DSMLs for Enterprise Architecture Management

Review of Selected Approaches

Heiko Kattenstroth
Information Systems and Enterprise Modeling Research Group
Institute for Computer Science and Business Information Systems (ICB)
University of Duisburg-Essen, Germany
heiko.kattenstroth@uni-due.de

ABSTRACT

Management of today's IT is a challenging task that requires a profound understanding of both the IT landscape and the relevant business context. In this regard, enterprise architecture management (EAM) aims at structuring an enterprise by providing purposeful abstractions of both IT and the surrounding action systems and thereby takes into account the relevant business context. For this purpose, it offers graphical modeling languages that provide stakeholders with specific and illustrative views on a company at various organizational levels, such as on value chains, business processes, or IT landscapes. Accordingly, the modeling languages are an essential part of EAM. The development of domain-specific modeling languages (DSML) has gained remarkable attention. They promise to increase the speed and ease of software development, prevent the construction of nonsensical models, improve modeling productivity as well as model quality.

In this paper, we explore to which extent current approaches for EAM utilize domain-specific modeling and how requirements from the very core of EAM are fulfilled by corresponding DSMLs. In this respect, an extensive analysis of modeling requirements in the context of EAM serves to assess existing approaches. Finally, we identify future research topics that should lead to a more sophisticated use of DSMLs for EAM and foster a closer collaboration between both communities.

Keywords

Domain-Specific Modeling Language, Enterprise Modeling, Enterprise Architecture Management, IT-Management

1. INTRODUCTION

Today's IT organizations are confronted with remarkable challenges: They have to deal with the tremendous *complexity* of present day enterprises and their IT. A complexity that results from the multitude of IT platforms, networks, and information systems as well as their interrelations. At the same time they are expected to efficiently *support the business* and to drive innovations. Furthermore, aligning the IT to the business demands the participation of *various stakeholders* from business and IT, such as executives, end-users, and IT experts. Their different backgrounds, perspectives on IT and business, and technical languages result in language barriers. These barriers hamper *communication* and collaboration between the stakeholders and consequently compromise the efficiency of the IT organization.

To cope with these challenges, methods are required that purposefully reduce the complexity inherent to enterprises – with regard to both business and IT –, facilitate communication among groups of stakeholders, and support IT management as well as the development of large scale information systems. In this respect, approaches from the field of Enterprise Architecture Management (EAM) and Enterprise Modeling (EM) such as TOGAF [26], ArchiMate [27], and MEMO [10], have gained remarkable attention over the last years. They seem to provide a promising foundation for several reasons: (1) They serve to structure an enterprise by providing purposeful abstractions of IT and the surrounding action systems, often arranged on different layers and/or perspectives (cf. Figure 1). (2) They provide stakeholders with specific and illustrative views on a company at various organizational levels, such as on value chains, business processes, or IT landscapes [12]. (3) They usually build upon meta models that include organizational and technical context, such as for goal, business process or organizational structure modeling on the one hand and for modeling the IT organization on the other [11].

Domain specific languages (DSLs) and domain specific modeling languages (DSMLs) primarily originate from the field of software engineering [29]. The reconstruction of a domain and its language as a *modeling language* (e.g. a meta model) provides users with concepts they are familiar with. Accordingly, a DSML is intended to provide concepts on a higher level of domain-specific semantics. This allows for a consistent, intuitive and semantically rich modeling [17] and at the same time promotes modeling productivity and contributes to model quality as implicit integrity constraints prevent the construction of nonsensical models. At the same time, a DSML will usually include a mapping of their concepts to those of programming languages, thus enabling code generation. As the modeling languages and the generators are build or adapted for their particular application scenario (i.e. their specific domain), they result in higher productivity and quality as well as reduced costs [16, 17].

