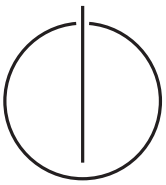
Technische Universität München

Fakultät für Informatik

Framework for the strategic planning of enterprise architectures

Master's Thesis in Wirtschaftsinformatik

Philip Achenbach



Technische Universität München

Fakultät für Informatik

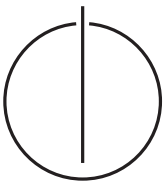
Framework for the strategic planning of enterprise architectures

Framework für die strategische Planung von Unternehmensarchitekturen

Master's Thesis in Wirtschaftsinformatik

Philip Achenbach

Supervisor:	Prof. Dr. rer. nat. Florian Matthes
Advisor:	Dipl.-Inform. Univ. Ivan Monahov
Submission date:	January 15, 2013



Declaration

I assure the single handed composition of this master's thesis only supported by declared resources.

Munich, January 15, 2013,

. . . . . .

Abstract

Modern enterprises are not only highly interwoven systems, consisting of numerous technical and human components, but are also facing an increasing number of regulatory, technical, and economic changes. In this highly dynamic environment, enterprises are forced to continuously adapt the way they do business, and the kind of business they do. These transformations however need considerable planning efforts to succeed, and hence benefit from an overarching perspective on the enterprise — spanning business and IT. This perspective is taken by enterprise architecture (EA) management to coordinate the transformation efforts and to provide an understanding of the current and the target state. This management function employs models as the foundation for planning activities, and as a means for the documentation of the architectural state.

Existing EA management approaches employ fairly distinct perspectives regarding the enterprise transformation planning. Furthermore, the topic of EA modeling is only partially covered in the present frameworks. This work therefore seeks to establish a unified perspective on enterprise transformation planning and proposes a framework for the strategic planning of the EA. As a foundation for this overall objective, a mapping of different terminologies found in state-of-the-art EA management approaches is contributed. This foundation helps to describe the common concepts and their relationships as requirements for a framework targeting the strategic EA planning. As a further contribution of this work, a planning-framework is presented, structuring the common concepts as to their application in an EA planning endeavor. This framework and the underlying requirements have furthermore been discussed in expert interviews and evaluated in an online survey. Overall, this work provides a foundation for the development of modeling techniques and can act as a basis for a possible tool support of strategic enterprise architecture planning.

Acknowledgements

This thesis would not have been possible without the ideas, friendship, support, and assistance of numerous people. In the following lines, I would like to take the opportunity to express my gratitude to the persons that supported me most.

First and foremost, I would like to thank my supervisor Prof. Dr. Florian Matthes for his support and the valuable feedback regarding diverse aspects of my research.

I would like to thank the iteratec GmbH for their support in conducting this research, and especially the managing director, Inge Hanschke, for her interest in a research cooperation and the discussions of the research results.

It is a pleasure to thank my academic advisor, Ivan Monahov, for his scientific guidance, the personal atmosphere, and the constant support until far into the night.

I owe my deepest gratitude to Dr. Christian M. Schweda, my advisor at iteratec, for his invaluable specialist input, the candid discussions, and the short-term reviews.

Special thanks to the unnamed experts, I was given the opportunity to interview, as well as to the anonymous participants of the online survey, each for their time and valuable input.

I am indebted to my colleagues at iteratec, for the kind and productive working environment and their continuous interest in my work.

I would like to thank my friends in my personal environment for their patience as well as for ensuring that I take a rest sometimes.

Last, but most of all, I want to thank my family for teaching me to become what I am, and for their love, encouragement, and trust throughout all these years.

Thank you.

Contents

List of Figures

List of Tables

1. Introduction

In the last decade, the landscape for business and *information technology* (IT) has changed extensively. The widespread availability of information technology and the continuous Internet-connectivity, granted by the domestic and mobile service providers, resulted in new standards for the interconnection of organizations and customers. Consequently, transaction costs across organization boundaries have been driven down and made the traditional organizational structures less attractive, both from a strategic and an economical point of view. The rapidly changing regulatory, technical, and economic environment furthermore forced the organizations to continuously rethink the ways in which they do business, as well as the type of business they do [Vas+01]. The flexibility to adapt to changes as well as to quickly implement new business capabilities became both vitally important for organizations, regardless of their size and type. On the other hand, the highly interwoven architecture of organizations and the increased need for information systems led to a remarkable expansion of IT-complexity. To exemplify this complexity, the *Credit Suisse*, a Swiss financial service provider, has over 2 800 business application systems, consisting of more than 100 000 000 lines of code, and distributed over more than 100 sites [MWF08].

The need to adapt to a changing environment on the one hand, and the dependency of IT-systems together with the involved complexity on the other hand, results in serious difficulties faced by the organizations. This difficulty repeatedly becomes aware in the occurrence of major complications. Only recently, the *UniCredit Bank Austria* was confronted with such a complication in a switch-over of IT systems, which resulted in major news coverage and a loss of 21 million euro[1][2].

Emerging paradigms, such as *service oriented architectures* (SOA), claim to be helpful in the reduction of this complexity. Their implementation in an organization however appears to be a difficult task. The restructuring of an IT landscape towards an implementation of SOA is a long-lasting endeavor, needing not only quite a few information about the applications, but also on the related business processes and business objects. This information is mostly not available at the beginning of a transformation project and needs to be gathered in an extensive process. Even if this information has been gathered, the transformation project can not assume the information to be endlessly valid. The transformation project therefore has to

[1] URL: http://derstandard.at/1350260586771/

[2] URL: http://wien.orf.at/news/stories/2564791/

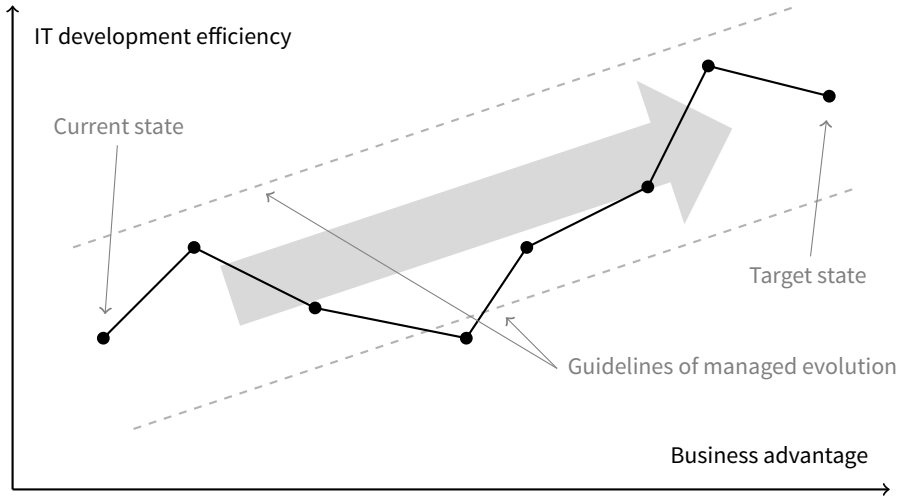


Figure 1.1.: Managed evolution of an EA [MWF08]

be continuously realigned with the steady changing environment. This need for a *business/IT alignment* goes beyond a mere *provider*-role of the IT, in which IT resorts itself to fulfill the business requirements. Instead, IT becomes an *enabler* for business agility and business transformation [Buc+10a].

