Empowering Business Users to Analyze Enterprise Architectures: Structural Model Matching to Configure Visualizations

Sascha Roth, Matheus Hauder, Marin Zec, Alexej Utz, and Florian Matthes
Technische Universität München
Boltzmannstr. 3
85748 Garching, Germany
{*roth, matheus.hauder, marin.zec, alexej.utz, matthes*}*@tum.de*

***Abstract*—Visualizations are common means to analyze Enterprise Architecture (EA) models and support decision makers with relevant information on organizational processes, information systems, infrastructure, and their interconnections. These EA visualizations are tailored typically to the specific information demand of stakeholders. Currently, creating such stakeholder-specific visualizations requires experts with a strong technical background due to complex configurations and inflexible tool solutions. In particular business users often lack technical expertise. At the same time, their concerns and questions often arise spontaneously. In this vein, visualizations that can be generated without expert knowledge would enable business users to perform ad-hoc analyses of an EA information model. Against this background, we propose a solution which facilitates analyses of arbitrary EA information models by non-technical stakeholders. Our approach is based on ad-hoc, configurable visualizations. We introduce an end-user friendly wizard that lowers the barrier for the creation of EA visualizations. The wizard is based on structural pattern matching of models.**

Key to our approach are abstract viewpoints that model best-practice knowledge of EA visualizations and abstract view models which model the information demand of abstract viewpoints. Our contributions in this paper are 1) a meta-information model capable to capture both the technical information demand of an abstract viewpoint and the information offer of an EA information model, 2) a pattern matching algorithm calculating viable configurations for bindings of visualizations to information, and 3) a wizard to support non-technical stakeholders with the creation of these visualizations. We present an implementation of our approach and show user interface design of the wizard. The wizard recommends feasible configurations automatically in order to unburden the configuration process for non-technical stakeholders.

***Keywords*-Enterprise Architecture, visual analysis, structural model matching, EA visualizations**

I. Introduction

Over the past decades, business applications and respective business application landscapes of organizations have grown to systems of systems intrinsically complex and hard to manage as a whole [1], [2]. At the same time, frequently changing market conditions force enterprises to transform their business to outperform competitors. In this vein, the mutual alignment of business and IT enables faster business transformations and ultimately contributes to increase shareholder return. Due to the intrinsic complexity of both business and IT, realizing effective and efficient IT support of constantly changing business processes is a major challenge of enterprises today. As a result, Enterprise Architecture (EA) management seeks to manage the arising complexity in order to increase business and IT alignment while reducing operational costs. Well executed EA management is commonly perceived as a strategic advantage since enterprises can respond faster and more reasonably to continuously changing market conditions.

Information about the EA may serve multiple stakeholders with different concerns and potentially unclear goals [3]. Common means for the analysis of this information are visualizations of these EA information models [4], [5], [6]. Due to the diverse nature of these concerns, decision makers require individual viewpoints that are tailored to their specific demand [7], [8]. Large enterprises commonly contain a large number of information systems, organizational processes, and infrastructure systems that are interconnected via interfaces. The information volume leads to large visualizations that are typically created with model-driven approaches by expert users [9], [10], [11]. Considering the background of EA visualizations[1] the EA management community employs a finite number of views with variations identified as patterns (cf. [14]), such that they can be predefined in an abstract manner including some variability points [12], [11]. This means, the EA information model is unknown to the visual designer [11] and bound at runtime via model mapping techniques [10].

EA endeavors commonly start by documenting the current state of the organization in an EA repository (e.g. [15], [16], [17], [18]). This status quo of the EA is typically documented with respect to a formalized information model (cf. [19]) capturing organizational processes, information systems, infrastructure, and their interrelations. The resulting EA information model consists of highly interlinked entities. Although there are proposals for a standard EA information model [20], as of today, enterprises tend to develop their

[1]We refer the interested reader to [12], [13], [3], for a predefined set of best-practice EA visualizations gathered from industry.

own organization-specific EA information model [21] to capture changing requirements and information demands from stakeholders. Thereto, EA management initiatives use highly specialized EA tools to manage EA information. With respect to the observations made above, some of these EA tools, in the following referred to as EA repositories, enable an adaptation of their underlying EA information model [22]. At the same time, EA information models may change frequently in case the data is automatically gathered from federated information sources (cf. e.g. [23], [24], [25], [26]).

Prevailing approaches for generating EA visualizations typically take into account multiple steps performed by a variety of different actors as shown in Figure 1. During the modeling step the required information is gathered from stakeholders or existing information sources in the organization (cf. e.g. [23], [24], [25], [26]). The result of the data binding step is a mapping between information model elements and an abstract viewpoint definition. Typically, this step requires specific knowledge from expert users. The definition of the abstract viewpoint is performed within a visualization design step. During configuration, the abstract viewpoint is detailed with e.g. color encodings, axis descriptions, and symbols. As a result, these subsequent steps lead to a communication effort between the various actors.