The modeling concepts and modeling languages used to support EAM bear resemblances to DSMLs (e.g., reconstructions of key concepts of a domain). Some approaches even refer to their modeling languages as DSMLs. However, the primary focus of EAM is on documentation and supporting various analysis – especially using high-level abstractions to address upper management. In this respect, it remains unclear, for example, to what extend the created models are used to generate code or, more general, how the modeling languages used in the context of EAM can be compared to those proposed by the DSML community. Considering the effort that is required to create models of an enterprise and the amount of knowledge they capture as well as the advantages of DSMLs it seems worthwhile to investigate whether DSMLs can be applied more thoroughly to support EAM and the challenges it has to face. Against this background we investigate the current state-of-the-art in using DSMLs for EAM and identify future research topics. The results of our analysis are intended as foundation for discussion with and discursive evaluation by peers and domain experts. Amongst others, the results should foster collaboration between both research communities.

In the next section we present a short introduction to EAM and develop a conceptual framework to assess the different approaches for EAM and the DSMLs they propose. For this purpose, we review current literature to identify requirements from the EAM domain. In Section 3 we discuss related work. Selected approaches are reviewed in Section 4. In Section 5, we identify future research topics. Section 6 presents concluding remarks.

2. EA MANAGEMENT: OBJECTIVES & REQUIREMENTS

Objectives: Current business practice requires an aligned and integrated approach to management of business and IT. Achieving alignment and integration as well as managing change depend on a holistic approach that provides appropriate abstractions for both business and IT [18]. Against this background, introduction and use of an *enterprise architecture* (EA) have gained remarkable attention from academia and practice. The architecture of an enterprise depicts the "fundamental organization of a system, embodied in its components, their relationships to each other and the environment" [15]. Organizational aspects such as developing an enterprise architecture as well as defining and enforcing standards, rules, and guidelines are part of *enterprise architecture management* (EAM). As depicted in Figure 1, an EA can be divided into *layers* that describe different elements of an enterprise (see [31]). To guide the evolution of the enterprise, the as-is state describing the status quo, at least one envisioned to-be state and several intermediary states are modeled. Accordingly, *projects*, which often affect the enterprise on more than one layer, transform the enterprise and its architecture from the as-is state to the envisioned to-be state (see Δ in Fig. 1). Although the various approaches for EAM emphasize different management aspects and suggest different modeling approach for these aspects, they nevertheless agree that it is necessary to provide a holistic view on an enterprise, which accounts for aspects from all layers, ranging from business to IT [6]. Besides documenting an enterprise and guiding its evolution, EAM supports further scenarios, e.g., achieving consolidation and integration after mergers & acquisitions [2].

Requirements: For the purpose of this paper, we divided the requirements into two groups: (i) basic requirements and (ii) requirements that are specific for the modeling languages and their use. The former requirements are summarized in Table 1; the later are discussed below.

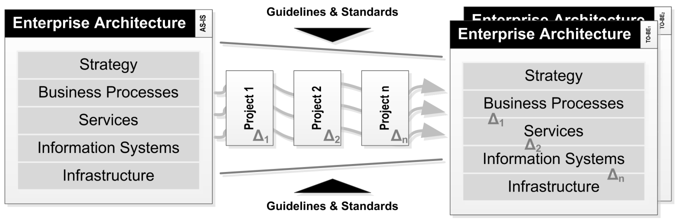

Figure 1: Enterprise (Architecture) Evolution (based on [9])

ID	*Basic Requirement (BR)*
BR 1	Manage transformation projects
BR 2	Support flexible visualizations
BR 3	Enable cross-disciplinary analyses
BR 4	Support for architecture governance
BR 5	Account for different perspectives

Table 1: Summary of Basic Requirements

The relevance of information systems for an enterprise and their potential for improving its competitiveness is unquestioned. However, the development, introduction, and maintenance of information systems remain a substantial challenge for many firms. At the same time, creating an enterprise architecture, keeping it up to date and promoting accompanying management structures also requires major effort that comes at high costs and implies significant risks.

SR 1 – Model usage: An approach for EAM should foster a widespread use of the models, from development to maintenance and management of information systems. Therefore, it should include concepts at different organizational levels that can also be mapped to implementation-level concepts according to clear transformation rules. Code generators and transformations should facilitate and improve the development of information systems [17]. In this respect, an approach for EAM needs to balance the need for fine grained models that allow for code generation and the effort to create such models (i.e. practicability vs. operationalization).