1.1. Motivation

The *enabler*-role of the IT, discussed in the preceding section, builds upon a holistic view of an organization. This understanding is the foundation for the definition of an *enterprise architecture* (EA). According to Buckl et al. [Buc+10a] and in line with *ISO/IEC 42010:2007* [ISO07], an enterprise architecture is defined as follows:

> Enterprise Architecture (EA) is the fundamental conception of the organization in its environment, embodied in its elements, their relationships to each other and to its environment, and the principles guiding its design and evolution.

The enterprise architecture comprises not only business- and IT-concepts, but also accounts for crosscutting aspects such as strategies and projects. Based on this understanding, a management function is derived that seeks to address the aforementioned alignment of business and IT by taking the embracing perspective of the overall EA. This management function defines the guidelines for the future

evolution of the EA and supervises its adherence. This *managed evolution* of the EA, according to Murer, Worms, and Furrer [MWF08], is depicted in figure 1.1. In line with Buckl [Buc11], this management function, known as *enterprise architecture management*, is defined as follows:

> EA management is a continuous management function seeking to improve the alignment of business and IT and to guide the managed evolution of an organization. Based on a holistic perspective on the organization, the EA management function is concerned with the management, i.e. the documentation, analysis, planning, and enactment, of the EA.

Models are indispensable means for the representation of enterprise architectures, and as such also for the EA management function. EA models build the foundation for the EA planning activities and are used to provide a documentation of the architecture. Consequently, the area of EA modeling has received increasing attention in the recent years from both, practice and research. Whereas several approaches for EA management exist, the topic of EA planning has not yet been addressed comprehensively. Initial attempts, as presented by Aier and Gleichauf [AG10a; AG10b] as well as Buckl et al. [Buc+11] include a terminological basis for the understanding of EA planning. Nevertheless, a comprehensive perspective on the topic has not yet been established and a framework for the strategic EA planning is still lacking.

The present work contributes an extensive view on the field of strategic EA planning and approaches the design and conceptual development of a framework targeting the planning of EA transformation. The detailed objective is presented subsequently.

1.2. Research objective and approach

Following the motivation presented in the preceding section, this work aims to provide a comprehensive perspective on the planning with EA models by providing a framework for the strategic planning of EA transformation. The research objective has been specified as follows:

> **Research objective:** Design and describe a framework for the strategic planning of enterprise architectures.

This research objective has further been subdivided into three distinct research questions. These questions guide the design of the framework and purport the approach towards the final result. In the following, the research approach is presented and the research questions are introduced.

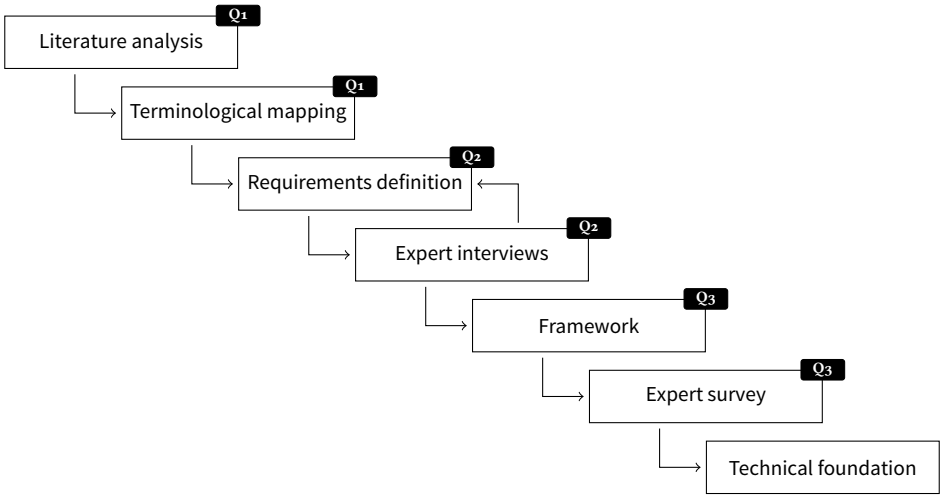


Figure 1.2.: Research approach

This work starts with an analysis of state-of-the-art EA management approaches. These approaches are analyzed regarding their contributions to the field of EA transformation planning and the major concepts are discussed. Based on this analysis, a terminological mapping is presented. This mapping relates the distinct terms used for common concepts and provides a foundation for the further discussion of EA planning. These initial two steps contribute the answer to the first research question, defined in the following:

> **Q1** Which concepts concerning the strategic planning with EA models can be identified in state-of-the-art EA management approaches?

Based on the identified concepts and the proposed common terminology, the research approach is moved further towards the design of an EA framework with a definition of concerning requirements. These requirements specify the techniques needed to support strategic planning activities and define the functionality of the planning-framework. These requirements are iteratively discussed in expert interviews, to evaluate the completeness of the requirement set and to gain input regarding possible refinements. Both steps contribute to the overall objective by answering the following research question:

> **Q2** Which requirements regarding the strategic planning with EA models exist?

The refined requirements set and the input from the expert interviews is then used as the foundation for the conceptual development of a strategic EA planning framework.

This framework and the underlying requirements are subsequently evaluated in an online survey regarding the importance and completeness of the requirements and the suitability of the framework. These further two steps contribute to the third research question and therefore conclude the main research objective. The research question is defined in the following:

> **Q3** How does a framework for the strategic planning with EA models look like?

The final contribution of this work goes beyond the core research objective and presents a first step towards an implementation of the framework in a software application. In this step, possible technical foundations are investigated and evaluated regarding their suitability for the EA framework.

The transcription of the presented research approach into the single chapters of this thesis is described in the following outline.

1.3. Outline

The following section presents the outline of this thesis and gives a brief summary of the contents of each chapter.

Chapter 2 gives an overview about related literature and discusses the key concepts introduced in several state-of-the-art EA management approaches. Section 2.1 describes the selection of the approaches. In section 2.2, the relevant approaches are introduced and their key concepts are discussed. Section 2.3 provides a mapping of terms regarding common concepts found in the literature analysis.

Chapter 3 specifies the requirements needed to support EA-planning activities and presents the strategic management framework. In section 3.1, the requirements are defined and subsequently discussed. Section 3.2 presents the design of the framework and discusses its structure. In section 3.3, the application of the framework in an exemplary context is described. Section 3.4 presents the evaluation of the framework and its underlying requirements based on expert interviews and an online survey.

