Towards a Systematic Approach for Capturing Dynamic Transformation in Enterprise Models

Stephan Aier
Institute of Information Management
University of St. Gallen, Switzerland
stephan.aier@unisg.ch

Bettina Gleichauf
Institute of Information Management
University of St. Gallen, Switzerland
bettina.gleichauf@unisg.ch

Abstract

The transparency created by enterprise models is a valuable support for organizational engineering and especially enterprise transformation. However, current approaches are hardly suitable to also create transparency of the enterprise transformation process itself. The paper at hand contributes a systematic approach for capturing dynamics of enterprise transformation in enterprise models. Therefore we differentiate dimensions of time (modeling time, valid time) of enterprise models on a macro level as well as a set of model transformation operations on a micro level. The proposed model transformation operations on a micro level are suitable for describing the differences of as-is and to-be models in an enterprise transformation program.

1. Introduction

Enterprise modeling (EM) addresses the modeling of business processes, goals, strategy, information entities, business structure, support systems, skills and people of an enterprise [11]. Thus, enterprise models integrate conceptual models of information systems and models of supported business functions and can provide the necessary transparency for a systematic support to organizational engineering [9]. By representing both organizational and technical infrastructure, EM may also provide a broad and consolidated view of an entire corporation or government agency [17], also known as enterprise architecture [15, 20, 26, 32, 35].

The complexity and thus maintainability of enterprise models leads to a trade-off between detailed but isolated models on the one hand side or aggregated but broadly connected models on the other hand side [18]. In order to provide support for transformation in an efficient way, EM has to be driven by business and/or IT oriented application scenarios [34] based on stakeholders concerns [22, 23, 36] (*goal orientation*) [20, 26]. Since the involvement of heterogeneous stakeholder groups may create conflicting requirements in a complex environment, an appropriate documentation and communication of the enterprise models is vital. A suitable degree of *formalization* is needed in order to ensure traceable and repeatable results [33]. Further-

more (semi) formalized models and well structured methods are needed to enable division of labor and common understanding among the stakeholder groups [12, 14].

While documentation and analysis of enterprise models (represented by as-is models) are well covered in academic and practitioner approaches, EM for planning transformation purposes is covered much less so far. Since neither the corporation or government agency itself, nor its environment remains static during a transformation project, and because to-be models may change as projects are launched, the consideration of dynamic aspects is important for planning a transformation program.

As we will illustrate in the next section there are already a few contributions that give first structure to the challenges of EM for planning purposes, e.g. [1, 6]. Based on these contributions this paper aims at answering the following research questions:

1. How can dynamics be represented in enterprise models?
2. What implications arise for the characteristics of enterprise meta models?
3. How can consistency between enterprise model snapshots of different points in time be assured?

For answering these questions the core contribution of this paper will be a framework of temporal dimensions that need to be covered by enterprise models. However, the paper does not consider the planning of future states of a corporation or government agency by the means of enterprise models but the transformation of current states to planned states. Therefore it is presumed that at least a to-be model exists and that the decision about the planned implementation/transformation has already been made.

The remainder of this paper is structured as follows. Section 2 will derive requirements for capturing dynamic transformations in EM from related work in the field of enterprise architecture. Section 3 will review solution components from related work in the fields of software and database engineering, concerned with representing dynamics and time. In section 4 we will present our proposal for capturing dynamics in EM, including a case study demonstrating its application. The article closes with a discussion and an outlook.

2. Related approaches for capturing transformation in enterprise models

Up to now only a few approaches for planning in the large based on and reflected in enterprise models exist. While there are various contributions dealing with related questions, none of the existing approaches addresses enterprise planning from business to IT covering artifact relationships in semi-formal models and/or addressing model dynamics.

Historically, the topic evolved from strategic IS planning which was firstly addressed in an MISQ contribution by King in 1978 [16]. This paper proposes a process to design a management information system (MIS) in accordance to the strategy of a corporation or government agency and thereby define a MIS strategy comprising MIS objectives and MIS constraints. As markets, organizational structures and system landscapes added more complexity to the matter of strategic planning and the alignment of business and IT, this approach as well as similar contributions were evolutionary refined. Strategic enterprise-wide information management [31] and more institutionalized IS planning processes became an issue in the 1990ies [10]. A prominent example for IS planning methods is IBM's Business System Planning (BSP) [13]. BSP aims to (re-)group IT functionalities according to data use and thereby identify application candidates with high internal integration intensity, but limited external interfacing to other applications.