The problem of aligning business and IT requires an EA to include not only IT related artifacts but also strategies, goals, organizational roles, business processes, and further aspects of an enterprise. In this respect, it is widely accepted to separate an enterprise architecture into different layers according to the fundamental structure of an enterprise [31]. However, for these domains different modeling languages, tools, visualizations, and even different objectives as well as stakeholders exist.

SR 2 – Integration & reuse: The separation into different modeling languages is to a certain degree artificial as it should be possible to merge them into one modeling language using one large meta model. However, one has to account for the situation in practice where the different domains are not approached in an integrated way. It seems not feasible to force an enterprise to start with an all-encompassing approach from scratch. Rather, an approach for EAM should account for already existing techniques (e.g., modeling languages, models, and tools for business processes) as well as foster their integration and reuse at an appropriate level of abstraction [18].

The complexity of today's organizations, which is among others caused by the large number of information systems and their various relationships, result in extensive architectural descriptions. These descriptions contain elements on different levels of detail to fulfill the diverse needs of the stakeholders. This results in models that might overburden users with too much detail and leads to the challenge to maintain the consistency of the EA.

SR 3 – Adaptable complexity: The complexity of the model editor and corresponding models should be adaptable to the different professional backgrounds and interests of the intended user as well as the to particular application scenario. Accordingly, an approach for EAM should provide language concepts that allow for hiding details such as relations, model elements, and attributes not needed in a particular situation.

On the one hand, the needs and interests of enterprises with regard to enterprise architecture management differ. Neither a fixed starting point nor fixed modeling languages or management structures exist that can be applied without changes to a large variety of enterprise [6]. This results in the requirement to support *organization-specific adaptations*. On the other hand, organizations are constantly forced to change, for example due to changing markets or technological innovations. Any change that is made in an enterprise has to be aligned with the enterprise architecture. Most changes can be handled on the model level (e.g., by modifying reference models). However, some changes require modifications of the underlying modeling languages. Therefore, existing models and code generators are likely to become invalid. This leads to the two following requirements.

SR 4 – Enterprise-specific adaptations: An approach to EAM should support the safe and convenient adaptation of the modeling language, the management structures, and the tool along with its visualizations, transformations and code generators. Convenient refers to the effort it takes to realize a particular adaptation to organization-specific needs. Safe refers to support for the update of existing artifacts such as models, analysis, and code generators to maintain integrity of the EA.

SR 5 – Meta model evolution: An approach for EAM should support the coordinated and controlled evolution of meta models. Accordingly, corresponding models need to evolve as well (often referred to as "co-evolution") to remain compliant to the meta model and not become eventually invalid. For this purpose, the approach should account for existing techniques for (meta) model evolution (e.g., [22]).

Development and management of information systems requires particular information about instance, for example of business processes or IT resources (i.e., instance level data). There is a wide range of tools that aim at preparing and presenting these values, e.g., from data warehouse to monitoring and reporting tools. However, they usually do not support users in interpretation and assessment of the presented values. Associating such values with the corresponding conceptual level to add information about the (business) context contributes to a more anticipatory and prudent management. Such an integration of models with corresponding instance information, i.e., the use of models at run-time, fosters a more profound decision-making and a larger variety of analyses than analyzing at instance level only.

SR 6 – Integration with instance data: An approach for EAM should account for the integration of models and modeling tools with systems that manage corresponding instance level data (or integrate a corresponding component). It should be possible to navigate from the instance level to the conceptual level – and vice versa. Moreover, information about instance should be used to enrich code generators and transformations, e.g., to generate access control lists to support IT security management.