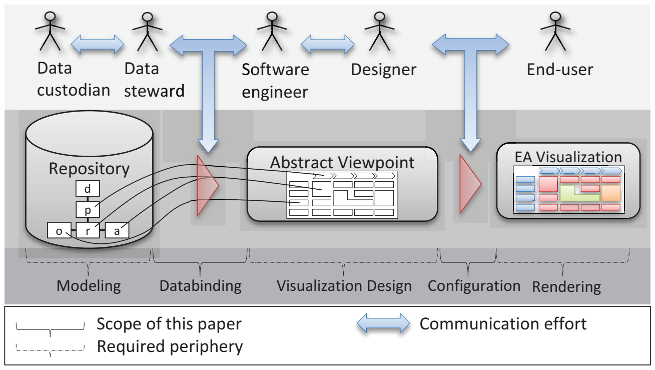


Figure 1. Overview of the current state and the scope of this paper

We aim at supporting end-users to create visualizations on their own to reduce this effort and allow the creation of ad-hoc visualizations. With regard to the considerations made above, we conclude to the following research question:

How can business users be empowered to visually analyze data of arbitrary EA information models?

Our contributions in this paper are 1) a meta-information model capable to capture both the technical *information demand* (abstract view model) of an abstract viewpoint (cf.[11]) and the *information offer* of an EA information model, 2) an algorithm for structural pattern matching of models calculating viable configurations for bindings of visualizations to information, and 3) a wizard to support non-technical stakeholders with the ad-hoc creation of EA visualizations.

Answering above research question, the remainder of this paper is structured as follows. First, we briefly revisit related work relevant for the scope of this paper in Section II. Then, we present our approach in Section III followed by a description of its implementation in Section IV. The paper concludes with an outlook on further research in Section V.

II. Related Work

Decoupling of visualizations and (model) data is discussed by research as well as industry (e.g. [27]). In particular in the field of software visualization, Panas et al. [28] describe an approach and a corresponding proof-of-concept for configurable software visualization. Instead of hand-coding mappings between model and visualization, they introduce a general and flexible framework for online-configuration of views. Their approach is based on *graph structures* of both the data model (i.e. software entities) as well as view model. These graph structures consist of nodes modeling entities or n-ary relations, edges modeling binary relations, and arbitrary properties of nodes and edges. Panas et al. provide a predefined set of mapping functions to map elements from the data graph to elements of the view graph. A visualization is configured by assigning a data graph property and the desired binding function to each view property. Then, the view graph is mapped to a scene graph which is specific to the rendering engine and serves as actual basis for the visualization. While Panas et al. distinguish between model, view, and scene and thus allow reuse of mappings, they do not offer suggestions for the end-user who might not have the required technical skills to specify a mapping from information offer to information demand.

Kruse et al. [10] discuss the motivation and advantages of decoupling models and visualizations in the EA domain. They acknowledge that the specification of a model transformation of an information model to a view model is typically a difficult task for business users. Kruse et al. suggest to use *higher order transformations* in order to avoid the need of specifying a new transformation for each viewpoint. They provide a prototypical implementation of their approach using the Eclipse Modeling Framework[2]. However, Kruse et al. recognize that the visualization configuration process is still "knowledge-intensive and error prone" and call for an intuitive interface.

In their works, Goldschmitt et al. [29] coin the term view-based modeling, i.e. using a domain-specific modeling language to manipulate underlying information. The abstract concept of manipulating model information by stakeholder-specific views is similar to one of the solutions presented in this paper. This research group concentrates on text-based approaches to synchronize abstract and concrete syntax. However, they do neither apply their techniques to best-practice EA visualizations nor empower business users to configure these.

[2]http://www.eclipse.org/modeling/emf/, last accessed: April 22, 2013

Visual notations, e.g. approaches like the Graphical Modeling Project[3], are based on concrete models and thus can only be used for concrete EA models. Since there is no common standard for EA models [30], organizations tend to develop an organization-specific model that describes their EA with respect to the organization's terminology and concepts.

The visualization approaches mentioned above do not address two major issues. First, ad-hoc information needs can not be met spontaneously by business users because knowledge-intensive modeling is required for view configuration in one way or another. Second, enterprise models tend to be dynamic [31], [32]. Visualizations should be flexible and adjustable to changes in the information model without demanding from the business user to have programming or sophisticated modeling skills.

In [11], we present both requirements and a conceptual framework for interactive EA visualizations. In this vein, we show how to define domain-independent visualization types and distinguish between information offer and information demand of a visualization type. The former is captured in the information model, the latter is captured in a so-called *abstract view model*. The information demand needs to be met by an associated information offer in order to be able to generate a particular visualization in a model driven manner. While the information model and the visualization are decoupled in our approach, we did not detail how the information model is mapped to the abstract view model. However, we did not discuss how business users can perform the rather abstract task of mapping the information model to the abstract view model without prior knowledge of the information model's details, modeling expertise, and/or programming skills. Especially in [22], [33] Matthes et al. analyze the state-of-the-art in EA management tools and conclude that current EA management tools require experts with programming skills to adapt visualizations.

Schweda [34] presents formal theories describing when two information models embed into each other. Thereby, Schweda distinguishes between strong subsumption (read/write) of models and weak subsumption of models (read-only). Based on his theories, we described the underlying idea of the solution presented in this paper in [35]. However, we neither presented a concrete implementation of the pattern matching algorithm nor did we provide an user interface design enabling business users to configure rather complex model mappings. The scope of this work is to show a prototypical implementation for visualization-driven ad-hoc analysis in the EA domain. We chose a structural over a semantic model matching since the latter requires semantic annotations and/or rule definitions. Those approaches often demand technical knowledge which we intend to avoid in the first place.

[3]http://www.eclipse.org/modeling/gmp/, last accessed: April 22, 2013.