Chapter 4 presents possible technical foundations for an implementation of the framework. In section 4.1, the open-source model repository EMFStore is evaluated. In section 4.2, the MoVE model versioning tool is analyzed. Section 4.3 compares the results of both evaluations and discusses their suitability as the technical foundation for the planning-framework.

Chapter 5 concludes this work with a summary of the central results and an outlook towards possible future research. In section 5.1, the results are summed up and the major contributions recapitulated. In the final section 5.2, the work is critically reflected and possibilities for future research are shown.

2. Literature review

In this chapter, different EA management approaches from academia and practitioners are analyzed with regard to their support of strategic EA planning. In section 2.1, the selection of approaches, based on their relevance for the area of strategic planning, is described. These approaches are then subsequently discussed and analyzed in section 2.2. Following that, a terminological mapping is presented in section 2.3, that relates the common terms.

2.1. Selection of relevant approaches

Since two decades, the field of EA management receives increasing attention from academic researchers, practitioners, and standardization bodies. This is well reflected in the increasing number of publications on the topic. The extensive literature analysis of Mykhashchuk et al. [Myk+11] shows that the topic is not well received in the high-ranking journals on information systems or computer science. A literature search based on the indices of top journals in these disciplines therefore may not lead to a comprehensive coverage of the state-of-the-art. A search using non-topical search mechanisms, like search engines, will result in an extensive list of publications with limited relevance for the review. The selection of relevant approaches is therefore based on the work of Buckl and Schweda [BS11]. The authors conducted an extensive analysis of the state-of-the-art in enterprise architecture management literature. The analysis covers 22 EA management and EA management-related approaches from different research groups and individual authors, spanning a total of over 150 publications. The authors distinguish between the methodical aspects of each approach, described and indicated in the literature, and the proposed language prescriptions for modeling the EA. Additionally, a fact sheet is provided, covering general information about the approach, such as the name, issuing organization and dedicated tool support.

The method-centric analysis examines several aspects of the methodical prescriptions. The *integration* aspect is concerned with the exchange of information between the EA management function and the other enterprise management functions such as project portfolio or strategy management. The authors thereby distinguish between no integration, unidirectional integration and bidirectional integration, regarding the direction of exchange between a receiver and a sender function. The *develop & describe*

perspective targets the creation of EA models representing different architectural states and the guidelines that constrain the architecture evolution. The authors classify approaches describing the current, planned and target states, as well as the principles that describe the guidelines and the questions that build an understanding in respect to EAs. The same classifications are used in the context of the *communicate & enact* aspect to gather information about the communication of EA-related information to the corresponding stakeholders. The *analyze & evaluate* perspective focuses on the analysis and comparison of architectural states. A classification is performed covering the current, planned and target state, as well as a comparative delta-analysis. Regarding the *configuration* of the EA management function to the organizational environment, the authors distinguish between approaches providing no mechanisms for configuration, mechanisms for the configuration to an organizational context, and mechanisms to configure to scope and reach. The same classifications are used regarding the *adaption* of the EA function, that might arise from the successful implementation and increased maturity of the management function.

The language-centric analysis focuses on the description languages used to document and describe the architectural states. The analysis framework therefore distinguishes between a *business & organization* layer, an *application & information* layer, and an *infrastructure & data* layer. These three architectural layers are regarded taking up various perspectives. The *black-box perspective* may be recognized as a functional decomposition of the enterprise, focusing on the general functionality of each layer and hiding the respective inner structure. In contrast, the *white-box perspective* illuminates this inner configuration and can thus be regarded as a structural decomposition of the enterprise. The analysis of *strategies & projects* covers the organizational evolution of the enterprise through projects and their general direction in strategies. *Visions & goals* analyze the support for the description of an intended end-state, *principles & standards* the support for design constraints. The perspective *questions & metrics* focuses on the support for measurement techniques and their relations to the corresponding architectural elements that should be measured. Finally, the flexibility of the language is analyzed taking up the *configure & adapt* perspective. Therein, the approaches are classified according to their possibility to configure and adapt the language to the specifics of the organization. The authors distinguish between approaches providing no mechanisms for configuration, approaches that allow an initial configuration of the language, and approaches that allow for a continuous evolvement of the language.

The selection of approaches is based on a subset of these analyzed aspects. Thereby both, the method and the language aspect of the analysis is covered. Regarding the method-centric analysis, classifications related to the development as well as the communication of architectural states are chosen. The selection therefore comprises the *develop & describe* as well as the *communicate & enact* perspective, both regarding

the current, planned, and target states of the EA. These states manifest the evolution of the architecture and act as a basis for communication. Regarding the language-related analysis, the selection focuses on the classifications of *strategies & projects*. These aspects cover the projects that drive the organizational evolution as well as the strategies, that indicate their direction. As all three organizational layers together build the enterprise architecture, all three classifications are included in the selection.

Table 2.1 shows the analyzed approaches and their classifications according to the selected aspects. The selection focuses on approaches that feature both, broad methodical support for the three architectural states as well as an extensive support of language concepts regarding the description of strategies and projects. Overall, 8 approaches offer corresponding qualities. These are: TOGAF, described in section 2.2.1, ArchiMate (section 2.2.2), GERAM (section 2.2.3), SEAM (section 2.2.4), BEAMS (section 2.2.5), the approach of the university of St. Gallen (section 2.2.6), the approach of the TU Lisbon (section 2.2.7) and the approach of Hanschke (section 2.2.8).

Table 2.1.: Classification of EA management approaches according to [BS11]

	Method classifications						Language classifications				
	Develop & Describe			Communicate & Enact			Strategies & Projects				
Approach	current	planned	target	current	planned	target	Business & Organization	Application & Information	Infrastructure & Data	Evaluate	
ArchiMate	✓	✓	✓	✓	—	✓	✓	✓	✓	✓	2.2.2
ARIS	✓	—	✓	✓	—	✓	—	—	—	—	
BEAMS	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	2.2.5
DEMO	✓	—	✓	✓	—	✓	—	—	—	—	
DYA	—	—	✓	—	—	—	—	—	—	—	
E2A	✓	✓	✓	✓	✓	✓	—	—	—	—	
EA³ Cube	✓	✓	✓	—	—	—	—	—	—	—	
EAP	✓	—	✓	✓	—	✓	—	—	—	—	
FEAR	✓	—	✓	✓	—	✓	—	—	—	—	
GERAM	✓	—	—	✓	—	—	✓	✓	✓	✓	2.2.3
Hanschke	✓	✓	✓	✓	✓	—	✓	✓	✓	✓	2.2.8
IAF	✓	—	✓	✓	—	✓	—	—	—	—	
KTH Stockholm	✓	✓	—	✓	—	—	—	—	—	—	
MEMO	✓	—	—	✓	—	—	—	—	—	—	
MIT	—	—	✓	—	✓	✓	—	—	—	—	
Niemann	✓	✓	—	✓	—	—	—	—	—	—	
SEAM	✓	✓	—	✓	✓	—	✓	✓	—	✓	2.2.4
SOM	✓	—	✓	✓	—	✓	—	—	—	—	
St. Gallen	✓	✓	—	✓	✓	—	✓	✓	—	✓	2.2.6
TOGAF	✓	✓	✓	—	—	—	✓	✓	✓	✓	2.2.1
TU Lisbon	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	2.2.7
Zachman	✓	—	—	✓	—	—	—	—	—	—	

2.2. Overview about relevant approaches

In the following section, each of the preceding selected EAM approaches is described in summary. Thereby, a rough overview about each approach is given and complemented with details about aspects relevant for the strategic planning of enterprise architectures.