3. RELATED WORK

Several reviews of the state-of-the-art in enterprise architecture management have been published. A first analysis by Urbaczewski & Mrdalj [28] focuses on a comparison of EAM frameworks. They propose a method that should guide the selection of an framework based on specific stakeholders needs. However, the analysis as well as the proposed method remain on a high level of abstraction and do not account for modeling aspects. A more elaborate review is presented by Aier et al. [1]. They analyze the state-of-the-art in enterprise architecture management as described in literature as well as carried out in business practice. In this respect, they conduct a literature review and present current practices based on results of an empirical study. The main focus is on management aspects of EAM. However, they assess different approaches for EAM with regard to their modeling languages and their coverage of the different layers and elements of an enterprise (see Fig. 1) without going into details. Winter et al. [30] make use of a similar approach. They review existing literature in the context of EAM management to assess the different characteristics and shortcomings of corresponding approaches. Based on these findings, they investigate the state-of-the-art in practice using an online survey. This study is not concerned with modeling aspects or modeling languages. Solely focusing on academic publications in the area of EAM, Mykhashchuk et al. [21] chart the landscape of EAM research in terms of research communities and citations, whereas Lucke et al. [19] analyzes challenges and open issues to guide future research. Both reviews remain on a rather superficial level with regard to modeling aspects of EAM and do not discuss modeling languages. An appropriate tool is necessary to create, evolve, and maintain an enterprise architecture. Accordingly, tools are essential for the success and the acceptance of EAM. Matthes et al. [24, 20] analyze and assess commercial tools for EAM along with a brief presentation of typical task such tools should support. As most of the tools allow for adaptations of the meta model and the study aims at an analysis of the market for EAM tools, modeling aspects are not considered in detail.

4. REVIEW OF SELECTED APPROACHES

Based on the requirements analysis from Section 2, this section provides an assessment of selected approaches for EAM.

Requirement (SRx)	*TOGAF*	*Archimate*	*MEMO*
Model usage (1)	supports primarily management through high level analyses, can be used for more specific scenarios (e.g., in the context of SOA), no code generation	supports different organizational levels through views, no code generation	scope shifted from software development to other scenarios: e.g., support IT-management [14], generate code for security management, uses models as a front end for dashboards [13]
Integration & reuse (2)	core metamodel can be extended, does not provide any guidance how to integrate existing models	explicitly accounts for integration of other modeling languages – in most cases more detailed or more specific languages	flexible language architecture allows for extensions, well described integration between different modeling languages through shared concepts, comprises meta modeling language
Adaptable complexity (3)	adaptable complexity supported by flexible views, not directly supported via modeling constructs	provides flexible, stakeholder-specific views (*viewpoints*), not considered directly in modeling language	partially supports decomposition, provides concepts such as `information system` as an abstraction over hardware and software, flexible visualizations
Enterprise-specific adaptations (4)	a-priori adaptation of management guidelines (ADM), does not explicitly account for adaption of modeling aspects	meta model can be customized, allows for extensions	supports *method engineering* for tailoring of modeling methods, existing meta models rather static
Meta model evolution (5)	not addressed	not addressed	not addressed
Integration with instance data (6)	not addressed	not addressed	instance data used to enrich models (*models@run-time*), not used for code generation

Table 2: Summary of Assessment

For the framework selection, we included existing reviews [28, 1, 30, 21, 23] and searched for respective publications. We reduced the list as we rather focus on modeling aspects (i.e., specific requirements) such as modeling languages than on management aspects. Accordingly, we examine popular approaches for EAM that provide – at least partial – specifications for their modeling languages. We make an exception for the *Zachman Framework* that is presented due to its historical role.

One of the first frameworks was presented by *Zachman* [32]. Inspired by an architect's paradigm and the use of building plans, he wanted to present customers and other stakeholders with comprehensible representations of an information systems architecture. He proposed a two dimensional matrix that distinguishes five modeling layers (*scope*, *business*, *information systems*, *technology*, and *detailed representations*) and six dimensions (based on the central questions of *what*, *how*, *where*, *who*, *when*, and *why*), with each cell representing a particular perspective of an organization. Zachman gives a few examples for modeling perspectives but the framework lacks comprehensive language specifications.

TOGAF is one of the most prominent EAM frameworks [26]. It is developed by "The Open Group", an international consortium that comprises large software vendors and user organizations. An essential component of TOGAF is the *Architecture Development Method* (ADM). A method, consisting of eight phases, that supports and guides the development and maintenance of enterprise architectures. The *content metamodel* provides a basic *core* meta model and a number of *extensions* to address specific issues in more detail, e.g., process modeling. The changes that are required to use an extension are described briefly. A set of architecture diagrams is part of TOGAF. Moreover, it proposes to distinguish between *business*, *data*, *application*, and *technology architectures* and is supported by major EAM tools. The focus of TOGAF is on developing an EA management function using ADM, whereas it fosters individual adaptations.

