches.**

In this paper, we addresses the challenge of conceptualizing cross-cutting aspects in EA modeling. Cross-cutting aspects refer to EA-related concepts, like goals or projects, which may exert influence on other concepts of the EA model. Therefore, the state-of-the-art in EA conceptualization is revisited resulting in the identification of four cross-cutting aspects of EA modeling: *goals, lifecycles, projects,* and *standards*. Along these aspects we raise questions concerning a suitable alternative modeling language for EA conceptualizations and discuss options of future research.

Index Terms—**enterprise architecture, cross-cutting aspects, conceptual modeling**

I. Introduction and motivation

Today's enterprises find themselves confronted with a broad variety of challenging environmental factors that make an integrated approach to business and IT indispensable. Getting an holistic view on what has in recent years been called *enterprise architecture* (EA) is a complicate task. The enterprise willing to model its EA has to cope with two challenges. Firstly, it must define procedures for gathering the needed information. Secondly, the enterprise must devise a conceptual model defining the necessary information. The first challenge might be the more obvious one, as each enterprise forms a specific organizational context that requires dedicated information gathering procedures, etc. Nevertheless, also the conceptual model defining what the using enterprise regards an EA to be, differs from organization to organization. This may be ascribed to the fact that different enterprises pursue different goals from an integrated perspective. The rich literature on EA management, i.e. the management discipline concerned with the EA, enumerates a plurality of such goals, ranging from "increasing transparency" [1], [2] over "increase standardization" [3], [4], [2], [5] to "enhance strategic agility" [6], [4], [2]. Each of the aforementioned goals may target specific parts, i.e. area-of-interests, in the enterprise, hence calling for a goal-adequate conceptualization of the overall architecture. Moreover, it has repeatedly been discussed in literature e.g. by Buckl et al. in [7] or Aier et al. in [8] that having a "complete" conceptualization, i.e. one addressing all possible goals without asking whether the enterprise is interested in or not, leads to an overly complex model of the EA. This may especially be true, as the EA is considered to incorporate a broad variety of aspects from business to technical infrastructure but also encompasses manifold cross-cutting aspects, as *strategies*, *projects*, or *standards* (cf. Figure 1).

A "giant" model trying to cover the entire EA in an all-embracing manner typically demands for costly gathering and maintenance processes for information that is not needed to pursue the actual goals. On the contrary, a situation where an enterprise willing to manage its EA, has to develop a conceptual model for the EA "from scratch" also reflects an unsatisfactory situation. Instead of re-inventing the wheel, an enterprise should be able to call on best-practice solutions for addressing typical concerns and goals in EA modeling. This need is to some extent reflected by different EA approaches and frameworks, as The Open Group Architecture Framework (TOGAF) [10] or the pattern-based approach to EA management introduced by Buckl et al. in [7]. These approaches present fragments for conceptual models for the EA targeting different architectural concerns and goals. These fragments are, as the majority of conceptual models (cf. Aier et al. [11], Johnson and Ekstedt [3] or Buckl et al. [7]) for the EA, described in an object-oriented manner, using concepts as *class*, *property*, and *relationship* to conceptualize the EA. Nevertheless, over the years specific aspects in EA modeling, e.g. temporality (cf. Saat [12]), property-dependency (cf. Johnson [13] and Buckl et al. [14]) and project-dependency modeling (cf. Buckl et al. [15]), were analyzed more in-depth. The results of these analyses hint towards possible limitations of "purely" object-oriented conceptualizations of the EA. In line with these arguments, this paper analyzes the state-of-the-art in modeling cross-cutting aspects, as *goals* or *projects*, in current conceptual models of the EA. The corresponding research questions driving the paper read as follows:

1) *What are typical cross-cutting aspects in the EA that target manifold architectural concepts?*

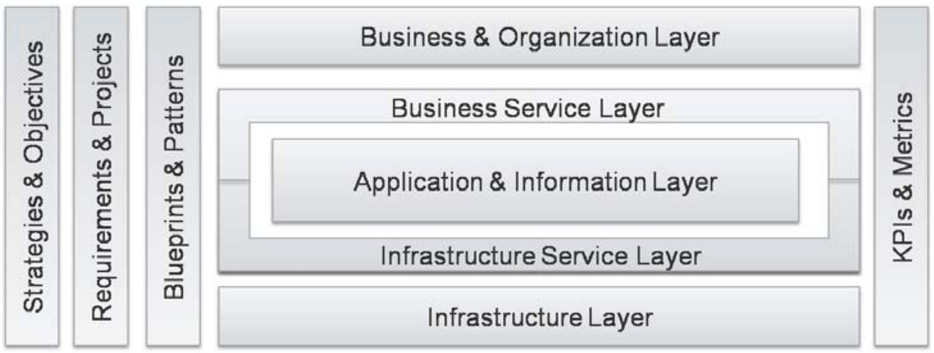

Fig. 1. Layers and cross-cutting aspects of an EA (according to Matthes et al. [9])

2) *How can these cross-cutting aspects be adequately conceptualized?*

These two research questions are addressed in the remainder of the article as follows. In Section II, we revisit selected approaches to EA modeling in respect to aspects that aim at manifold different concepts in the EA. The findings of this section are consolidated to four cross-cutting aspects of EA modeling in Section III. Along these aspects we propose a suitable alternative modeling language for EA conceptualizations. Concluding Section IV summarizes the findings of the paper and provides an outlook on directions for future research.