Especially the field of enterprise architecture (EA) is concerned with aggregated models covering a broad scope from business to IT. Therefore also in EA related approaches for planning and transformation were developed e.g. by Spewak (the wedding cake model) [29, 30], Pulkkinen and Hirvonen [24, 25], Op't Land et al. [23] and Niemann [21].

However, the majority of research results only focus on a unidirectional planning process that aims at improving the current structure. This includes a defined start date and end date of the process as well as a defined result, i.e. one target for one point in time. In addition, most sources cover individual dynamic aspects such as adaptations of target models to changing conditions, life cycles of individual artifacts, the evaluation of model alternatives or the support of transformation from as-is to to-be models.

Only recently the works of Buckl et al. [5, 6] and Aier et al. [1] address a comprehensive modeling method for planning purposes in the EA context. While Buckl et al. propose a set of meta model requirements for modeling temporal aspects, their proposal focuses on application landscapes [6]. However, they take into account important temporal dimensions, e.g. the time a

model is created and the time a model should be valid for the past, the present and the future, as well as different variants of future models [6].

Also Sousa et al. take the dynamic evolution of enterprise models into account, providing a formal approach for generating blueprints of enterprise architecture from existing project management sources [28]. By applying this bottom-up approach, the current and the planned state of the enterprise architecture as well as its evolution can be visualized. However, this approach reconstructs EA from existing project plans rather than considering planning EA as a whole.

Aier et al. address different complexity levels for representing dynamics in EA planning [1]. On the highest of their seven complexity levels a comprehensive planning and transformation approach has to comply with the following requirements:

- Model as-is state
- Model to-be states
- Model transformation paths from as-is to to-be states
- Model alternatives for models and paths for a respective point in time
- Model an unlimited number of points in time as well as the respective transformation paths
- Model deviations from plans

These requirements set an environment in which transformation between as-is and to-be states can be planned. However, the work of Aier et al. does not elaborate on the mechanisms that can handle the different model snapshots, how transformation can be captured within models or how transformation paths can be derived.

As a consequence—although contributing to—none of the discussed contributions delivers an approach to graph dynamic transformation in enterprise models. In order to do so, a further construct is necessary to analyze and describe the differences between as-is and to-be models in general as well as on a model element level taking the complex impact of time into account. The following section will analyze related approaches from software and database engineering.

3. Related work from other fields

3.1. Representing time

The challenge to capture time in order to process temporal information in information systems has been discussed in the fields of object-oriented programming and database design. Therefore it seems useful to take those approaches into consideration for enterprise modeling.

In object-oriented design, the concept of patterns is widely used to develop design solutions to recurring problems. With regard to the representation of temporal aspects Carlson et al. developed a set of temporal patterns [7]. Among them the patterns *Temporal Property* and *Temporal Association* enable a system to capture properties and relationships that change over time. Additionally, the pattern *Snapshot* can be used to introduce a snapshot object that represents a temporal property and/or relationship at a certain point in time.

Another approach regarding temporal patterns is the proposal of Arnoldi et al. [3]. In order to capture what changes happened when, they distinguish *History*, the part that doesn't change, and *Version*, the changing part of an object. This idea is represented by the pattern *Version History*, which defines two classes of objects. The history object acts as a container of the version objects that represent different versions of the conceptual object. Enhancing these two classes, the pattern *Perspective* is proposed which represents that date on which a version is committed to its history.

The basic idea of distinguishing between historiography or versioning information on the one hand and actual changes on the other hand is described in detail in the *bitemporal table* concept presented by Snodgrass [27]. Bitemporal tables record two dimensions of time:

- *valid time*, i.e. the time of changing reality or the time an enterprise model is built for
- *transaction time*, i.e. the time of states of a changing table or the time an enterprise model is created

Transaction time could also be called construction time or version. Both time dimensions are covered by respective timestamps within the table. The coverage of both time dimensions allows to capture changing information and to keep historical data at the same time. Furthermore, bitemporal tables enable queries on either dimension as well as on the interdependencies between both. Therefore it is possible to analyze which information was recorded when as well as snapshots of a certain state of information at a certain point in time.