III. Empowering Business Users to Configure EA Visualizations

As we detailed in [11], model-driven approaches to generate EA visualizations, e.g. [10], require a mapping of an EA information model to a particular abstract viewpoint, i.e. a 'visualization type'. Since the information model in particular may evolve over time [32], questions and concerns of stakeholders cannot be entirely anticipated. This also holds true for visualizations addressing them. At the same time, EA stakeholders often lack technical expertise and cannot be considered model experts nor experienced programmers.

As a reaction, we illustrate how business users (end-users) can be empowered to provide a model mapping of an EA information model to an abstract view model. As above-mentioned such a mapping is a basic prerequisite for generating an EA visualization in a model driven manner. First, we introduce a meta-information model used as a common denominator for both models. Subsequently, we introduce an algorithm that operates on two models complying to this meta-information model in order to compare two models structurally.

A. A Meta-information model for matching model structures

In line with model driven approaches to generate visualizations [10], [36], [11], we distinguish between technical information demand of a visualization, the so-called abstract view model, and the information demand given by an EA information model. Figure 2 illustrates a simplified meta-information model (m3) that is used to abstract from the concrete implementation of both the EA information model and the abstract view model, i.e. information offer and demand (models).

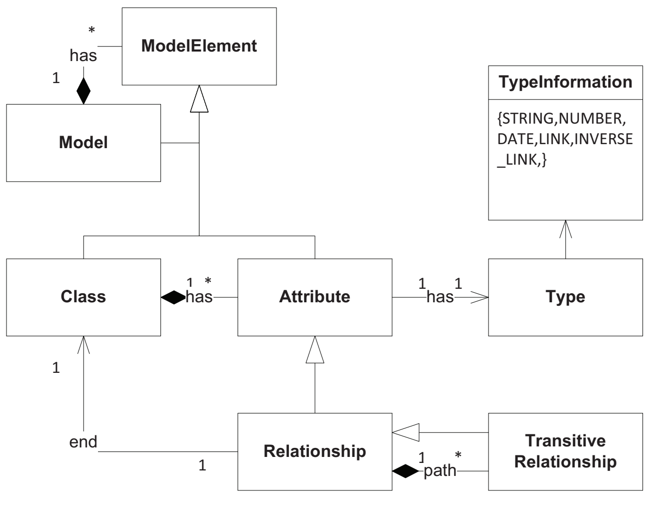


Figure 2. Meta-information model

The meta-information model covers basic concepts of the meta object facility (MOF)[4], i.e. Classes, Attributes, and Re-

[4]http://www.omg.org/mof/, last-accessed April 20, 2013.

lationships. For the latter, we capture transitive relationships explicitly and a path that gives information how to traverse the model to navigate through this transitive relationship. Eventually, this information is used in the later model-driven generation of an EA visualization. UML distinguishes between navigation forward, backward, or both. In our model, we consider relationships to be bidirectionally traversable. Against this background, our approach focuses on read-only EA visualizations (cf. also Schweda [34]). For our considerations, start and end of a relationship are swapped if the underlying information model uses directed relationships that are technically traversable in both directions.

B. An algorithm to match model structures

After abstracting the EA information model to an information offer model m_o and the abstract view model of an abstract viewpoint to an information demand model m_d (cf. [11]) to above described meta-information model, we use these models to facilitate the configuration of the visualizations.

Algorithm 1 matches both models, i.e. retrieves $matches \subseteq m_o$ that structurally fit m_d. The result of this algorithm may serve end-users in a twofold manner. On the one hand, structural matches of both models deliver valid configuration variants to the end-user. In this vein, a valid configuration can be utilized to bind m_o to m_d. Subsequently, this binding can be used to generate a visualization based on data conforming to m_o. Moreover, the distance of potential candidates is calculated which may serve as a suggestion for model extensions in a directed way. This means, business users get informed how to extend their model in such a way that extending a particular concept results in a valid configuration for an abstract viewpoint, i.e. the concept can be visualized.

The invocation of the algorithm starts with the function $compareModel(m_d, m_o)$ (line 1). This function receives the two models mentioned above - demand m_d and offer m_o - as parameters. In a first step (line 2), we retrieve all relationships existing in m_o including transitive relationships. In a second step (line 3-6) we enrich the classes of m_o by adding respective transitive relationships retrieved before. This step is required since we assume m_o is an abstraction of an EA information model with explicitly described direct relationships only.

To compare the models m_o and m_d, we compare the classes contained in these models. This is done in the last step (line 9-16) of the function. Therefore we calculate the number of non-matching attributes in every pair of classes. This metric is returned by $compareClass(c_d, c_o, elementsToIgnore)$ (line 12). In case of a match, we increase the matchCounter (line 14) and add the compared classes to the result set *matches* (Line 15). In addition, *compareModel* returns (line 17) the number of classes in m_d that did not match classes in m_o.