Developed in a dutch research project and later transferred to The Open Group, *ArchiMate* supplements TOGAF with a modeling language for EA [18, 27]. Archimate differentiates between three layers(*Business*, *Application*, and *Technology Layer*) and three aspects (*Passive Structure*, *Behavior*, and *Active Structure*). To keep the language understandable, it has been explicitly designed to be as small as possible and to include extensions (e.g., to support implementation and migration of architectures). Furthermore, it allows for custom modifications ("profiles") through adding attributes to concepts and relationships as well as specialization of concepts. Archimate is accompanied by a set of related methods and guidelines. Moreover, it supports the flexible composition of views on the EA ("viewpoints") and provides a graphical notation.

The method for *Multi-Perspective Enterprise Modeling*(MEMO) originates primarily from academic research [10, 12]. It is based on a high-level conceptual framework that represents a "ball park view" on an enterprise. In this respect, it is composed of three generic perspectives (*strategy*, *organization*, *information system*) each of which can be further detailed into various aspects (*resource*, *structure*, *process*, *goal*). To allow for more elaborate analyses, each selected perspective is associated with a set of diagram types, which are realized through DSMLs. All modeling languages (e.g., ITML for IT landscapes, OrgML for business processes) are specified using the MEMO Meta Modeling Language (MML, [11]). This fosters their integration since they are specified using the same modeling concepts. This consequently leads to integrated models at type level, e.g., integrated IT and business process models. Dedicated graphical notations that correspond to concepts of the particular domains are part of every modeling language. To allow for adaptation of the modeling languages, MEMO provides support for *Method Engineering* (see, e.g., [3]).

The results of the assessment of EAM frameworks against the modeling-specific requirements are summarized in Table 2. In addition to larger frameworks we also considered smaller contributions. Some authors emphasize the advantages of DSMLs and suggest to enrich existing approaches for EAM accordingly. Chiprianov et al. [7, 8], for example, argue that existing EA modeling approaches are inadequate at the technical level. Therefore, they proposes extensions for the telecommunication sector. Silingias & Butleris [25] are concerned with tools for EAM. They argue that the lack of standards for tools based on DSMLs make them unsuitable for EAM. Instead, they propose to use UML profiles to customize UML tools. Although they do not vote against DSMLs in general, their proposal can be questioned – especially with regard to recent surveys (e.g. [20]), which present mature and sophisticated tools. A different perspective is presented by Buckl et al. [5]. They question whether it is feasible to define a meta model (i.e. a modeling language) for EAM that could gain broad acceptance at all. Therefore, they propose an approach to construct an organization-specific “information model” based on patterns [4].

5. CONCLUSIONS & OUTLOOK

This paper investigates to which extent current approaches for EAM utilize domain-specific modeling and how these DSMLs fulfill corresponding modeling requirements. In this respect, our analysis shows that most existing approaches for EAM make only limited use of DSMLs and a number of open issues remain:

- The profound use of an EA for further scenarios (e.g., code generation) requires an analysis of resulting requirements and the development of corresponding solutions – including a modeling tool.
- To foster reuse of existing models as well as modeling methods an integration approach is needed. This requires a flexible language architecture and should be based on a meta modeling approach, for instance, as provided by MEMO [11]. Moreover, the modeling concepts currently provided by frameworks such as TOGAF and Archimate – compared with domain-specific modeling languages as such – remain on a rather generic level (see 2 in Figure 2). To allow for code generation, more detailed concepts are necessary.
- To account for different enterprises and their specific needs there is need for flexibility and adaptability. Based on their priorities an approach for EAM should allow for its customization (e.g., modeling language, diagrams) – without loosing its main advantages: The accurate reconstruction of key concepts of a certain domain. This requires corresponding guidelines and expertise.
- The need to provide stakeholder-specific views on an EA is widely accepted and supported by EAM approaches. However, there is a lack of concepts that allow to adapt complexity of the provided modeling concepts (e.g., derive transitive closure of relationships to hide certain elements, provide user-specific abstractions).
- The support for the evolution of modeling languages and their corresponding meta models, which also includes the evolution of corresponding code generators and existing models, is an open issues for both communities.