2.2.1. TOGAF

The Architecture Forum of *The Open Group*, a global consortium that develops open and vendor-neutral IT standards and certifications, published the first version of *The Open Group Architecture Framework* (TOGAF) in 1995. This first version was based on the *Technical Architecture Framework for Information Management* (TAFIM), developed by the US Department of Defense (DoD). The current version 9.1 of TOGAF was released in July 2011 [OG11]. TOGAF is an architecture framework that provides methods, models and techniques for the development of an EA management function. It uses the terminology of *ISO/IEC 42010:2007* [ISO07]. TOGAF is one of the most prominent and widely used EA management frameworks with lots of tool support [Mat+08]. The framework consists of seven main parts, that each specializes on a different aspect of the management framework.

Part I provides a high level *Introduction* to the key concepts of enterprise architecture management and the TOGAF approach in particular. It contains the definitions of terms and describes the changes between the TOGAF versions.

Part II contains the *Architecture Development Method* (ADM), which describes a method for developing and managing the lifecycle of an enterprise architecture in an iterative step-by-step approach. As this part focuses on the methodical aspects, it builds the core of TOGAF.

Part III provides the *ADM Guidelines and Techniques*, that cover aspects of adaptability and configuration of the ADM, to deal with different usage scenarios, process styles and specific architectures.

Part IV presents the *Architecture Content Framework*, a conceptual metamodel for architectural content, that allows the major artifacts of the ADM process to be structured and defined.

Part V describes the *Enterprise Continuum & Tools*, which discusses methods for the classification and organization of architectural artifacts and tools for architecture development.

Part VI contains the *TOGAF Reference Models*, more specifically the *TOGAF Foundation Architecture Technical Reference Model* (TRM), which provides a model and a taxonomy of generic platform services, and the *Integrated Information Infrastructure Reference Model* (III-RM), which focuses on the application software.

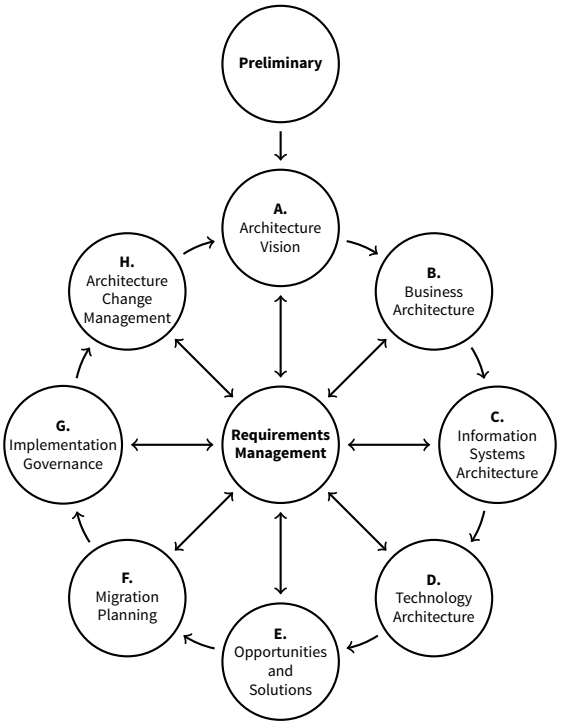


Figure 2.1.: TOGAF Architecture Development Method cycle [OG11]

Part VII finally describes the *Architecture Capability Framework*, which discusses the organizational structures, processes, roles and responsibilities needed to establish and operate an enterprise architecture function.

Altogether, these seven parts contain a comprehensive collection of aspects that need to be addressed in the context of an EA management function. The main part of TOGAF, the Architecture Development Method, describes the process of architecture development. The ADM is a continuous, iterative process, consisting of eight phases, preceded by a preliminary preparation phase and complemented with a central requirements management activity. Figure 2.1 gives an overview about this process.

The cycle is initiated in the *Preliminary* phase, which is about the preparation and initialization of the EA management function. In this phase, an EA team is established and the key drivers and the organizational context are identified. The architecture principles are defined and an EA framework, together with supporting tools, is selected.

The first phase in the ADM cycle covers the *Architecture Vision*. This phase defines

the scope of the management approach. The stakeholders and their concerns are identified and a high-level vision of the architecture is derived. This vision is then approved by the management.

The following phases B, C, and D describe how the *Business Architecture*, *Information Systems Architecture*, and *Technology Architecture* is derived. In each phase, a target architecture is being developed, based on the current baseline architecture and under consideration of the architecture vision. A gap analysis is then performed to identify the differences between the two architectures. Thereafter, the transition procedure is described in a roadmap and the impact of the architectural change is handed to the following architectural layer.

In the phase *Opportunities and Solutions*, the transition procedures of the three architectural domains are consolidated into a cross-domain roadmap. Planned changes are grouped into projects, programs, and portfolios to deliver continuous business value.

The preceding initial step in delivering an implementation and migration plan is finalized in the *Migration Planning* phase. In this phase, the transition between the baseline and the target architecture is detailed and coordinated with the enterprises overall change portfolio.

In the following phase, the *Implementation Governance*, the selected projects are initiated, executed and monitored. The outcome of the projects is then reviewed for architecture compliance.

The final phase of the ADM cycle, the *Architecture Change Management* phase, prepares the next iteration of the architecture development process. In this phase, the current iteration gets analyzed and monitoring tools are deployed.

Simultaneous to the execution of the ADM cycle, the central activity of *Requirements Management* is performed. This activity deals with the dynamic nature of the requirements, which may change throughout the process in an unforeseen manner. Within this activity, the requirements are recorded in a repository and synchronized with the relevant architecture development phases.

Overall, the TOGAF framework presents a comprehensive collection of activities and techniques that can be used to implement and perform an EA management function. Regarding the strategic planning, the ADM describes in detail how such a process could look like and which steps to execute in succession.