Algorithm 1 Pattern matching

```
1  — compareModel ($m_d$, $m_o$)
2  $rels \leftarrow getRelationshipEnds(m_o)$
3  foreach $rel_t \in rels.transitive$ do
4      foreach Class $c_o \in m_o$ do
5          if $c_o.classifier \equiv rel_t.start.classifier \wedge rel_t \notin c_o.attributes$
           then
6              $c_o.attributes \leftarrow c_o.attributes \cup rel_t$
7  $r \leftarrow 0$
8  $matchCount \leftarrow 0$
9  foreach $c_o \in m_o$ do
10     foreach $c_d \in m_d$ do
11         $elementsToIgnore \leftarrow \varnothing$
12         $r \leftarrow compareClass(c_d, c_o, elementsToIgnore)$
13         if $r \equiv 0$ then
14             $matchCount \leftarrow matchCount + 1$
15             $matches \leftarrow matches \cup (c_d, c_o)$
16             break
17 return $m_d.classes.count - matchCount$
18 — compareClass ($c_d$, $c_o$, elementsToIgnore)
19 foreach $a_d \in c_d.attributes$ do
20     if $a_d \notin elementsToIgnore$ then
21         foreach attribute $a_o \in c_o$ do
22             $r \leftarrow r + compareAttribute(a_d, a_o, elementsToIgnore)$
23     if $r \equiv 0$ then
24         $matches \leftarrow matches \cup (a_d, a_o)$
25     else
26         $closestMatches \leftarrow closestMatches \cup (a_d, a_o, r)$
27 return $r$
28 — compareAttribute ($a_d$, $a_o$, elementsToIgnore)
29 if
   $a_d.lowerBound \leq a_o.lowerBound \wedge a_o.upperBound \leq a_d.upperBound$
   then
30     if $a_d.type \equiv STRING$ then
31         $matches \leftarrow matches \cup (a_d, a_o)$
32         return 0
33     else
34         if $a_d.type \equiv RELATION \wedge a_o.type \equiv RELATION$ then
35             $r \leftarrow compareRelationship(a_d, a_o, elementsToIgnore)$
36             if $r \equiv 0$ then
37                 $matches \leftarrow matches \cup (a_d, a_o)$
38             return $r$
39         else if $a_d.type \equiv a_o.type$ then
40             $matches \leftarrow matches \cup (a_d, a_o)$
41             return 0
42         else
43             $closestMatches \leftarrow closestMatches \cup (a_d, a_o, 1)$
44             return 1
45 else
46     $closestMatches \leftarrow closestMatches \cup (a_d, a_o, 1)$
47     return 1
48 — compareRelationship ($r_d$, $r_o$, elementsToIgnore)
49 $relsToIgnore \leftarrow relsToIgnore \cup (r_d)$
50 $r \leftarrow compareClass(r_d.start, r_o.start, relsToIgnore)$
51 $r \leftarrow r + compareClass(r_d.end, r_o.end, relsToIgnore)$
52 if $r \equiv 0$ then
53     $matches \leftarrow matches \cup (r_d, r_o)$
54 else
55     $closestMatches \leftarrow closestMatches \cup (r_d, r_o, r)$
56 return $r$
```

Comparing models is done by comparing respective classes with each other. In order to compare two classes the *compareClass*$(c_d, c_o, elementsToIgnore)$ function (line 18) is used. The comparison is performed by analyzing and comparing the attributes of the classes. Every attribute of a class from the demand model is compared to each attribute of the offer model class (line 22). The result of the comparison is either 1 or 0, whereby 0 indicates a match of the attributes. This indicator is summed up over all attributes and returned as the result of *compareClass*. The matched pair of attributes is collected (line 24) for later use in the configuration of the visualization.

Attributes are compared calling *compareAttribute* (line 22). This function returns either 0 indicating a match or 1 for a mismatch. For a structural match, attributes are compared by their data types and cardinalities. First we check whether cardinalities of an attribute in m_o satisfy the multiplicity of the demand model attribute (line 29). In the next step, we check the data type of the demand model attribute. In line with Schweda [34], the algorithm takes into account that data types {*byte, integer, decimal, boolean, date, time, dateTime, duration, URI, ...*} (cf. also [37]) have a lexical representation[5], i.e. they can be converted to a string. That is, when an attribute a_d is of type string, then a_o matches irrespective of its own data type. We assume that even relationships can be converted to string representations. Attribute a_d may also be a relationship (line 35) or of any other type (line 39).

Relationships between classes are compared in *compareRelationship*. The algorithm compares source (line 50) and target (line 51) classes of a relationship utilizing the *compareClass* (line 18). In order to avoid circularity the current relationships are ignored while comparing the classes (Line 49). If both start and end of r_d match r_o, then *compareRelationship* returns 0 and respective information of the match is added to the result. To avoid infinite recursion through recursive relationships of a model, the algorithm takes into account collection *elementsToIgnore* representing elements that are not traversed any further.

During the design phase of the algorithm, we considered that an algorithm could extend m_o in a first step. Such an extension would basically transform the offer model in such a way that attributes become relationships to classes whereas the type of these classes equals the respective primitive type of the attribute. This way, on the one hand, m_d is minimal in terms of required information to generate the visualization and on the other hand, chances that m_o fits m_d are increased. However, although this is technical feasible, end-user feedback led our design into a different direction. The major reason for our design decision was that such a solution would produce unnecessary choices and thus possibly confuse end-users.

[5]http://www.w3.org/TR/xmlschema-2 last-accessed: April 20, 2013.

IV. Prototypical Implementation of the Visualization Wizard

The pattern matching algorithm described above serves as a basis for our visualization wizard to recommend feasible configurations to stakeholders. This wizard guides the user through three essential steps. First, a predefined abstraction viewpoint, e.g. a binary matrix, a portfolio diagram, or a cluster map, has to be selected (cf. Figure 5 ❶). Our framework can be extended by new abstract viewpoint definitions.