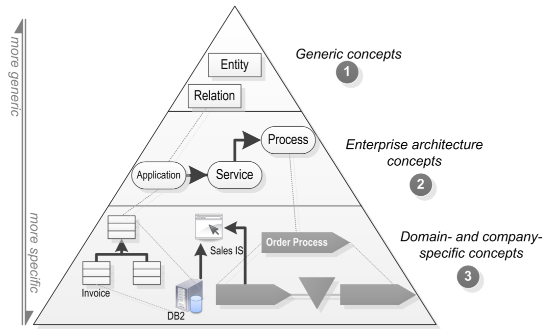

Figure 2: Concepts at different levels of specificity (based on[27])

Existing work on EAM in most cases neither provides solutions nor discusses these challenges in detail. Against this background, it seems promising to foster a closer collaboration between both research communities. Although, we could find only few sources, in which DSMLs (or DSLs) are mentioned explicitly or even considered as an essential part of EAM, the importance of appropriate and comprehensive modeling languages for the success of EAM in general is undisputed. With regard to the terminology, we come to similar conclusions as Schoenherr [23] who, based on a literature review, criticizes the lack of a common terminology in the context of enterprise architecture management. The terms “modeling language”, “domain-specific modeling language”, “domain-specific language”, and “architecture description language” are used inconsistently. In a similar way, the terms “enterprise architecture”, “enterprise architecture management”, “enterprise modeling”, “enterprise engineering”, and “enterprise architecting” are mostly used without clear differentiations. Especially with regard to the lack of a consistent terminology in the DSML community (DSL vs. DSML vs. DSPL, graphical vs. non-graphical modeling, etc.) and to foster a closer collaboration of the DSML and EAM communities, the development of a precise terminology is essential. Due to the nature of this paper, we had to limit the number of approaches and requirements for our analysis. For a more comprehensive review of the state of the art, we will extend the requirements analysis and conduct a more detailed comparison of modeling languages for EAM.

6. REFERENCES

[1] S. Aier, C. Riege, and R. Winter. Unternehmensarchitektur - Literaturüberblick und Stand der Praxis. *WIRTSCHAFTSINFORMATIK*, 50(4):292–304, 2008.

[2] M. Alaranta and S. Henningsson. An approach to analyzing and planning post-merger IS integration: Insights from two field studies. *Information Systems Frontiers*, 10(3):307–319, 2008.

[3] S. Brinkkemper. Method engineering: engineering of information systems development methods and tools: Method Engineering and Meta-Modelling. *Information*

and Software Technology, 38(4):275–280, 1996.

[4] S. Buckl, A. M. Ernst, J. Lankes, and F. Matthes. Enterprise Architecture Management Pattern Catalog. Technical report, TU Munich, Germany, 2008.

[5] 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 *Proc. of the WI 2007*, 2007.

[6] S. Buckl, F. Matthes, and C. M. Schweda. Towards a Method Framework for Enterprise Architecture Management - A Literature Analysis from a Viable System Perspective. In *Proc. of the 5th Int. Workshop on Business/IT Alignment and Interoperability (BUSITAL 2010)*, pages 46–60, 2010.

[7] V. Chiprianov, I. Alloush, Y. Kermarrec, S. Rouvrais, et al. Telecommunications Service Creation: Towards Extensions for Enterprise Architecture Modeling Languages. In *6th Intl Conf. on Software and Data Technologies (ICSOFT)*, pages 23–28, 2011.

[8] V. Chiprianov, Y. Kermarrec, and S. Rouvrais. Extending Enterprise Architecture Modeling Languages: Application to Telecommunications Service Creation. In *Proc. of the 27th Symposium on Applied Computing (SAC)*, 2012.

[9] G. Dern. *Integrationsmanagement in der Unternehmens-IT: Systemtheoretisch fundierte Empfehlungen zur Gestaltung von IT-Landschaft und IT-Organisation.* Vieweg+Teubner, 2011.

[10] U. Frank. Multi-Perspective Enterprise Modeling (MEMO): Conceptual Framework and Modeling Languages. In *Proc. of the 35th Hawaii Int. Conf. on System Sciences (HICSS-35)*, 2002.