2.2.2. ArchiMate

The ArchiMate approach roots in the work of Jonkers et al. [Jon+03], who therein outline a “language for coherent enterprise architecture descriptions”. In 2009, this language has been adopted by *The Open Group* and published as the *ArchiMate 1.0 Specification* [OG09]. In this specification, a language is described that provides

a uniform representation of architecture descriptions. Besides the specification, there are complementing documents that enrich the model description language to a comprehensive EA management function. Overall, ArchiMate offers an integrated EA approach and provides a description and visualization language for the different architectural domains and their underlying relations and dependencies. Thereby, the language is oriented towards existing modeling standards such as the *Unified Modeling Language* (UML) [ISO12a; ISO12b].

Inspired by the natural language, where a sentence has a subject, verb, and object, the ArchiMate language distinguishes between *active structure* elements (subject), *behavior* elements (verb) and *passive structure* elements (object). The active structure elements represent the business actors, application components, and devices that exhibit actual behavior. This behavior is represented by the behavior elements, who build the connection between the active and passive structure elements. The passive structure elements finally mark the objects on which behavior is performed. These are usually information- or data-objects, but may also be physical objects.

Incorporating ideas of the service orientation paradigm, the ArchiMate specification distinguishes between an *external* and an *internal* view on systems. A service represents a unit of functionality that is described through an interface and provides some value. A system exposes this service to its environment by implementing the interface. Other systems can then use this service without having to implement the functionality themselves. The internal view describes the system implementing the service, whereas the external view describes the calling system. Figure 2.2 depicts this generic core metamodel of ArchiMate. This metamodel formalizes the differentiation between structural and behavioral elements and makes clear the distinction between the external and the internal view.

The language furthermore defines three main layers that specialize the core concepts described above. For each layer, there is a metamodel defined, that constitutes the foundation for respective architectural models. The *Business Layer* contains concepts that allow products, contracts, and business services to external customers to be modeled. The *Application Layer* focuses on the application services that support the business layer and that are realized in software applications. The *Technology Layer* offers the infrastructure services, like processing, storage, and communication, that run the applications and are realized by computer- and communication-hardware and system software [OG09; Lt04]. Figure 2.3 gives an overview about these language concepts and maps common areas of interest to the framework.

As the second EA related standard under the aegis of The Open Group, the two standards ArchiMate and TOGAF (see section 2.2.1) can easily be used in conjunction. Corresponding to their scope, TOGAF defines the methods and tools for developing an enterprise architecture, whereas ArchiMate provides the language used to describe and visualize the models. The structure of the ArchiMate language, with its three

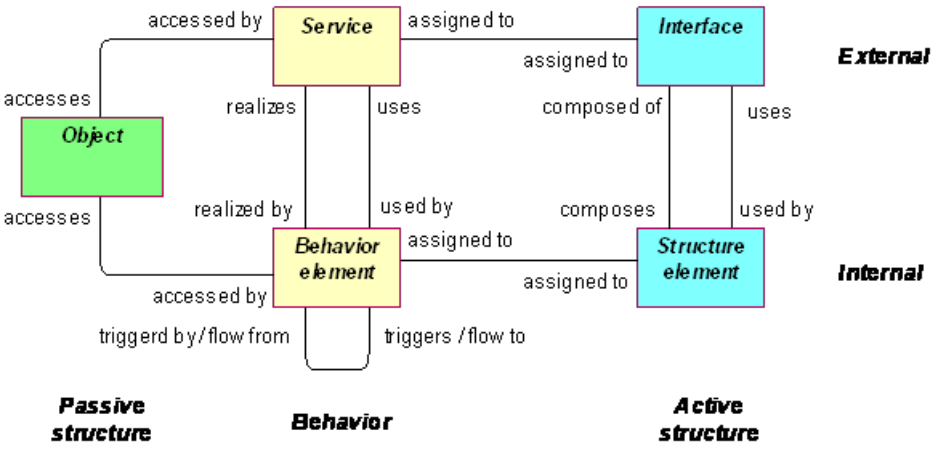


Figure 2.2.: ArchiMate Core Metamodel [OG09]

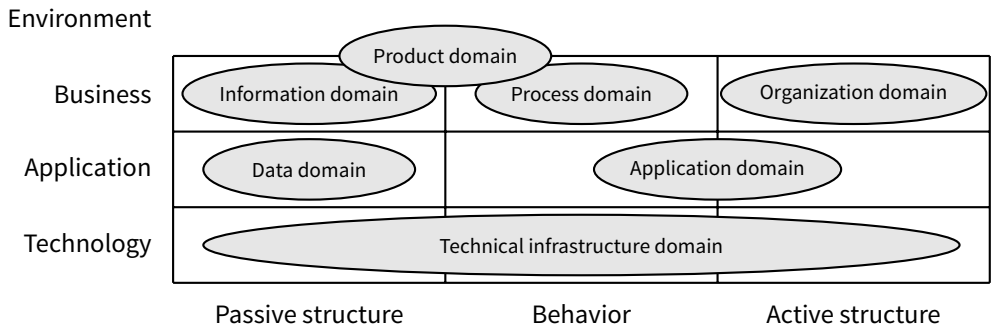


Figure 2.3.: ArchiMate Architectural Framework [OG09]

layers for the business, application, and technology aspect, thereby corresponds to the phases B, C, and D of the TOGAF ADM cycle, where the business architecture, information systems architecture and the technology architecture is developed [JPT09a]. Considering only the core ArchiMate specification document [OG09], there are TOGAF views that have no counterpart on the ArchiMate side, like e.g. for the migration planning. Conversely, the ArchiMate concepts that deal with the relationships between the architectural layers find no representation in the TOGAF views. Besides considerations to further merge the two standards [JPT09b], these missing mappings are already partly provided by ArchiMate extensions.

One such extension, the *Motivation Extension* [QEJ10], focuses on the integration of TOGAF's early ADM phases and the requirements management process. The authors propose three steps, which build a generic requirements engineering cycle that can be repeated at the successive phases of the ADM process. In the first step, the *problem investigation*, the stakeholders and their problem-related concerns are analyzed. Subsequently, goals are derived, that deal with the problem. The second step is used to *investigate solution alternatives*. In this step, the goals are refined in order to find possible solutions. The last step handles the *solution validation*, where alternative solutions are validated to choose the most appropriate one. This requirements engineering cycle can be repeated in the successive phases of the TOGAF ADM process, which leads to the concept of *problem chains*, where a problem gets solved by a solution, which in turn becomes a problem for a subsequent layer. A solution for a business problem could e.g. become a problem in the context of the application layer, where a new software application must be deployed. The proposed "way of working", which integrates this approach, translates each iteration into a refinement of an architecture model, that solves one problem. In case of alternative solutions, a trade-off analysis can be performed to determine which model satisfies the predefined goals best. In this process, the refined model should be defined relative to the respective preceding one, as it acts as a frame of reference to the further developments.