3.2. Dynamic software architectures

The task of modeling complex systems and architectures has been addressed in the field of software engineering, especially software architectures for a couple of years. In this context, architecture description languages (ADL) are used to specify architectures and can be seen as more specific counterparts to EA meta modeling languages [19]. A long list of scientists have dealt with the topic how to model and build software architectures that are able to change over time and to react to externally triggered changes. Such architec-

tures are called *Dynamic Software Architectures* or *Self-Managed Software Architectures*.

The concept of architecture based evolution presented by Cimpan et al. intends to prepare architectures for dynamic changes by providing "evolution entry points", using a specialized ADL [8].

The authors distinguish planned and unplanned evolutions that need to be anticipated. For both types of evolution the architecture is prepared with a specialized architectural element that is able to receive changing information from outside. Such information comprises deletion and creation of connectors as well as new components to be connected and might be incorporated in a file format.

A specialized approach compared to dynamic or evolving software architecture are self-managing architectures. Such architectures are implement to respond to externally triggered change, but also to initiate, select, and assess the change autonomously [4]. Bradbury et al. state that all dynamic changes have four steps: (1) initiation of change, (2) selection of architectural transformation, (3) implementation of reconfiguration and (4) assessment of architecture after reconfiguration [4]. With regard to the reconfiguration they identify four possible operations, namely the addition and removal of components and connectors.

4. Proposal for modeling dynamic transformation

The attempt to fully comprehend the multitude of temporal aspects as well as their impact on model transformation is a highly complex task. This assessment is backed up by the findings from related fields.

Besides that, enterprise models themselves are complex, usually consisting of multiple sub-models and elements. Thus, the question arises where to start and where to model transformation and transformation planning, especially because the number of hierarchy levels contained in enterprise models varies from case to case. Therefore, we propose to examine the dynamic transformation of enterprise models from two separate perspectives, expressed by two "examination levels": First, we will describe a macro level which focuses on whole models, e.g. an application landscape model, as the main concepts that are planned as a basis for transformation. Second, we zoom in on the elements of the temporal relationship between two models, e.g. applications and its predecessors/successors, and describe the concepts that are needed to realize transformation on a micro level. Both levels have a clear mutual hierarchical relationship while their application on enterprise modeling levels is flexible.

The presented approach has to be considered as a theoretical framework that guides the modeling of dynamics within a planning process. In order to illustrate its practical application, we will present a short case study in section 4.3.

4.1. Macro level

On the macro level we consider the planning of to-be models for different points in time. In doing so, the two time dimensions valid time and transaction time presented by Snodgrass [27] are applicable, which has also been shown by Buckl et al. [5]. Because enterprise modeling deals with modeling tasks rather than with transaction tasks, we rename transaction time *modeling time*, i.e. the time a certain enterprise model is created or updated. Hence, the modeling time dimension is used to capture versioning of enterprise models.

Figure 1 depicts the valid time on the horizontal axis and the modeling time on the vertical axis. While valid time is only sectioned in years, modeling times

are shown in irregular intervals. A different granularity of both time dimensions reflects realistic conditions: While valid time intervals can be interpreted as release planning cycles, modeling time intervals might be subject to unplanned impacts [cf. 8] and therefore occur irregularly. Such an impact is shown exemplary in Figure 1 for May 2009. The necessity to consider irregular re-planning of models has already been explicated by Aier et al. [1]. It requires a certain degree of dynamic in the planning and transformation process.

Following the explanations of Snodgrass, valid time and modeling time can be recorded by timestamps [27]. For reasons of clarity, only the *valid from* date of each model is depicted in Figure 1. Although the information on the sequence of the different planned models is immanent in the valid time, the models are additionally named A.0, A.1 and so on in order to visualize their sequence. The sequence is valid for a certain point in modeling time, visualized as a perspective following the term used by Arnoldi et al. [3].