II. State of the art

Manifold EA management approaches propose models for the EA, more precisely introduce *information models*[1] outlining what an architecture description should consist of. These models differ in respect to depth and width of coverage of architectural concepts in general, and cross-cutting aspects in special. in the following paragraphs, we revisit a selection of prominent EA management approaches. We thereby focus on approaches providing an EA information model and put special emphasis on models incorporating cross-cutting aspects.

a) Zachman Framework: The perhaps most frequently quoted framework in the context of EA management is the Zachman framework, which dates back to the article **??** of Zachman on "information systems architecture". In its first version the framework applies a set of generic *abstractions* and *perspectives* to the enterprise as a whole. In terms of Zachman each abstraction describes a different characteristic of the enterprise, i.e. shapes a specific area-of-interest, such as "data", "network", or "people". Conversely each abstraction may be considered from different perspectives ranging from a "conceptual" one describing scope and objective of the corresponding elements over a "logical" system model perspective down to a code-oriented "out-of-context" perspective. Over this range of different perspectives the level of detail increases and certain architectural concepts enter or leave the center of attention. While this at first sight seems a prominent example of cross-cutting aspects of an enterprise's architecture, a look into the more detailed version of the framework as presented by Sowa and Zachman in [16] sheds a different light on this fact. The "detailed cell metamodel" contains information models separate for the different perspectives but integrated over the different characteristics. These information models are structurally equivalent over the perspectives, but undergo a "renaming" which might mirror an *ontological instantiation* linking them. For example the fact that a "business relationship involves business entities" on a higher level perspective is reflected by an isomorphic modeling of "data entity relationship involving datat entities" on a lower level. In this sense, the framework of Zachman covers the – ontologically interesting – aspect of repeated ontological instantiation, which may nevertheless not be considered as a "true" cross-cutting aspect applying on different architecture elements.

b) The Open Group Architecture Framework (TOGAF): A well-known framework for EA management is "The Open Group Architecture Framework" (TOGAF) [10], that in its most recent version 9 brings along a information model for architectural descriptions – the *content meta-model*. This information model does not explicitly account for cross-cutting aspects but describes several types and attributes that are cross-cutting of nature. Most evidently, this nature especially applies for all types that the framework labels with "associated with all objects", which are the types Principle, Constraint, Assumption, Requirement, Gap, and Work Package. Three of these types are intended to reflect *evolutional* and *intensional* aspects in a model, namely:

- Principles reflect guidelines how an element in the architecture should be evolved,
- Constraints describe "prohibited" regions for architecture development, and
- Requirements mirror needs that the architecture evolution should address.

Complementing especially the latter type, Gaps between the current and the target state are described and Work Packages for closing these gaps as part of a project are devised. Work Packages may be regarded actual cross-cutting types, representing parts of the work breakdown structure of a

1

project[2]. GAPS in contrast do not have such clear nature in TOGAF, existing somehow between architectural states and not clearly being parts of an architecture, but mere constructs and vehicles of architecture analysis. While the reification of differences between distinct architectural states as well as their description may be an interesting aspect of EA modeling, TOGAF does not provide additional information on the usage of GAPS in EA models, such that we cannot ultimately decide on their cross-cutting nature. The type ASSUMPTION is finally used to express uncertainty in respect to an architectural fact, i.e. uncertainty about the existence of an architectural concept or a relationship, respectively.

Delving deeper into the documentation of the content metamodel some more cross-cutting aspects can be uncovered. Different kinds of components, i.e. parts of the "physical" structure of the architecture, are equipped with attributes that reflect the *life-cycle* of corresponding instances. Speaking more precisely, these types possess attributes to model, when the corresponding instance was set "operational" or is "retired". Another cross-cutting aspect reverberates through manifold types in TOGAF's content metamodel, namely the aspect of *standardization*. Many logical and physical components of the architecture can be classified according their level of standardization, ranging from "Non-Standard" via "Standard" to "Retired Standard". In this sense, TOGAF mirrors the importance of standardization as a main goal commonly pursued in EA management (cf. e.g. Aier et al. in [17] or Buckl et al. in [18]) in the content meta-model.

c) Core Business Metamodel and University of St. Gallen approach: In [19], Österle et al. present the *Core Business Metamodel* as information model for modeling EAs in the context of business-driven EA management. This information model puts strong emphasis on structural and static aspects of the architecture and stays on the level of types and relationships, i.e. does not specify attributes. In this respect, cross-cutting aspects are hardly discussed or made explicit. A sole exception applies for goals which are introduced in the information model and are discussed as applicable on multiple different types. The core business metamodel exemplifies the role of goals with a relationship to business level concepts, but alternative applications are discussed alongside.