A second extension [Jon+10] focuses on the implementation and migration aspect that follows the three architecture development phases. The implementation and migration phases E, F, and G of the TOGAF ADM target the identification of possible work packages and projects, to implement the developed architectures. Besides the introduction of model elements representing projects and programs, the authors propose the concept of *plateaus*. In line with the TOGAF Transition Architectures, which reflect periods of transition between the baseline and the target architectures, plateaus describe a certain time interval, whereby a transition between two plateaus corresponds to a specific change in the enterprise architecture. Two plateaus are separated by a *gap*, which conceptualizes the difference between them and can be used as the output of the gap analysis in the architecture development phases. Alternatives

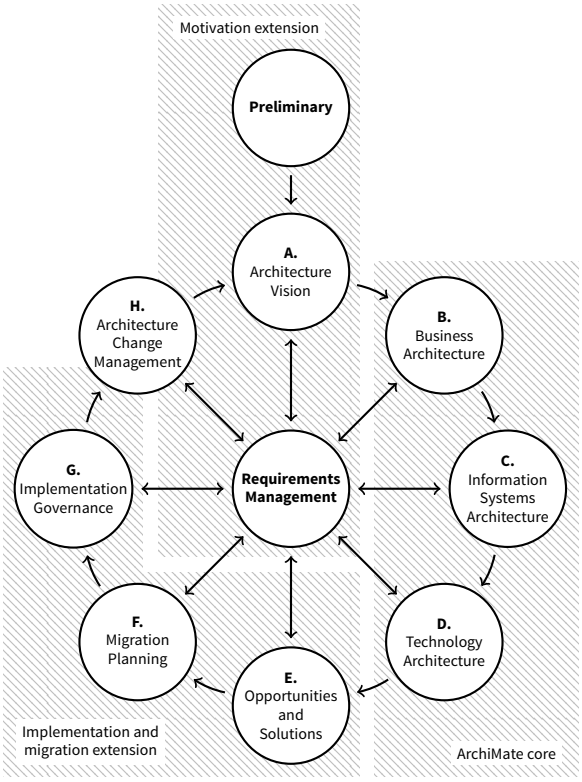


Figure 2.4.: ArchiMate Extensions and their relations to the TOGAF ADM [Jon+10]

for a certain plateau can be modeled with *splits* and *joins*. This allows multiple transition procedures to be developed and evaluated in parallel.

The two described extensions enrich the ArchiMate modeling language to support the phases of the TOGAF ADM cycle that are not targeted by the core specification. The first extension augments the language to be able to support the modeling of goals, requirements, and principles in the TOGAF ADM cycle and provides a methodology that facilities its use. It thus targets the Requirements Management process, Architecture Change Management, and the development of the Architecture Vision, together with the preceding Preliminary phase. The second extension focuses on the introduction of modeling concepts that support the implementation and migration aspect and thus enriches the phases Opportunities and Solutions, Migration Planning, and Implementation Governance. Figure 2.4 depicts these extensions in the context of the ArchiMate core specification and the TOGAF ADM process.

Overall, ArchiMate focuses on the modeling aspects of EA management and can thus be used to supplement the TOGAF standard or other EA management approaches.

Besides the core language specification, there are complementing documents that extend the specification and add missing pieces with regard to the TOGAF ADM. Considering the strategic planning, this approach supports an extensive language for the description and visualization of enterprise architectures and provides descriptions of collaborative planning activities based on these models.

2.2.3. GERAM

In the context of several emerging EA-related frameworks in the 1970s and the 1980s, the *International Federation of Information Processing* (IFIP) and the *International Federation of Automatic Control* (IFAC) established a task force aiming at the development of a reference framework, that supports the comparison and evaluation of existing approaches. This *IFIP-IFAC Task Force on Enterprise Integration* subsequently developed the *Generalised Enterprise Reference Architecture and Methodology* (GERAM) [99], a framework that in 2000 became part of the international standards *ISO 15704:2000* [ISO00] and *ISO/DIS 19439* [ISO]. GERAM covers the methods, models, and tools needed to build and maintain an integrated enterprise and organizes existing enterprise integration knowledge [BNS03]. It is intended to compare, evaluate, and combine existing EA management approaches and can be used to identify their overlaps and complementing benefits [99].

GERAM identifies nine components, depicted in figure 2.5, that together build a standard for tools and methods, to handle the change processes that occur during the operational lifetime of an enterprise. The components, which are described subsequently, do not impose any particular set of tools or methods, but define the criteria that must be satisfied by an EA management approach.

The *Generalised Enterprise Reference Architecture* (GERA) represents the core component of GERAM. It describes the basic concepts to be used in enterprise engineering and integration projects. GERA distinguishes between human oriented concepts, process oriented concepts, and technology oriented concepts. The *human oriented concepts* cover human aspects such as capabilities, know-how, skills, and competencies. They capture the knowledge about the roles of individuals and groups, and describe the way those roles are structured and coordinated through responsibilities and reporting lines. The *process oriented concepts* aim at modeling both, the functionality and the behavior, of business processes as well as the life-cycle, life-history, and activities of enterprise entities. Two kinds of enterprise entity types are defined: The *Operation oriented Enterprise Entity Types* are concerned with different types of operations, such as products, projects, service, and manufacturing. In contrast, the *Recursive Enterprise Entity Types*, deal with the necessity, means, methodology, and results of enterprise engineering and integration efforts. Finally, the *technology oriented concepts* provide descriptions of the technology involved in both, the enterprise