Figure 1. Macro level

Arrows between two successive models depict the transformation paths. According to Cîmpan et al. the transformation information, i.e. the information how to get from one state to the next, should be captured in a construct that is external to the actually considered model [8]. Moreover, this construct should be built (and the transformation information be formalized) before the reconfiguration is effectively applied. Thus, transformation between enterprise models can be represented in transformation models. Transformation models are depicted as black boxes in Figure 1. In analogy to enterprise models, they are labeled with a name which is abbreviated in the figure for the sake of clarity. The inner structure of transformation models will be closer examined in the next section.

With proceeding time, planned models become current models (visualized by dark grey boxes). At this point, we abstain from labeling the models as *as-is* or *to-be* models because this information is more formally expressed by the validity timestamps in conjunction with actual time.

Finally, new versions of the enterprise models are developed, which is visualized in Figure 1 along the vertical axis. In doing so, not necessarily all models are changed but some are simply taken over from the last point in modeling time (visualized by a dashed line). As models change, the transformation path between them, i.e. the transformation model, also changes. Figure 1 depicts this by displaying versions of the transformation models.

Figure 1 depicts this by displaying versions of the transformation models.

4.2. Micro level

On the micro level (cf. Figure 2) we examine the inner structure of the respective planning subjects, i.e. the elements of the models that are an initial point and target of the planning intentions. In doing so, the micro level details the valid time dimension defined on the macro level because this is the dimension where actual transformation planning activities take place. Thus, the following question is addressed: Given a model A.0 and successor model A.1, how can transformation from A.0 to A.1 be planned?

In planning the transformation of enterprise models, we follow the process steps proposed by Bradbury et al. [4]. The change initiation is done by the decision to plan a transformation between two enterprise models and might be triggered by different events. (The precise process for initiating transformation planning is not subject to this work.) A selection of affected architectural elements and an appraisal of the necessary transformation steps needs to be performed before the transformation can be actually implemented. (Also the approach presented by Cîmpan et al. proposes the encapsulation of transformation information [8].)

Figure 2. Micro level step 1: analysis of deviations

In a first step, this requires an *analysis of the deviations* between the models by answering the following question: Which elements in A.1 are successors of elements in A.0? In this case, we understand elements as basic items depending on the level of consideration. On both sides of a successor relationship the elements are of the same type. Answers lead to different cases for each considered element:

1. Relationship 1:1

One element in model A.0 has exactly one successor in model A.1. The example in

Figure 2 shows element 11 in model A.1 as a successor of element 4 in model A.0. Element 1 represents a special case of explicit successorship, though: Here, the element in model A.0 is identical to the element in model A.1. This is emphasized in the picture by an identical label.

2. Relationship 1:n

One element in model A.0 has more than one successor in model A.1. This might be the case if a component is decomposed in the future model. In the example element 2 in model A.0 is decomposed into elements 9 and 10.

3. Relationship n:m

Several elements in model A.0 have multiple successors in model A.1. This reflects complex restructuring of elements in the future model. In the example,

elements 7 and 8 in model A.0 are rearranged in elements 14 and 15 in model A.1. In this case, the successor relationships cannot be expressed in 1:n or n:1 relationships, because different parts of the elements 7 and 8 might be rearranged in both the new elements 14 and 15.

4. Relationship n:1

Several elements in model A.0 have exactly one successor in model A.1. This is the case if an aggregation of elements is planned in the future model. The example shows elements 3 and 5 which are aggregated into element 12 in model A.1.

In addition to cases 1 to 4, two special cases can be identified:

5. Relationship 1:0

One element in model A.0 has no successor in model A.1. This reflects a termination of an element in the future model like it is exemplified by the element 6 in Figure 2.

6. Relationship 0:1

One element in model A.0 has no predecessor in model A.0. This represents a new element in model A.1. In the example, element 13, that does not replace an existing element in model A.0, is added to element 12 in model A.1.

Legend:

Figure 3. Micro level step 2: scheduling of transformation steps

As a second step in transformation planning, the individual *transformation steps* need to be *scheduled*. Thereby sequenced, conjoint or parallel processing of affected elements can be performed. As a result, a procedure model is established that serves as a guideline for the transformation realization. An example for such a procedure model is depicted in Figure 3. In order to

visualize the sequenced and parallel steps, a simplified version of the UML notation for activity diagrams is used [2]. In doing so, the actions *delete x* and *create y* represent the deletion and creation of model elements.