[11] U. Frank. The MEMO Meta Modelling Language (MML) and Language Architecture. ICB Research Report 43, Universität Duisburg-Essen, 2011.

[12] U. Frank. Multi-Perspective Enterprise Modeling: Foundational Concepts, Prospects and Future Research Challenges (accepted for publication). *Software & Systems Modeling*, 2012.

[13] U. Frank, D. Heise, and H. Kattenstroth. Use of a Domain Specific Modeling Language for Realizing Versatile Dashboards. In *Proc. of the 9th OOPSLA workshop on domain-specific modeling (DSM '09)*, 2009.

[14] U. Frank, D. Heise, H. Kattenstroth, D. Ferguson, E. Hadar, and M. Waschke. ITML: A Domain-Specific Modeling Language for Supporting Business Driven IT Management. In *Proc. of the 9th OOPSLA workshop on domain-specific modeling (DSM '09)*, 2009.

[15] International Organization for Standardization (ISO). Systems and software engineering - Recommended practice for architectural description of softwareintensive systems. ISO/IEC 42010:2007.

[16] J. Kärnä, J.-P. Tolvanen, and S. Kelly. Evaluating the Use of Domain-Specific Modeling in Practice. In *Proc. of the 9th OOPSLA workshop on domain-specific modeling (DSM '09)*, 2009.

[17] S. Kelly and J.-P. Tolvanen. *Domain-specific modeling: Enabling full code generation.* Wiley-Interscience and IEEE Computer Society, 2008.

[18] M. M. Lankhorst. *Enterprise architecture at work: Modelling, communiation and analysis.* Springer, 2005.

[19] C. Lucke, S. Krell, and U. Lechner. Critical Issues in Enterprise Architecting – A Literature Review. In *Proc. of the 6th Americas Conf. on IS*, 2010.

[20] F. Matthes, S. Buckl, C. M. Schweda, and J. Leitel. Enterprise Architecture Management Tool Survey 2008. Technical report, TU Munich, Germany, 2008.

[21] M. Mykhashchuk, S. Buckl, T. Dierl, and C. M. Schweda. Charting the landscape of enterprise architecture management - An extensive literature analysis. In *Proc. of the 10th Int. Conf. on Wirtschaftsinformatik*, 2011.

[22] L. Rose, R. Paige, D. Kolovos, and F. A. Polack. An analysis of approaches to model migration. In *Proc. Joint MoDSE-MCCM Workshop*, pages 6–15, 2009.

[23] M. Schönherr. Towards a Common Terminology in the Discipline of Enterprise Architecture. In *Service-Oriented Computing – ICSOC 2008 Workshops*, Lecture Notes in Computer Science, pages 400–413. Springer, 2009.

[24] Sebis. Enterprise Architecture Management Tool Survey 2005. Technical report, TU Munich, Germany.

[25] D. Silingas and R. Butleris. Towards Customizing UML Tools for Enterprise Architecture Modeling. In *Proc. of the IADIS Int. Conf. IS*, 2009.

[26] The Open Group. *TOGAF Version 9: A Manual (TOGAF Series).* Van Haren Publishing, Zaltbommel, 9 edition, 2009.

[27] The Open Group. *ArchiMate® 2.0 Specification: Open Group Standard.* Van Haren Publishing, Zaltbommel, 1 edition, 2012.

[28] L. Urbaczewski and S. Mrdalj. A comparison of enterprise architecture frameworks. *Issues in Information Systems*, 7(2):18–23, 2006.

[29] A. van Deursen, P. Klint, and J. Visser. Domain-Specific Languages: An Annotated Bibliography. *ACM SIGPLAN Notices*, 35(6):26–36, 2000.

[30] K. Winter, S. Buckl, F. Matthes, and C. M. Schweda. Investigating the state-of-the-art in enterprise architecture management method in literature and practice. In *Proc. of the 5th Mediterranean Conf. on Information Systems (MCIS '10)*, 2010.

[31] R. Winter and R. Fischer. Essential Essential Layers, Artifacts, and Dependencies of Enterprise Architecture. *Journal of Enterprise Architecture*, 3(2):7–18, 2007.

[32] J. A. Zachman. A framework for information systems architecture. *IBM Systems Journal*, 26(3):276–292, 1987.