A different cross-cutting aspect is discussed by Saat in [12]. In the article he emphasizes on the importance of time for EA models in a twofold sense. Firstly, the EA is alluded to as evolving subject that changes over time, leading to the need that also the corresponding models adapt to the changed situation. Secondly, Saat details on the importance of lifecycle in the architecture, i.e. describes that certain architectural components, e.g. business applications or infrastructure components, go through different phases in their existence. Exemplifying this, a business application can be "in development", "in introduction", "operational", or "retired".

[2]TOGAF sees itself as a framework for "EA management projects", whose structure is modeled in corresponding work packages.

d) EA Analysis and KTH Stockholm approach: Johnson and Ekstedt present in [3] a number of models dedicated to analyses of EAs. These models are thereby described via *architecture theory diagrams*, which relate observable (manageable) properties of EA concepts to properties that operationalize EA-relevant goals. Revisiting the goal of *performance*, the corresponding theory diagram decomposes the abstract property to more concrete properties, e.g. *latency* and *throughput*. In a similar way, other goals, as *availability* or *security* are operationalized via measurable architecture properties. Along the discussion of how to analyze such properties, Johnson and Ekstedt further discuss on the architectural concepts on which these properties might apply. While business applications are often stated as valuable subjects of these goals, other concepts, e.g. business processes, may also be subjected to corresponding analyses. An exemplary case presented by Buckl et al. in [14] shows how architecture theory diagrams can be applied to analyze availability of EA concepts ranging from infrastructure components to business services. This can be regarded a clear indication for the cross-cutting nature of EA-relevant goals and their operationalizations into architectural properties. In [3] Johnson and Ekstedt further discuss relationships connecting different EA-relevant goals with each other and with corresponding architectural concepts. These relationships are exemplified in a information model relating goals to projects that are performed to pursue the goal. The project may be linked to "affected architectural concepts".

In [20] Johnson et al. outline a tool for performing the EA analysis, more precisely for computing actual values for properties that operationalize EA-relevant goals. The general mechanism of the tool is therein based on the conception of the Bayesian belief network. While the details of this mechanism are not of high interest for this paper, the usage of belief networks emphasizes on an important cross-cutting aspect discussed by Johnson et al., namely the aspect of *uncertainty*. The mechanism for dealing with uncertain information in the EA model presented in the paper goes far beyond the basic notion of uncertainty as discussed by TOGAF. Nevertheless, where TOGAF allows to explicitly mark some piece of information as uncertain, the mechanism of Johnson et al. [20] sees uncertainty as a general and inevitable fact in any EA model. Hence, no mechanism to distinguish between certain and uncertain information in an architecture model is provided.

e) EA management Pattern Catalog: Buckl et al. introduced in [7] a pattern-based approach to EA management. This approach has ever since been further refined and is nowadays presented as wiki incorporating a catalog of so called EA management patterns (see [21]). This catalog describes three distinct types of EA management patterns, namely *methodology patterns* (M-patterns), *viewpoint patterns* (V-patterns), and *information model patterns* (I-patterns), of which the latter are most interesting for understanding cross-cutting aspects in EA models. Each I-pattern presents an information model consisting of the types, relationships and properties needed to address a specific EA-related problem. Each I-pattern is thereby focused on the corresponding problem, such that the same types may occur more than once in the entire pattern catalog. Exploring the occurences of the model types, different cross-cutting aspects can be discovered:

- PROJECTS are contained in multiple I-patterns and affect different other types, as BUSINESSAPPLICATIONS or TECHNOLOGIES.
- Different architectural types are subject to lifecycle, as modeled via a list of properties indicating start- and end-date of corresponding lifecycle phases, e.g. for BUSINESSAPPLICATIONS or INFRASTRUCTURECOMPONENTS.
- STANDARDS are, explicitely as dedicated types and relationships or implicitly via corresponding standardization-related properties related with different architectural concepts, as BUSINESSAPPLICATIONS or INFRASTRUCTURESERVICES.
- STRATEGIES and GOALS are modeled via dedicated types that are in different I-patterns linked to different architectural concepts, e.g. BUSINESSAPPLICATIONS or BUSINESSSERVICES.

With each pattern describing a practice-proven solution that has been observed in at least three practice cases, the aforementioned list clearly hints to the importance of cross-cutting aspects in EA models.

Synopsis The above selection of EA management approaches and their corresponding EA information models provides more than a few indications on recurring cross-cutting aspects in EA modeling. Aspects as *project-dependencies*, *lifecycles*, *goal-dependencies*, and *standardization* are discussed more than once, and may clearly be candidates for a more detailed analysis.

III. MODELING CROSS-CUTTING ASPECTS

The analyses of the state-of-the-art in EA modeling revealed four different types of cross-cutting aspects that are included in current EA modeling approaches, namely *lifecycles*, *standardization*, *projects*, and *goals*. For modeling these aspects, the approaches bring along specific classes that may be linked to all "regular" classes representing EA concepts from the architecture layers (cf. Figure 1). To understand the specifics of the different types of classes involved in these models, we resort to ontological foundations of conceptual modeling as e.g. described by Guizzardi in [22]. In this work, the author analyzes the general notion of *type* and relates it to the concept *class* from object-oriented modeling. From there, Guizzardi derives a taxonomy of different "types of type" and introduces conceptions that we will rely on in our subsequent discussions. Guizzardi introduces a distinction between:

- *SortalUniversal* describing a type that supplies a *principle of identity*[3] and
- *MixinUniversal* describing a type that covers entities with different *principles of identity*, i.e. is *dispersive*.