Depending on the actual dependencies between model elements, constraints arise for the possible sequences of transformation steps. For example, two

initial nodes depict that both the deletion of elements 2 and 4 as well as 7 and 8 can be performed in parallel, because they are not interrelated. The diagram shows multiple parallel actions (visualized by fork/join nodes) which are possible if elements can be deleted or created independently. However, several alternative sequences might be possible. For example, the deletion of elements 3 and 5 could as well be parallelized to the creation of elements 9 and 10. Moreover, in the case of a large number of changes to be performed, it can be useful to group these changes into atomic sub-sets which are to be performed together. In general, there is not necessarily a single solution for a model specifying the transformation steps between two enterprise models.

4.3. Case study

The following case describes the IT planning process at a large banking solutions provider and shows how the presented framework can be applied. The company's primary product is an integrated banking platform that is offered to private banks and universal banks. The development activities are planned and controlled by the architecture team using a home grown solution to create to-be models and manage development projects within the banking platform. Major challenges within the architectural development plan are the coordination of the activities of the development teams and assurance that milestones of the various integration and development activities are met simultaneously. The simple requirement of coordinated and integrated development becomes very demanding as the banking platform comprises of over 200 applications, each consisting of a multitude of components that each have their own lifecycles as well as predecessor and successor relationships.

In order to cope with the complexity resulting from a large number of perspectives in the form of operating models, hardware models, application landscape models, component models and interface models, the planning activities are split up. On the one hand, high level roadmaps for the operating and IT architecture are defined. These consist of models representing a snapshot of the desired architecture for up to three points in time in the future, taking into account existing vendor specific constraints if applicable. This corresponds to the macro level presented above. However, the concept of versioning is not explicitly used in this case but models are regularly adapted to represent the current state.

The detailed planning, i.e. the planning of the development of the individual components, is performed on another level. Based on the requirements specified by the operating architecture roadmap model, the need for updating, replacing or buying new middleware and

hardware components is deduced. This is done by matching the current architecture model to the to-be model of a future valid time timestamp (cf. the analysis of deviations described above). The same applies for the IT architecture roadmap and the resulting development needs regarding the applications of the banking platform. It is then possible to define which changes regarding middleware and hardware components as well as applications have to be done in which sequence. This determines the rough project program schedule in order to implement the transformation.

Of course, the analysis of applications is not sufficient in terms of development planning. Therefore, also to-be models of the application landscape are modeled (macro level) and analyzed in detail regarding the required changes in terms of components and interfaces to be altered, taken out of service or introduced (micro level). Finally, this analysis enables the definition and scheduling of development projects.

5. Discussion

At the macro level, the presented proposal provides a structure for temporal dimensions that are relevant for modeling and representing dynamic transformation. Thereby, approaches from database and software engineering deliver useful ideas concerning the representation of time in enterprise models. The use of different timestamps in enterprise models enables analyses of different points in time, and ensures a historiography of enterprise model versions at the same time.

Beyond that the use of a new model type, the transformation model, was deduced from approaches in dynamic software architecture engineering. By means of this model type the transformation can be planned and the relevant information can be stored separately from the enterprise models.

At the proposed micro level, the transformation planning is performed by analyzing the initial and the target enterprise model and establishing a procedure model for the transformation. In doing so, the consistency between the respective enterprise models is checked and can thus be assured before implementing the transformation. If conflicts arise during the analysis step, this information should trigger adjustments of the to-be model. Such an information flow needs to be integrated by a comprehensive planning process.

The identified temporal dimensions modeling time and valid time require enterprise models to provide attributes that are able to capture timestamps. Hence, a prerequisite for enabling transformation of enterprise models is an adjustment of the respective meta models. While valid time should at least be captured by a *valid from* timestamp (as shown in Figure 1) modeling time

should be represented in the form of a version number in order to enhance readability for the model user. As current practices indicate that also the modeling of ideal “vision models” is required [1], the meta model should furthermore provide an additional attribute for marking this kind of special model.