Every abstract viewpoint definition contains both a sample preview of the viewpoint (cf. Figure 5 ❷) and a potential information model (Figure 5 ❸). The screenshot in Figure 5 shows a ternary matrix to realize a process support map (see [13]). Furthermore, an explanation of the ternary matrix concept (cf. ❹ in Figure 5) and a typical usage scenario for EAM is given (❺ in Figure 5).

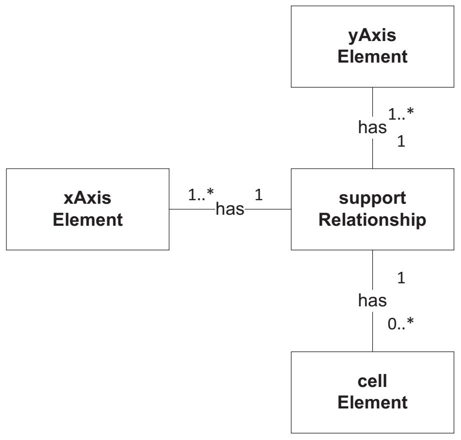
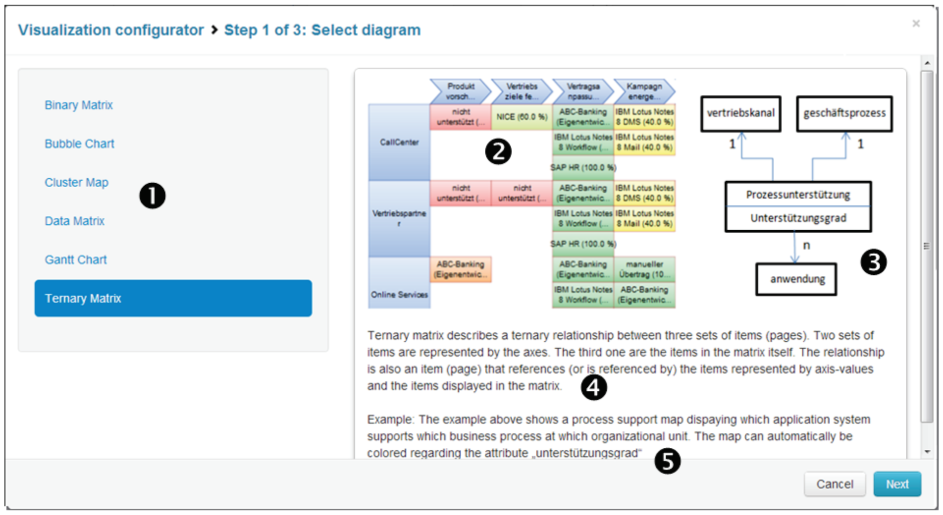


Figure 3. Abstract view model (m_{d*}) of a ternary matrix

When an abstract viewpoint has been selected, the wizard guides the user through the configuration of the data binding between the EA information model and the abstract viewpoint. The information demand for the chosen abstract viewpoint is captured by the abstract view model, which describes the information required to generate the visualization. Figure 3 shows the respective abstract view model of a ternary matrix, i.e. x-axis, y-axis, and a cell element connected via a relationship such that each triple can be identified uniquely. Technically, elements of this abstract view model have to be bound to elements of an EA information model in order to generate a concrete visualization, i.e. mappings to an EA information model has to be provided by the user.

Although the EA information model can be an arbitrary information model for our approach, we use a concrete EA information model to exemplify our approach. Thereto, Figure 4 illustrates a subset of the ArchiMate 2.0 [20] specification. Subsequently, it serves as an example for an



Ternary matrix describes a ternary relationship between three sets of items (pages). Two sets of items are represented by the axes. The third one are the items in the matrix itself. The relationship is also an item (page) that references (or is referenced by) the items represented by axis-values and the items displayed in the matrix.

Example: The example above shows a process support map dispaying which application system supports which business process at which organizational unit. The map can automatically be colored regarding the attribute „unterstützungsgrad"

Figure 5. First step of the visualization wizard: diagram selection

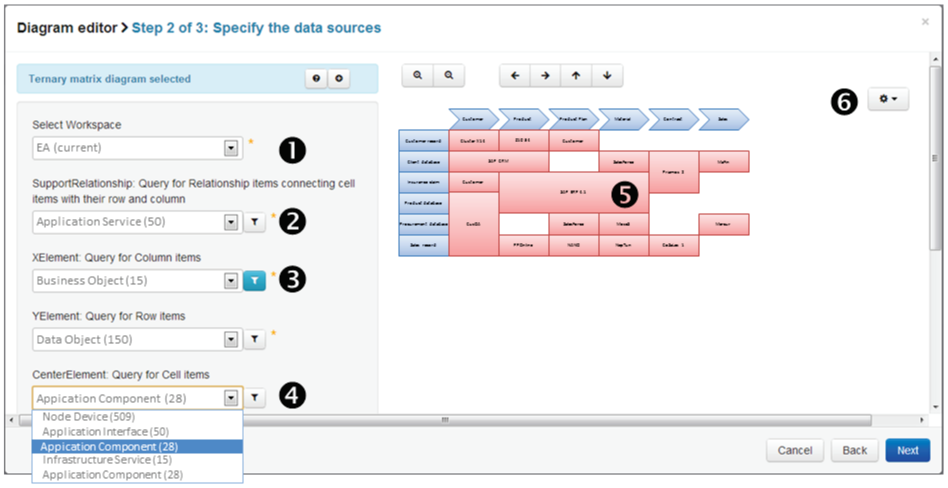


Diagram editor > Step 2 of 3: Specify the data sources
Ternary matrix diagram selected
1
2
3
4
5
6
Node Device (509)
Application Interface (50)
Appication Component (28)
Infrastructure Service (15)
Application Component (28)
Cancel
Back
Next