[3]A *principle of identity* supports the judgement, if two entities of that type are the same or not.

a) Modeling projects and lifecycles: Based on this distinction, we revisit the typical modeling idioms used incorporate cross-cutting aspects in conceptual models for the EA. Figure 2 shows the idiom PROJECT-AFFECTS-ANYOBJECT, which can be found in different modeling approaches (cf. Section II), but also in different EA management tools (cf. Matthes et al. in [9]). The ANYOBJECT class used in the modeling idiom can be regarded to be a dispersive type, i.e. a type that does not supply a principle of identity. In line with this argument we added a stereotype proposed by Guizzardi in [22] to emphasize the mixin-nature of the corresponding class. Conversely, the class PROJECT can be regarded to be a sortal, i.e. a type supplying a principle of identity. Another stereotype proposed by Guizzardi is used to indicate this.

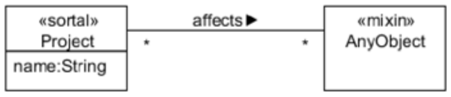

Fig. 2. Modeling building-block PROJECT-AFFECTS-ANYOBJECT

The modeling idiom presented in Figure 2 is a simplistic but suitable building-block for incorporating project-effect modeling into a conceptual model for EA management. Nevertheless, later discussions undertaken by Buckl et al. in [23] call for a more sophisticated building-block allowing to distinguish different types of project-effects, namely *introduces*, *changes*, and *retires*. Figure 3 shows the corresponding conceptual model as an extension of the initial project-effect modeling idiom presented above. This extended building-block (PROJECT-INTRODUCES-CHANGES-RETIRES-ANYOBJECT) employs the concept of the *relationship inheritance* as provided by the Unified Modeling Language (UML) [24].

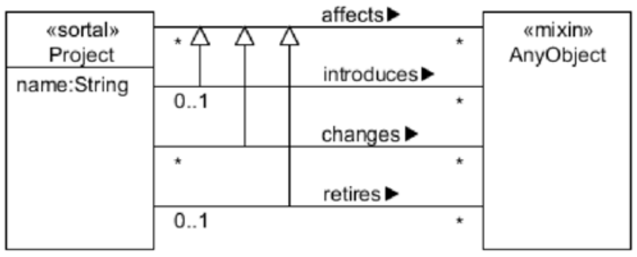

Fig. 3. Modeling building-block PROJECT-INTRODUCES-CHANGES-RETIRES-ANYOBJECT

Albeit the UML provides appropriate modeling elements and notations to describe the extended building-block, the question how the relationships INTRODUCES, CHANGES, and RETIRES relate to the relationship AFFECTS deserves additional attention. The ontological framework of Guizzardi (cf. [22]) provides a terminology to analyze this relation[4]

[4]In this section, we use the term *relation* to denote the "inheritance" connecting INTRODUCES and AFFECTS, CHANGES and AFFECTS, and RETIRES and AFFECTS, respectively to avoid ambiguous statements. In line with this, we slightly deviate from the original terminology used by Guizzardi in [22].

in more detail. At first, Guizzardi points to a critical distinction between different types of relationships, namely *formal relationship* and *material relationships*. The relationships from conceptual model PROJECT-INTRODUCES-CHANGES-RETIRES-ANYOBJECT are all relationships of the later type, i.e. relationships which "alter the history of the involved *relata*" (cf. Bunge [25]). In contrast, formal relationships exist as tuples interlinking certain entities without changing the entities' very nature. The relationships found in the conceptual models for EAs in this section all are material relationships and hence may be reified by a corresponding *relator* type. In the considered example these may be the types EFFECT, INTRODUCTION, CHANGE, and RETIREMENT, respectively. Based on these reifying types, we can express the relation between the different relationships more precisely, leading to a type hierarchy as shown in Figure 4. At this point it should be noted that the different relators commit to different, but compatible, multiplicities for the concepts in relationship.

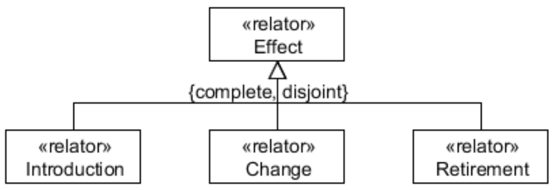

Fig. 4. Relations between EFFECT, INTRODUCTION, CHANGE and RETIREMENT

Before delving deeper into the intricacies of project modeling, we shall make a digression on the cross-cutting aspect of the *lifecycle*. Different classes modeled in EA meta models as the ones discussed in Section II are not only of "punctiform" interest, i.e. EA models cover the evolution of corresponding instances over time. To incorporate the evolving nature of some EA concepts, as business applications or infrastructure components, we call on *non-rigid* typing, which means that an instance can change its type over time. Guizzardi discusses in [22] several variants of non-rigid types, of which the *phased sortal* is most appropriate to cover lifecycle aspects. Phased sortals describe the different phases that an instance of an according type goes through in a way that only one phase applies at a time. Exemplifying this along the business application concept as discussed by Saat in [12], we provide a corresponding modeling building-block LIFECYCLED BUSINESSAPPLICATION as shown in Figure 5.