Using a transformation model also demands an adaptation of the meta model. The transformation model must be able to capture information on successor relationships, the affected model elements in other enterprise models as well as their interrelations. The respective attributes need to be provided by the meta model capturing links to other model elements, for example. In order to identify the necessary transformation steps and establish the transformation procedure model, the support of an adequate modeling tool is recommended. Such a tool must be able to analyze and compare models and propose a useful sequence of transformation steps depending on the interrelations and interdependencies between the model elements. Moreover, the tool could support the task of representing temporal information as well as model numbers or label in a manageable way.

For the correct assessment of the affected elements that are part of the transformation, it seems inevitable to clear when two elements are regarded as identical to assess deviations between elements of different models. From a theoretical perspective, two model elements in two different models are identical if they disclose no changes with regard to the transformation which is to be performed. Consequently, it seems unreasonable to define that only elements having completely identical attribute values are regarded as identical. On the contrary, attributes that are regarded as relevant to the planning and transformation purposes and therefore for deviation analysis, should be defined situationally. For example, only the change of a release version of a used database management system software could be relevant for planning, so the comparison would only be performed upon this attribute.

The presented approach provides a concept for enabling transformation of enterprise models under certain limitations.

The case study illustrates the application of most of the elements proposed in this paper. It therefore demonstrates the feasibility of the presented approach. However, some information is not contained in additional models or visualizations but is represented as additional attributes to already established model elements. This especially holds true for the proposed usage of transformation models.

With regards to the highly complex systems that are subject to enterprise modeling, one will presumably come across multiple modeling levels, opposed to the two presented levels in this paper. Though, the pre-

sented macro and micro level are universally suitable to different modeling levels within enterprise modeling (and thus recursively applicable) because they are differentiated by their central subject of consideration: While on the macro level whole models and their temporal development are examined, the micro level focuses on the inner structure, i.e. on the model elements. However, the question on which modeling levels the macro and micro framework is applied, i.e. on which level the planning is conducted, can only be answered by situational decisions about the planning process and may require additional heuristics.

In order to show a general concept, only a generic type of model element is considered at the micro level. Although this type is not specified, the addition of model elements of different nature with different characteristics might result in a higher complexity because of additional interdependencies between the different element types. As stated above, one model element could itself consist of links to other models, for example. Such recursivity is not considered in the presented approach and should be examined and integrated in a further contribution.

As stated by Bradbury et al. [4] the removal and addition of connectors between architectural elements are two out of four possible modifications. The illustrated approach to plan the transformation from one enterprise model to another does not consider connectors, i.e. interrelations between model elements. The planning of transformation of relationships requires a formal description in order to ensure consistency and functionality in the planned model. A comprehensive procedure model for transformation planning will have to include this aspect.

Finally, as already stated in the introduction, the presented approach does not cover decisions creating to-be models or choosing a certain transformation path. Hence, this approach can be understood as the complementary concept to a model planning process.

6. Summary and Outlook

With regard to planning purposes, this paper presents a systematic approach to capture dynamic transformation in enterprise models. The approach distinguishes a macro and a micro level, making the transformation process more transparent. On a macro level we have proposed a framework for temporal dimensions that are relevant to represent model transformation. From this framework, valuable propositions for adapting enterprise meta models in order to enable transformation can be derived.

On a micro level, we propose a procedure model for analyzing differences between as-is and to-be mod-

els with the objective of deriving effective transformation operations. The proposed procedure enables a consistency check between as-is and to-be models as well as the realization of enterprise model transformation for planning purposes.

While enterprise models create transparency and support for organizational engineering, the transformation of enterprise models themselves also requires a structured and comprehensive planning method. Such a planning method must embrace representational aspects as well as procedural aspects. Additionally, dynamic aspects like unplanned shifts and their effects on interdependent enterprise models must be considered. Lessons from the field of system theories will probably provide useful input in this context and should be evaluated in further research activities.

The presented approach on the macro level forms a contribution to the subject of representation. The long-term goal is to establish an information model that is able to capture all planning relevant aspects while taking dynamic changes into account. This will require further research regarding meta model engineering for enterprise models and especially transformation models. The construction of transformation models will furthermore require an adequate modeling technique.

Procedural aspects that are concerned with a planning process itself, i.e. with involved roles and activities, will complement this approach. The presented approach for establishing a procedure model for transformation constitutes a first step towards the planning procedure model, targeting the core aspect of transformation planning.