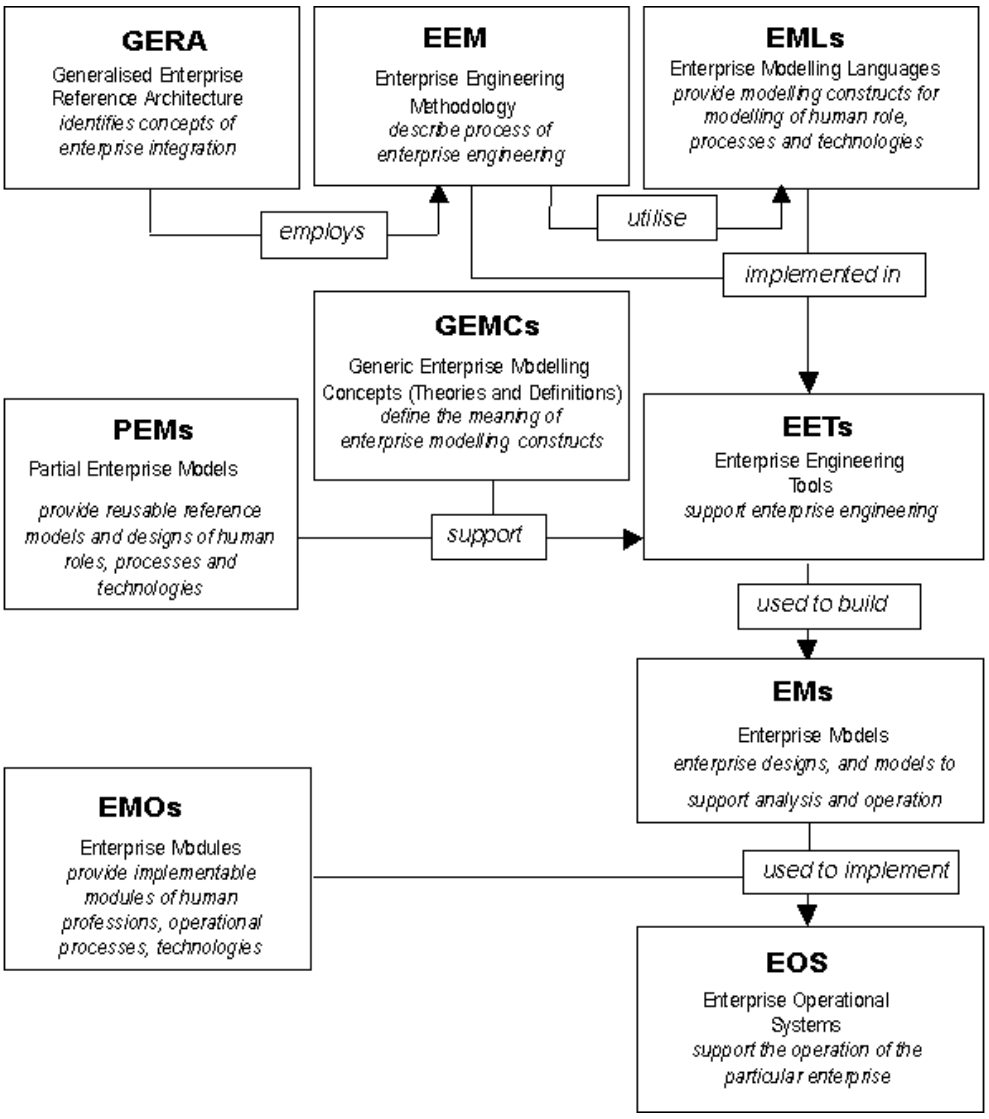


Figure 2.5.: GERAM Framework Components [99]

operation and the enterprise engineering efforts. Thereby, the focus of these concepts is on model portability and interoperability, as well as on model driven operational support, to enable a real time and online operation of the enterprise.

The second component, the *Enterprise Engineering Methodology* (EEM) describes the process of enterprise integration. An engineering methodology guides the user in the engineering task of enterprise modeling with process models or detailed instructions for each type of life-cycle activity in the integration process. Different methodologies may exist that cover distinct aspects of the change processes. To allow the combination of various processes in the context of the specific engineering task, the processes should be defined independent of each other. Thus, a methodology can be applicable to complete integration projects as well as incremental changes as experienced in a continuous improvement process.

The *Enterprise Modeling Language* (EML) defines the generic modeling constructs for enterprise modeling. More than one modeling language might be needed to develop an enterprise model. With respect to the enterprise engineering methodology, one suitable modeling language has to be selected for every enterprise entity type in every area of the GERA modeling framework. A model developed in one subject area must be able to be integrated with models of other subject areas if the information content is related.

Generic Enterprise Modeling Concepts (GEMCs) define and formalize the most generic concepts of enterprise modeling. The concepts might be defined in forms of natural language explanations (glossaries), meta-models (describing the relationships between modeling concepts), or formal definitions of the meaning (semantics) of enterprise modeling languages (ontologies).

Partial Enterprise Models (PEMs) are reusable reference models that capture concepts common to enterprises. Rather than developing models from scratch, these models capitalize on existing knowledge through model libraries and allow partial models to be reused in a 'plug-and-play' manner. They therefore make the modeling process more efficient and less error-prone. PEMs may cover the whole or a part of the enterprise under consideration, and typically concern various enterprise entities such as products, projects, and companies. However, such models generally still need to be adapted to a particular case.

An *Enterprise Engineering Tool* (EET) implements an enterprise engineering methodology and supports the creation, use, and management of enterprise models through enterprise modeling languages. It guides through the modeling process and employs analysis and evaluation capabilities to be used throughout the engineering activity. Reusable reference models are provided through a shared design repository. The tool needs to support collaborative as well as individual engineering, and should provide integration with enterprise management systems to keep the models up-to-date.

The *Enterprise Model* (EM) represents the particular enterprise. It is expressed

in enterprise modeling languages and includes all descriptions, designs and formal models that describe the various aspects of the enterprise. Enterprise models provide decision support for the evaluation of operational alternatives, allow the efficient business process execution through model driven operation control and monitoring, and acts as a communication tool that enables the mutual understanding of the enterprise.

Enterprise Modules (EMOs) are physical entities that can be used as building blocks or systems in the implementation of the enterprise. Examples for EMOs are human resources with given skill profiles or common IT infrastructure elements. In general, those modules are implementations of partial enterprise models.

The *Enterprise Operational System* (EOS) is the component that represents the operation of a particular enterprise. Its implementation is guided by the particular enterprise model which provides the system specifications and identifies the enterprise modules used in the operation.

In line with the objective of GERAM, to provide a generalized evaluation framework for EA management approaches, it abstains from presenting explicit EA related concepts, but instead focuses on the description of components needed by EA frameworks. Nevertheless, it covers requirements for strategic planning and provides abstract descriptions of EA models.

2.2.4. SEAM

The *Systemic Enterprise Architecture Methodology* (SEAM) is an approach for the "seamless" integration between business and IT from the group around Wegmann of the *École Polytechnique Fédérale de Lausanne* (EPFL). The approach stems from the work of Wegmann [Weg03], who diagnoses a lack of theoretical foundations in current EA management practices. The group thereupon tries to "bring more maturity to EA" and proposes SEAM, which is positioned as "an original methodology for Enterprise Architecture" [Weg03]. The work consists of several publications, that each complements and extends the methodology to a comprehensive approach. Overall, SEAM consists of a *philosophy*, a *method*, a *notation*, and SeamCAD [LW06], a prototypic computer-aided design (CAD) *tool*.

Using the discipline of system sciences, the systemic paradigm is introduced based on the distinction between complicated and complex systems. *Complicated systems* are described as deterministic systems, whose behavior can be predicted by an analysis of the components interactions. A typical complicated system is a computer. In contrast, *complex systems* are non-deterministic and their behavior is therefore not predictable by such an analysis. Furthermore, complex systems are subject of continuous evolution. Systems involving humans are typically described as such. An enterprise is therefore understood as a complex system, whose components are the

enterprise's resources. In light of this, the challenge for EA management is identified as the interaction and co-existence of complex and complicated systems. Wegmann [Weg03] proposes a "paradigm shift from the mechanistic paradigm used to understand complicated systems to the systemic paradigm used to understand complex systems". In contrast to the intuitively and implicitly used mechanistic principles, the systemic paradigm makes explicit the principles to reason about systems.