Select Workspace: EA (current)

SupportRelationship: Query for Relationship items connecting cell items with their row and column: Application Service (50)

XElement: Query for Column items: Business Object (15)

YElement: Query for Row items: Data Object (150)

CenterElement: Query for Cell items: Appication Component (28)

Figure 6. Second step of the visualization wizard: configuration of the data binding

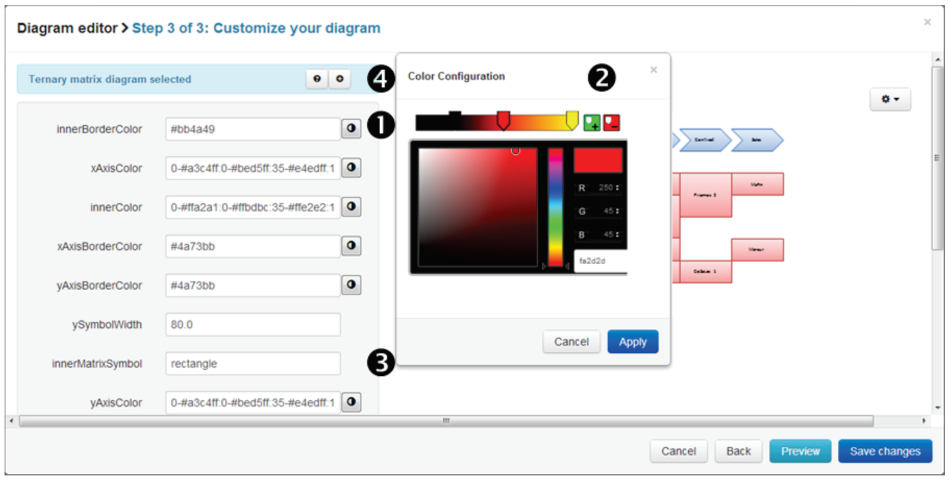


Diagram editor > Step 3 of 3: Customize your diagram
Ternary matrix diagram selected
Color Configuration
1
2
3
4
R 250
G 45
B 45
fa2d2d
Cancel
Apply
Cancel
Back
Preview
Save changes

innerBorderColor	#bb4a49
xAxisColor	0-#a3c4ff:0-#bed5ff:35-#e4edff:1
innerColor	0-#ffa2a1:0-#ffbdbc:35-#ffe2e2:1
xAxisBorderColor	#4a73bb
yAxisBorderColor	#4a73bb
ySymbolWidth	80.0
innerMatrixSymbol	rectangle
yAxisColor	0-#a3c4ff:0-#bed5ff:35-#e4edff:1

Figure 7. Third step of the visualization wizard: customize diagram

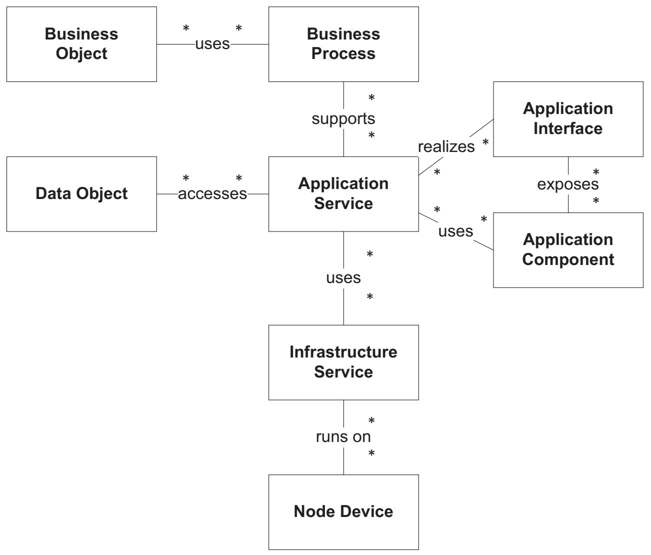


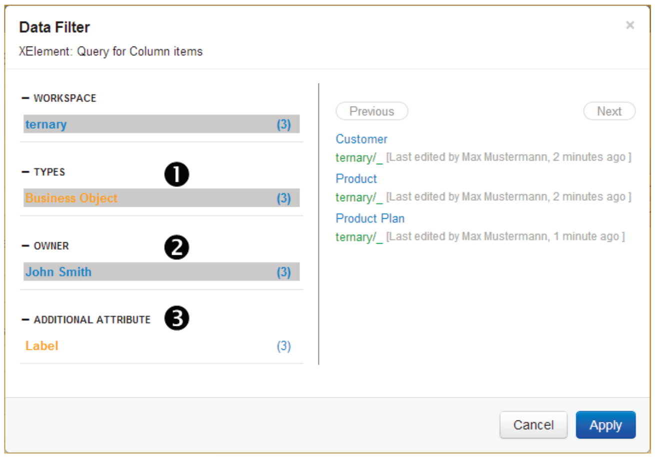
Figure 4. Illustrating example of an EA information model (m_{o*})

EA information model frequently applied in practice, which captures the following basic concepts and their interrelations:

- business layer, i.e. business processes and business objects,
- application layer, i.e. application service, application component, application interface, and data object, and
- infrastructure layer, i.e. node devices.