In an actual EA model each instance of a lifecycled type may retain information on the specific point in time, when a transition between two phases took place. Additionally, the phases may be subject to a sort of ordering or more general a set of constraints of the valid transitions between different phases. While a mechanisms to describe these constraints may be an interesting subject to explore, we end our digression on lifecycled types here and return to the cross-cutting aspect of project modeling. More precisely, we revisit the project

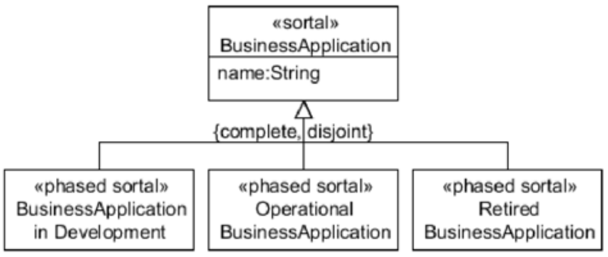

Fig. 5. Modeling building-block LIFECYCLED applied on BusinessAppplication

modeling technique presented by Buckl et al. in [26] from the perspective of the therein contained primitives of modeling.

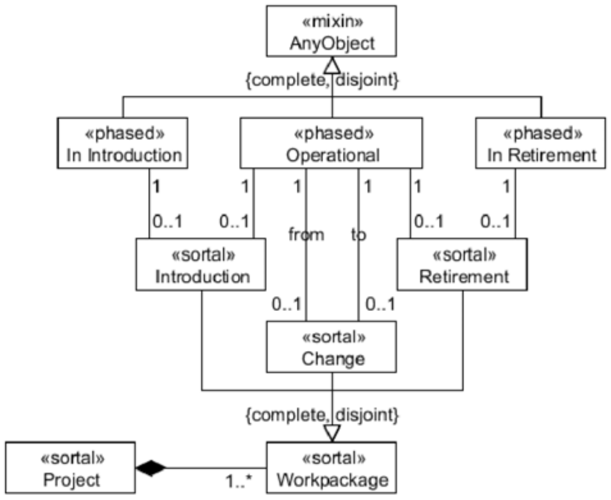

Fig. 6. Modeling building-block PROJECT-LIFECYCLE-ANYOBJECT

Decomposing a project into the single WORK PACKAGES that transform architecture components, the modeling building-blocks for modeling lifecycles and modeling projects can be composed into a comprehensive building-block PROJECT-LIFECYCLE-ANYOBJECT (cf. Figure 6). This block accounts for the fact that a concept may be in introduction, operational, or retired. Different types of work packages mediate the transitions between the different states that an EA component may commit to. More precisely, while INTRODUCTION sets an EA component operational and RETIREMENT retires an operational EA component, CHANGE is used to describe that one operational EA component is evolved into another operational component. This may be exemplified with a business application that is updated to a newer version.

b) Modeling standards: Having explored different options to model projects and their influences on EA concepts on different levels of the architecture (cf. Figure 1), we shortly explore different ways how standards are incorporated in an information model for EA modeling. Once again, the mixin concept may be used to concisely model the true nature of a "standard". A fairly simplistic model, as found in TOGAF's content metamodel and some patterns of the EA management pattern catalog, introduces a single property indicating whether an instance of the given type conforms to a standard or not. Rewritten in terms of a corresponding mixin this model building-block (STANDARDIZABLE) presents itself as shown in Figure 7.

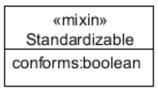

Fig. 7. Modeling building-block STANDARDIZABLE

More sophisticated models for describing standardization in the EA call on the reification of standards to EA entities, i.e. introduce types that mirror selected standards and indicate the standard conformity of an instance via an according relationship. A corresponding model building-block (STANDARD-STANDARDIZABLE) is presented in Figure 8.