Besides that, other aspects around the planning and transformation of enterprise models will be subject to further research. For example, strategic questions on where to start planning future states might be of interest. In this context, the idea of defining a “leading” model type or a “leading” model element seems reasonable.

7. References

- [1] S. Aier, B. Gleichauf, J. Saat, R. Winter, "Complexity Levels of Representing Dynamics in EA Planning", in: Proceedings of the 5th International Workshop on Cooperation & Interoperability – Architecture & Ontology (CIAO! 2009), Springer, Berlin, 2009, pp. 55–69.
- [2] S.W. Ambler, *The Object Primer: Agile Model-Driven Development with UML 2.0*, Cambridge University Press, 2004.
- [3] M. Arnoldi, K. Beck, M. Bieri, M. Lange (Time travel: A pattern language for values that change), <http://www.lifeware.ch/upload/DMS/TimeTravel.pdf>, 2008.
- [4] J.S. Bradbury, J.R. Cordy, J. Dingel, M. Wermelinger, "A Survey of Self-Management in Dynamic Software Architecture Specifications", in: Proceedings of WOISS, ACM, 2004, pp. 28–33.
- [5] S. Buckl, A. Ernst, F. Matthes, C.M. Schweda, "An Information Model for Landscape Management – Discussing Temporality Aspects", in: Proceedings of the 3rd Workshop on Trends in Enterprise Architecture Research (TEAR 2008), Sydney, 2008.
- [6] S. Buckl, A.M. Ernst, F. Matthes, C.M. Schweda, "An Information Model for Managed Application Landscape Evolution", *Journal of Enterprise Architecture*, Vol. 5, 2009, pp. 12–26.
- [7] A. Carlson, S. Estepp, M. Fowler, "Temporal Patterns", in: Proceedings of Pattern Languages of Program Design '98, Addison Wesley, Reading, Mass., 2000, pp. 241–262.
- [8] S. Cîmpan, H. Verjus, I. Alloui, "Dynamic Architecture Based Evolution of Enterprise Information Systems", in: Proceedings of the International Conference on Enterprise Information Systems (ICEIS'07), Madeira, Portugal, 2007, pp. 221–229.
- [9] J.L.G. Dietz, *Enterprise Ontology – Theory and Methodology*, Springer, Berlin, Heidelberg, 2006.
- [10] L.B. Eliot, *Information systems strategic planning*, Computer Technology Research Corporation, Charleston, SC, 1991.
- [11] M.S. Fox, M. Gruninger, "Enterprise Modeling", *AI Magazine*, Vol. 19, 1998, pp. 109–121.
- [12] U. Frank, "Perspective Enterprise Modeling (MEMO) – Conceptual Framework and Modeling Languages", in: Proceedings of the 35th Hawaii International Conference on System Sciences (HICSS-35), 2002, pp. 72–81.
- [13] IBM, *Business Systems Planning: Information Systems Planning Guide*, Vol. 1, IBM, Atlanta, 1984.
- [14] H. Jonkers, M. Lankhorst, R. van Buuren, S. Hoppenbrouwers, M. Bonsangue, L. van der Torre, "Concepts for Modelling Enterprise Architectures", *International Journal of Cooperative Information Systems*, Vol. 13, 2004, pp. 257–287.
- [15] H. Jonkers, M.M. Lankhorst, H.W.L. Doest, F. Arbab, H. Bosma, R.J. Wieringa, "Enterprise architecture: Management tool and blueprint for the organisation", *Information Systems Frontiers*, Vol. 8, 2006, pp. 63–66.
- [16] W. King, "Strategic Planning for Management Information Systems", *MIS Quarterly*, Vol. 2, 1978, pp. 27–37.