Algorithm 1 applied to m_{d*} and m_{o*} provides valid, i.e. feasible, configurations of data bindings between the abstract viewpoint (m_{d*}) and the EA information model (m_{o*}). The wizard assists stakeholders in the configuration of the desired viewpoint since only feasible configuration settings can be specified. For the selected ternary matrix (see Figure 6), the user can choose the EA information model (Figure 6 ❶) and the support relationship items (Figure 6 ❷). The number of respective instances for every type, attribute, and relationship is given in parentheses. The user can select elements of m_{o*} to define a mapping to elements of m_{d*}. For each of this elements the user can refine the selection of objects with the instance filter as shown in Figure 8. In the illustrated example, a filter already has been defined and is active for the type business object (Figure 6 ❸).

The filter dialog shows the context, i.e. selected model and type (Figure 8 ❶), and filters business objects by their attribute owner for a particular value, i.e. for the owner 'John Smith' (Figure 8 ❷). Additional attribute filters can be defined and applied in conjunction (Figure 8 ❸). The wizard further lists available relationships including transitive relationships (Figure 6 ❹). Since only feasible configurations are shown to the user, a generic preview of the actual visualization can be generated and is shown to the user (Figure 6 ❺).


Data Filter

XElement: Query for Column items

– WORKSPACE

ternary (3)

– TYPES ❶

Business Object (3)

– OWNER ❷

John Smith (3)

– ADDITIONAL ATTRIBUTE ❸

Label (3)

Customer
ternary/_ [Last edited by Max Mustermann, 2 minutes ago]

Product
ternary/_ [Last edited by Max Mustermann, 2 minutes ago]

Product Plan
ternary/_ [Last edited by Max Mustermann, 1 minute ago]

Figure 8. Data filter in the visualization wizard

In the third and final step, visual properties of the visualization may be customized (Figure 7). A variety of customization options are available for any given abstract viewpoint. For instance, the border color of the cell elements can be specified (Figure 7 ❶) or the symbol shapes can be customized, e.g. from rectangle to chevron shapes (Figure 7 ❷). Each visualization generated can be downloaded as PowerPoint (.pptx) slide (Figure 6 ❻) which can be easily modified and included in an executive presentation.

V. Conclusion and Outlook

Due to human cognition, visualizations are essential to support stakeholders with relevant information about an organization's architecture. Analyzing EA models using EA visualizations is often difficult because of inflexible configurations in prevailing tools that can only be created by experts with strong technical background. Most likely, an expert with programming or modeling skills is required. As a result, ad-hoc analyses of the organization's architecture often can not be performed in a timely fashion. However, stakeholders require instantaneous solutions in practice to satisfy ad-hoc information demands.

In this paper, we presented a lightweight meta-information model for matching structures of different models and an algorithm operating on respective models to calculate feasible configurations for bindings of visualizations to information. We further presented an implementation of our concepts in a user-friendly wizard to support stakeholders with the generation of EA visualizations and exemplified our approach based on an EA information model conforming to the ArchiMate 2.0 specification [20]. Thereby, we revealed user interface design and implementation details of the wizard. The wizard automatically recommends feasible configurations to the stakeholder in order to simplify the configuration process. First end-user feedback is positive. Currently, we are evaluating the presented approach with industry in real-world settings.

References

[1] P. Weill and J. Ross, *IT Savvy: What Top Executives Must Know to Go from Pain to Gain*. Harvard Business Press, 2009.

[2] L. von Bertalanffy, *General system theory: foundations, development, applications*, ser. International library of systems theory and philosophy. G. Braziller, 1968.

[3] 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.

[4] F. Matthes, "Softwarekartographie," *Informatik Spektrum*, vol. 31, no. 6, 2008.

[5] J. Heer, "Interactive Dynamics for Visual Analysis," *Communications of the ACM*, vol. 55, no. 4, 2012.

[6] R. Spence, *Information visualization*, ser. ACM Press books. Harlow, England: Addison-Wesley, 2001.

[7] S. Buckl, F. Matthes, S. Roth, C. Schulz, and C. M. Schweda, "A conceptual framework for enterprise architecture design," in *TEAR*, ser. Lecture Notes in Business Information Processing, E. Proper, M. M. Lankhorst, M. Schönherr, J. Barjis, and S. Overbeek, Eds., vol. 70. Springer, 2010, pp. 44–56.

[8] International Organization for Standardization, "ISO/IEC 42010:2007 Systems and software engineering – Recommended practice for architectural description of software-intensive systems," Geneva, Switzerland, 2007.

[9] R. I. Bull, "Integrating dynamic views using model driven development," *Proceedings of the 2006 conference of the Center for Advanced Studies on Collaborative research - CASCON '06*, p. 17, 2006.

[10] S. Kruse, J. Addicks, M. Postina, and U. Steffens, "Decoupling models and visualisations for practical ea tooling," in *Service-Oriented Computing. ICSOC/ServiceWave 2009 Workshops*, ser. Lecture Notes in Computer Science, A. Dan, F. Gittler, and F. Toumani, Eds., vol. 6275. Springer Berlin/Heidelberg, 2010, pp. 62–71.