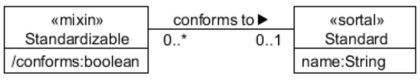

Fig. 8. Modeling building-block STANDARD-STANDARDIZABLE

Via the *derived* property as contained in the building-block STANDARD-STANDARDIZABLE we further show a mechanism that can be used to consistently exchange these building blocks. This especially means that an EA type being standardizable according to the simple model may be augmented to a type being standardizable via a relationships to a standard without loosing factual expressiveness. In this sense the derived property can be regarded as a vehicle of "backwards-compatability" of EA information models aligning to the corresponding notion of sub- and super-concerns in the context of EA modeling as explored by Buckl et al. in [27].

c) Modeling goals: In the different approaches to EA modeling mainly two different ways to model goals exist. These ways are explored in detail by Buckl et al. in [28], whereas we resort to giving a brief summary of their main findings here. The first technique for modeling goals is described as GOAL-AFFECTS-ANYOBJECT and contains a similar model as the PROJECT-AFFECTS-ANYOBJECT building-block shown in Figure 2. In analogy to this building such way of modeling may be used to denote which architectural elements are affected by which specific goals. Nevertheless, relevant parts of the "true" nature of a goal are neglected by such model, most notably the notion of "measuring the achievement of a goal". The second technique, METRICS-TO-INDICATORS, identified by Buckl et al. in [28] focuses on this aspect of goal modeling, more precisely of operationalizing goals to measurable metrics. These metrics are further aggregated to indicators that conversely characterize an architecture element in respect to the operationalized goal. By doing so, the technique augments an EA model for providing decision support in respect to a selected goal, but fails to establish an "in-model" relationship between that goal and its corresponding indicators.

The analyses presented by Buckl et al. in [28] give an indication of the special nature of goals compared to other cross-cutting aspects of EA modeling. This may best be exemplified by revisiting the aforementioned techniques from a conceptual perspective. While in GOAL-AFFECTS-ANYOBJECT a goal is modeled as instances in an EA model, the indicators used in METRICS-TO-INDICATORS are modeled as properties of types in an EA information model. Pur more colloquially, one would say that a goal is modeled "from above" in the first technique, while being modeled "from below" in the second one. Dealing especially with the latter perspective on goals, Buckl et al. propose in [28] the concept of the QUESTION which in line with the *Goal-Question-Metric*-approach of Basili et al. (cf. [29]) aggregates the metrics and indicators to a (dispersive) type, i.e. a mixin. From this perspective a QUESTION, helping to operationalize an actual goal (instance), can be incorporated as a cross-cutting aspect similar to the STANDARD (cf. Figure 7). Further, this points towards a solution to the problem of the twofold nature of a goal being a concept located on two ontologically different levels with respect to instantiation. Using a multiple ontological instantiations, this solution can be described as shown in Figure 9, where the types GOAL and QUESTION are defined on the same ontological level. From there, they are instantiated to an actual goal and a concrete question, of which the latter in turn acts as mixin on the ontological level below. On this level the sortal universals from reflecting non-cross-cutting architecture constituents reside and the question-mixin can be added as aspect to one of these. The hence augmented architectural concept is further instantiated to an actual architecture element describing a distinct and identifiable "real-world" element, while retaining the relationship to the according goal on an ontologically higher level.

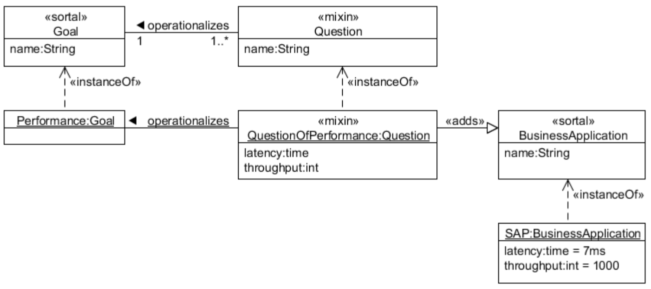

Fig. 9. Modeling building-block GOAL-QUESTION-METRIC applied on BusinessAppplication

Summary The discussions in the preceding paragraphs showed how selected cross-cutting aspects can be incorporated in modeling building-blocks facilitating the construction of an organization-specific EA information model. A brief example utilizing building-blocks to construct a specific information model was also given in Figure 5. The different considerations and discussions showed that for concisely modeling these building-blocks the modeling primitives provided by the UML [24] did not seem fully sufficient. By introducing more ontologically richer modeling primitives, as mixins and phases, we could achieve more concise models and building-blocks. Stated as response to the article's second research question, we derive that cross-cutting aspects can be incorporated into EA information models by using richer types, namely non-rigid types (phases) and types not committing to a principle of identity (mixins). Such "rich" types may in line with Guizzardi [22] be considered natural with respect to the way how humans characterize and conceptualize a universe of discourse. In this sense they should not be regarded as "artificial" extensions to a modeling language as the UML, but as a more detailed perspective on what many object-oriented languages subsume with the term "class". Reflecting the findings on goal modeling, further modeling mechanisms are needed, namely ones supporting more than one level of (ontological) instantiation. With such *multi-level* modeling being a requirement that applies to EA information modeling in general (cf. Buckl et al. in [30]), mechanisms as the UML *powertypes* or the *clabjects* proposed by Atkinson and Kühne in [31] may deserve a more in-depth evaluation. As the article presented here has a focus on *conceptualizing* cross-cutting aspects, we abstain from analyzing different modeling facilities and languages that may or may not be suited for representing such conceptualizations.