- [17] M. Kirikova, "Explanatory capability of enterprise models", *Data and Knowledge Engineering*, Vol. 33, 2000, pp. 119–136.
- [18] S. Kurpjuweit, R. Winter, "Concern-oriented Business Architecture Engineering", in: *Proceedings of the 24th Annual ACM Symposium on Applied Computing (SAC)*, Honolulu, Hawaii, 2009, pp. 265–272.
- [19] R. Lagerström, J. Saat, U. Franke, S. Aier, M. Ekstedt, "Enterprise Meta Modeling Methods – Combining a Stakeholder-Oriented and a Causality-Based Approach", in: *Enterprise, Business-Process and Information Systems Modeling, Proceedings of the 10th International Workshop BPMDS and 14th International Conference EMMSAD at CAiSE 2009*, Vol. LNBIP 29, Springer, Berlin, 2009.
- [20] M. Lankhorst, *Enterprise Architecture at Work: Modeling, Communication and Analysis*, Springer, Berlin et al., 2005.
- [21] K.D. Niemann, *From Enterprise Architecture to IT Governance. Elements of Effective IT Management*, Vieweg, Wiesbaden, 2006.
- [22] E. Niemi, "Enterprise Architecture Stakeholders – A holistic view", in: *Proceedings of the 13th Americas Conference on Information Systems (AMCIS 2007)*, Keystone, CO, 2007.
- [23] M. Op 't Land, E. Proper, M. Waage, J. Cloo, C. Steghuis, *Enterprise Architecture – Creating Value by Informed Governance*, Springer, Berlin, 2009.
- [24] M. Pulkkinen, "Systemic Management of Architectural Decisions in Enterprise Architecture Planning. Four Dimensions and Three Abstraction Levels", in: *Proceedings of the 39th Annual Hawaii International Conference on System Sciences (HICSS-39)*, Vol. 8, IEEE Computer Society, Honolulu, Hawaii, 2006, pp. 179a (171–179).
- [25] M. Pulkkinen, A. Hirvonen, "EA Planning, Development and Management Process for Agile Enterprise Development", in: *Proceedings of the 38th Annual Hawaii International Conference on Systems Sciences (HICSS-38)*, IEEE Computer Society, Los Alamitos, CA, USA, 2005, pp. 223.223 (221–210).
- [26] M. Rohloff, "Framework and Reference for Architecture Design", in: *Proceedings of the 14th Americas Conference on Information Systems (AMCIS 2008)*, Toronto, 2008, pp. 1–14.
- [27] R.T. Snodgrass, *Developing Time-Oriented Database Applications in SQL*, Morgan Kaufmann Publishers, 2000.
- [28] P. Sousa, J. Lima, A. Sampaio, C. Pereira, "An Approach for Creating and Managing Enterprise Blueprints: A Case for IT Blueprints", in: *Advances in Enterprise Engineering III, Proceedings of the 5th International Workshop CIAO! 2009 and 5th International Workshop EOMAS 2009*, Springer, Berlin, 2009, pp. 70–84.
- [29] S.H. Spewak, S.C. Hill, *Enterprise Architecture Planning – Developing a Blueprint for Data, Applications and Technology*, John Wiley & Sons, New York, 1993.
- [30] S.H. Spewak, M. Tiemann, "Updating the Enterprise Architecture Planning Model", *Journal of Enterprise Architecture*, Vol. 2, 2006, pp. 11–19.
- [31] A. Targowski, *The architecture and planning of enterprise-wide information management systems*, Idea Group Publishing, Harrisburg, 1990.
- [32] D.F. Tyler, T.P. Cathcart, "A structured Method for Developing Agile Enterprise Architectures", in: *Proceedings of the International Conference on Agile Manufacturing (ICAM 2006)*, Norfolk, Virginia, USA, 2006, pp. 1–8.
- [33] F. Vernadat, *Enterprise Modeling and Integration – Principles and Applications*, Chapman & Hall, 1996.
- [34] R. Winter, T. Bucher, R. Fischer, S. Kurpjuweit, "Analysis and Application Scenarios of Enterprise Architecture – An Exploratory Study", *Journal of Enterprise Architecture*, Vol. 3, 2007, pp. 33–43.
- [35] R. Winter, R. Fischer, "Essential Layers, Artifacts, and Dependencies of Enterprise Architecture", *Journal of Enterprise Architecture*, Vol. 3, 2007, pp. 7–18.
- [36] T. Ylimäki, "Potential Critical Success Factors for Enterprise Architecture", *Journal of Enterprise Architecture*, Vol. 2, 2006, pp. 29–40.