[11] M. Schaub, F. Matthes, and S. Roth, "Towards a conceptual framework for interactive enterprise architecture management visualizations," in *Modellierung*, Bamberg, Germany, 2012.

[12] A. M. Ernst, "A pattern-based approach to enterprise architecture management," Ph.D. dissertation, Technische Universität München, München, Germany, 2010.

[13] ——, "Enterprise architecture management patterns," in *PLoP 08: Proceedings of the Pattern Languages of Programs Conference 2008*, Nashville, USA, 2008.

[14] C. Alexander, S. Ishikawa, M. Silverstein, M. Jacobson, I. Fiksdahl-King, and S. Angel, *A Pattern Language*. New York, NY, USA: Oxford University Press, 1977.

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

[16] Department of Defense (DoD) USA, "DoD Architecture Framework Version 2.0: Volume 1: Introduction, Overview, and Concepts – Manager's Guide," 2009.

[17] J. A. Zachman, "A framework for information systems architecture," *IBM Syst. J.*, vol. 26, no. 3, pp. 276–292, 1987.

[18] M. M. Lankhorst, H. A. Proper, and H. Jonkers, "The architecture of the ArchiMate language," *Enterprise, Business-Process and Information Systems Modeling*, pp. 367–380, 2009.

[19] Y. Lee, "Information modeling: From design to implementation," in *Proceedings of the Second World Manufacturing Congress*. Citeseer, 1999, pp. 315–321.

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

[21] S. Aier, "Understanding the role of organizational culture for design and success of enterprise architecture management," in *Proceedings of the 11th International Conference on Wirtschaftsinformatik WI*, 2012.

[22] 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.

[23] M. Farwick, R. Breu, M. Hauder, S. Roth, and F. Matthes, "Enterprise architecture documentation: Empirical analysis of information sources for automation," in *46th Hawaii International Conference on System Sciences (HICSS), Maui, Hawaii*, 2013.

[24] S. Roth, M. Hauder, F. Michel, D. Münch, and F. Matthes, "Facilitating conflict resolution of models for automated enterprise architecture documentation," in *19th Americas Conference on Information Systems (AMCIS)*, 2013.

[25] S. Roth, M. Hauder, M. Farwick, R. Breu, and F. Matthes, "Enterprise architecture documentation: Current practices and future directions," in *11th International Conference on Wirtschaftsinformatik (WI), Leipzig, Germany*, 2013.

[26] M. Hauder, F. Matthes, and S. Roth, "Challenges for automated enterprise architecture documentation," in *Trends in Enterprise Architecture Research (TEAR) and Practice-Driven Research on Enterprise Transformation (PRET)*. Springer, 2012, pp. 21–39.

[27] J. Mackinlay, "Automating the design of graphical presentations of relational information," *ACM Trans. Graph.*, vol. 5, no. 2, pp. 110–141, Apr. 1986.

[28] T. Panas, R. Lincke, and W. Löwe, "Online-configuration of software visualizations with Vizz3D," in *In Pro-ceedings of the 2005 ACM Symposium on Software Visualization (SoftVis05)*, vol. 1, no. 212, 2005, pp. 173–182.

[29] T. Goldschmidt, S. Becker, and E. Burger, "Towards a tool-oriented taxonomy of view-based modelling," in *Modellierung*, ser. LNI, E. J. Sinz and A. Schürr, Eds., vol. 201. GI, 2012, pp. 59–74.

[30] M. Buschle, M. Ekstedt, S. Grunow, M. Hauder, F. Matthes, and S. Roth, "Automated enterprise architecture documentation using an enterprise service bus," in *AMCIS*. Association for Information Systems, 2012.

[31] S. Buckl, F. Matthes, I. Monahov, S. Roth, C. Schulz, and C. M. Schweda, "Towards an agile design of the enterprise architecture management function," in *EDOCW*. IEEE Computer Society, 2011, pp. 322–329.

[32] S. Roth and F. Matthes, "Future research topics in enterprise architectures evolution analysis," in *Design for Future (DFF) Workshop*, 2013.

[33] M. Berneaud, S. Buckl, A. Diaz-Fuentes, F. Matthes, I. Monahov, A. Nowobliska, S. Roth, C. M. Schweda, U. Weber, and M. Zeiner, "Trends for enterprise architecture management tools survey," Technische Universität München, Technical Report, 2012.

[34] C. M. Schweda, "Development of organization-specific enterprise architecture modeling languages using building blocks," Ph.D. dissertation, Fakultät für Informatik, Technische Universität München, Germany, 2011.

[35] M. Hauder, F. Matthes, S. Roth, and C. Schulz, "Generating dynamic cross-organizational process visualizations through abstract view model pattern matching," in *Architecture Modeling for Future Internet enabled Enterprise (AMFInE)*, 2012.

[36] M. Postina, J. Trefke, and U. Steffens, "An ea-approach to develop soa viewpoints," in *Enterprise Distributed Object Computing Conference (EDOC), 2010 14th IEEE International*. IEEE, 2010, pp. 37–46.

[37] International Organization of Standardization, "ISO/IEC 11404:2007 Information technology – General-Purpose Datatypes (GPD)," Geneva, Switzerland, 2007.