IV. Outlook

In this paper we revisited prominent EA information models in respect to their coverage of cross-cutting aspects. Thereby, we could show that aspects of project-, goal-, standards-, and lifecycle-modeling occur in different models and modeling building-blocks. Based on these findings, we discussed selected aspects in more detail and challenged their recurring modeling based on the UML [24]. Resorting to a richer language model grounded in the work of Guizzardi (cf. [22]), we could show how the selected cross-cutting aspects can be modeled conceptually in a more concise manner. By doing so, we laid the basis for modeling building-blocks that can be reused in different EA modeling approaches and corresponding information models, when cross-cutting aspects should be incorporated.

The findings of this paper are up to this point of theoretic nature. This on the one hand ascribes to the focus of this paper on *conceptualizing* cross-cutting aspects instead of proposing "full-blown" EA information model building-blocks, which were committed to a specific modeling facility and language. Nevertheless, for applying the results of this paper, the conceptualizations have to be mapped to a modeling language that adequately supports the richer type system of *sortal universals* vs. *mixin universals* as well as supports both *rigid* and *non-rigid* typing. Further, such language had to commit to multi-level modeling, i.e. should not be confined to two ontological levels of metaization connected via a single ontological instantiation. Selecting an appropriate language for this purpose has nevertheless to account for other requirements that apply to a meta-language for creating EA information models. Such requirements are for example put forward by Buckl et al. in [30].

Concretized using a specific meta-language, it further remains to be proven that the building-blocks targeting cross-cutting aspects can be utilized to build organization-specific EA information models. While we do not expect the building-blocks to be completely self-explanatory, information modeling experiments with enterprise architects may be conducted to show whether the building-blocks are "usable" for EA information model creation or not. Usability can in this context be concretized in different ways, e.g. as decreasing the number of modeling errors, fastening the creation of information models, or making the consequences of a certain modeling more explicit. All these different aspects of usability are nevertheless inevitably connected to the corresponding meta-language such that the selection of the language deserves critical attention in order to not deprive the approach of potential benefits. Linked to this, it would further be important to find adequate tool support for the creation of EA information models, not only in respect to the meta-language but also to the utilization of building-blocks.

Linking back to EA management tools as currently used to create, maintain and analyze models of EAs, another aspect of interest enters the center of attention. While only briefly discussed in literature, many of the tools provide role-based access control mechanisms on different levels of detail with respect to the EA information model or its instantiation. Matthes et al., who in [9] present practitioners' requirements for EA management tools, delineate that access control mechanisms are used to technically realize responsibilities and approve information interests of people concerned with EA management. In this sense different types of relationships between EA concepts and "EA management people" may actually exist in the EA management processes and functions of different organizations. Mapping them from the level of the activities to the level of the EA information models may lead to another interesting cross-cutting aspect that deserves a more in-depth research in the future.

References

[1] M. Lankhorst, *Enterprise Architecture at Work: Modelling, Communication and Analysis.* Berlin, Heidelberg, Germany: Springer, 2005.

[2] J. Schekkerman, *Enterprise Architecture Good Practices Guide – How to Manage the Enterprise Architecture Practice*. Victoria, BC, Canada: Trafford Publishing, 2008.

[3] P. Johnson and M. Ekstedt, *Enterprise Architecture – Models and Analyses for Information Systems Decision Making*. Pozkal, Poland: Studentlitteratur, 2007.

[4] K. D. Niemann, *From Enterprise Architecture to IT Governance – Elements of Effective IT Management.* Wiesbaden, Germany: Vieweg+Teubner, 2006.

[5] B. van der Raadt and H. van Vliet, "Designing the enterprise architecture function," in *4th International Conference on the Quality of Software Architectures (QoSA2008)*, Karlsruhe, Germany, 2008, pp. 103–118.

[6] G. B. Bird, "The business benefit of standards," *StandardView*, vol. 6, pp. 76–80, 1998.

[7] S. Buckl, A. M. Ernst, J. Lankes, K. Schneider, and C. M. Schweda, "A pattern based approach for constructing enterprise architecture management information models," in *Wirtschaftsinformatik 2007*. Karlsruhe, Germany: Universitätsverlag Karlsruhe, 2007, pp. 145–162.

[8] S. Aier, S. Kurpjuweit, C. Riege, and J. Saat, "Stakeholderorientierte dokumentation und analyse der unternehmensarchitektur," in *GI Jahrestagung (2)*, ser. LNI, H.-G. Hegering, A. Lehmann, H. J. Ohlbach, and C. Scheideler, Eds., vol. 134. Bonn, Germany: Gesellschaft für Informatik, 2008, pp. 559–565.

[9] F. Matthes, S. Buckl, J. Leitel, and C. M. Schweda, *Enterprise Architecture Management Tool Survey 2008*. Munich, Germany: Chair for Informatics 19 (sebis), Technische Universität München, 2008.

[10] The Open Group, "TOGAF "Enterprise Edition" Version 9," http://www.togaf.org (cited 2010-02-25), San Diego, USA, 2009.

[11] S. Aier, S. Kurpjuweit, O. Schmitz, J. Schulz, A. Thomas, and R. Winter, "An engineering approach to enterprise architecture design and its application at a financial service provider," in *Modellierung betrieblicher Informationssysteme (MobIS 2008) – Modellierung zwischen SOA und Compliance Management 27.-28. November 2008 Saarbrücken*, 2008, pp. 115–130.

[12] J. Saat, "Zeitbezogene abhängigkeitsanalysen der unternehmensarchitektur," in *Multikonferenz Wirtschaftsinformatik (MKWI) 2010*, M. Schumann, L. M. Kolbe, M. H. Breitner, and A. Frerichs, Eds., 2010, pp. 29–30.

[13] P. Johnson, L. Nordström, and R. Lagerström, "Formalizing analysis of enterprise architecture," in *Enterprise Interoperability*. London, UK: Springer, 2007, pp. 35–44.

[14] S. Buckl, U. Franke, O. Holschke, F. Matthes, C. M. Schweda, T. Sommestad, and J. Ullberg, "A pattern-based approach to quantitative enterprise architecture analysis," in *15th Americas Conference on Information Systems (AMCIS)*, San Francisco, CA, USA, 2009.

[15] S. Buckl, A. Ernst, F. Matthes, and C. M. Schweda, "An information model for landscape management – discussing temporality aspects," in *Pre-Proceedings of the 3rd Workshop on Trends in Enterprise Architecture Research*, P. Johnson, J. Schelp, and S. Aier, Eds., Sydney, Australia, 2008, pp. 63–77.

[16] J. F. Sowa and J. A. Zachman, "Extending and formalizing the framework for information systems architecture," *IBM Systems Journal*, vol. 31, no. 3, pp. 590–616, 1992.

[17] S. Aier, C. Riege, and R. Winter, "Unternehmensarchitektur – literaturüberblick stand der praxis," *Wirtschaftsinformatik*, vol. 50, no. 4, pp. 292–304, 2008.

[18] S. Buckl, A. M. Ernst, J. Lankes, F. Matthes, and C. M. Schweda, "State of the art in enterprise architecture management 2009," Chair for Informatics 19 (sebis), Technische Universität München, Munich, Germany, Tech. Rep., 2009.

[19] H. Österle, R. Winter, F. Hoening, S. Kurpjuweit, and P. Osl, "Der St. Galler Ansatz des Business Engineering: Das Core Business Metamodel," *Wisu – Das Wirtschaftsstudium*, vol. 2, no. 36, pp. 191–194, 2007.

[20] P. Johnson, E. Johansson, T. Sommestad, and J. Ullberg, "A tool for enterprise architecture analysis," in *11th IEEE International Enterprise Distributed Object Computing Conference (EDOC 2007), 15-19 October 2007, Annapolis, Maryland, USA*. Annapolis, Maryland, USA: IEEE Computer Society, 2007, pp. 142–156.

[21] Chair for Informatics 19 (sebis), Technische Universität München, "Eam pattern catalog wiki," http://eampc-wiki.systemcartography.info (cited 2010-02-25), 2010.

[22] G. Guizzardi, "Ontological foundations for structural conceptual models," Ph.D. dissertation, CTIT, Centre for Telematics and Information Technology, Enschede, The Netherlands, 2005.

[23] S. Buckl, A. M. Ernst, J. Lankes, and F. Matthes, "Enterprise Architecture Management Pattern Catalog (Version 1.0, February 2008)," Chair for Informatics 19 (sebis), Technische Universität München, Munich, Germany, Tech. Rep., 2008. [Online]. Available: \url{http://eampc-wiki.systemcartography.info/}

[24] Object Management Group (OMG), "Uml 2.2 superstructure specification (formal/2009-02-02)," http://www.uml.org (cited 2010-02-25), 2009.

[25] M. Bunge, *Treatise on Basic Philosophy – Ontology I: The Furniture of the World*. New York: Reidel Publishing, 1977.

[26] S. Buckl, A. M. Ernst, F. Matthes, and C. Schweda, "An information model for managed application landscape evolution," *Journal of Enterprise Architecture (JEA)*, vol. 5, no. 1, pp. 12–26, 2009.

[27] S. Buckl, F. Matthes, and C. M. Schweda, "Interrelating concerns in ea documentation – towards a conceptual framework of relationships," in *2nd European Workshop on Patterns for Enterprise Architecture Management (PEAM2010)*, Paderborn, Germany, 2010.

[28] ——, "A technique for annotating ea information models," in *6th international workshop on Enterprise & Organizational Modeling and Simulation 2010*, ser. Lecture Notes in Business Information Systems, J. Barjis, Ed. Springer, 2010.

[29] V. R. Basili, G. Caldiera, and H. D. Rombach, *The Goal Questin Metric Approach*. New York: Wiley, 1994.

[30] S. Buckl, F. Matthes, and C. M. Schweda, "A meta-language for ea information modeling – state-of-the-art and requirements elicitation," in *Enterprise, Business-Process and Information Systems Modeling*, ser. Lecture Notes in Business Information Systems, J. K. S. N. E. P. R. S. R. U. Ilia Bider, Terry Halpin, Ed. Springer, 2010, pp. 169–181.

[31] C. Atkinson and T. Kühne, "Reducing accidental complexity in domain models," *Software and Systems Modeling*, pp. 345–359, 2007.