In the context of SEAM, an epistemology and an ontology are defined that together build the main foundation of the systemic philosophy. The *epistemology* is defined generically and grounds in the constructivism principle, which states all knowledge being relative to the observer. In this context, the organizational levels, like the business-, application-, and technology-layer can be understood as different viewpoints. In contrast, the *ontology* is SEAM-specific and defines the basic model concepts. These concepts are based on RM-ODP [ISO98], a generic reference model [Weg+07]. The SEAM ontology characterizes model elements according to five basic modeling characteristics and two specification characteristics. The *basic modeling characteristics* are thereby defined as object, action, state, location in time, and location in space, the *specification characteristics* as type and instance. In combination, the two characteristics build a model element. The basic modeling characteristic 'action' together with the specification characteristic 'type <sale>' for example represents an exchange of money for goods.

Besides this SEAM philosophy, a method is provided that describes the application of the approach. To adapt the model to the continuously changing reality, the method is defined iteratively. Each iteration comprises three development activities: The *multi-level modeling* activity concerns the development of new enterprise models or the modification of existing ones. The models thereby have to cover the current state of the EA as well as the anticipated state, both in all relevant organizational levels. The gaps between the two states are then identified for each level in the *multi-level design* activity. Furthermore, the planned changes are analyzed for their influence on subjacent levels. The *multi-level deployment* activity finally targets the transformation of the planned changes to artifacts that help initiate the change. Such artifacts might e.g. be plans or job descriptions.

Rychkova, Wegmann, and Balabko [RWB03] further extend this method by the recursive application of a computational and an information viewpoint. The *computational viewpoint* thereby describes the system as a set of physical objects, which interact via interfaces. In contrast, the *information viewpoint* is concerned with the information and information-processing performed. It thus describes the behavioral aspects of the system. These two viewpoints are then applied recursively, so a computational view is further detailed by the application of an information viewpoint on the subsystems of interest. This procedure is then repeated for all relevant organizational levels.

This method is backed in [LW04; LW05] by a graphical notation based on UML. This notation puts a special emphasis on the hierarchical aspect of the models and annotates the layer-spanning relationships with specific UML stereotypes. The notation is further enriched in [RWB03] with an operational semantics based on a description language for abstract state machines. This allows for an automated simulation and model-checking and a verification of the functional alignment [RW06].

In summary, SEAM not only contributes theoretical foundations based on a systemic view to the field of enterprise architecture management, but also provides a comprehensive method and a formalization language, which allows for an automated analysis. Especially the method and the philosophy offer valuable input regarding the strategic and collaborative planning.

2.2.5. BEAMS

At the Technical University of Munich, the chair for *Software Engineering for Business Information Systems* (sebis) covers the topic of enterprise architecture management at least since 2004. Several contributions in this context have been made: Starting with a research project regarding "software cartography" [MW04] and the description of multiple viewpoints concerning the "application landscape" [LMW05a; LMW05b], the research shifted from the strictly visualization centric topics to method and modeling aspects with an EA management tool survey [seb05]. Therein, the authors analyze an extensive set of EA-related tools for their modeling capabilities and their methodological foundations by means of common scenarios. More recently, a collection of performance indicators broadened the analysis aspects of the groups approach [Mat+12].

In [Ern08; Ern10], the research group proposes to develop an organization-specific EA management function based on *EA patterns*. These patterns represent proven solutions to common problems in EA management [Buc+08b]. A first collection of patterns has been presented in [Buc+08b] and was further exemplified in [Buc+08c]. The authors distinguish between patterns concerning the methodology, the viewpoint, and the information model. The *methodology patterns* thereby "define the steps to be taken in order to address problems of a distinct type" [Sch11]. Furthermore, they describe the intended usage context and delineate consequences that might arise by applying the steps. *Viewpoint patterns* are used to illustrate the aspects, relevant to the methodology. These patterns define how the conceptualization of reality, i.e. the corresponding enterprise information model, is presented in light of certain concerns. The *information model patterns* finally provide the conceptual model fragments for the description of relevant aspects of the EA. Using this approach, the solution for a problem can be described as the selection of a methodology, the configuration of a suitable viewpoint, and the assembly of an appropriate information model.

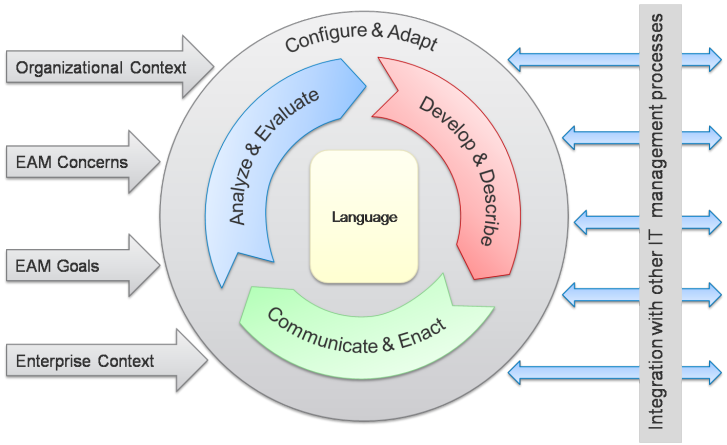


Figure 2.6.: BEAMS method framework [Buc+10a]

Diagnosing a lack of integration between the EA management patterns and describing the need for integration mechanisms, Buckl et al. [Buc+10a] introduce the *Building Blocks for Enterprise Architecture Management Solutions* (BEAMS) [Buc+10a]. As further detailed by Buckl [Buc11] and Schweda [Sch11], this approach refines the EA patterns with integration artifacts and provides an integrated method framework. This framework consists of four distinct development activities: The first activity, *Develop & Describe*, captures the description of either the current state of the EA or develops a planned or target future state. The second step in the cycle, *Communicate & Enact*, comprises the communication and the enactment of architecture states and principles to projects concerning the EA and related management functions as well as the project portfolio management. In the *Analyze & Evaluate* activity, architectural scenarios, i.e. planned states, are evaluated for their contribution to the pursued target state. Furthermore, the *Configure & Adapt* phase targets the EA management function itself and decides on concerns, goals and methods. The overall method framework is depicted in figure 2.6.

Based on this generic method framework, *Building Blocks* are used to configure the organization-specific management function in detail. Reflecting the dichotomy of method and language, the authors distinguish between *Method Building Blocks* (MBBs) and *Language Building Blocks* (LBBs). The former define the sequence of tasks that need to be performed in order to achieve a certain goal under a given organizational context [Buc+10a]. It details the steps to be taken, the decisions to be made, and the participants involved, in order to address a specific EA management concern [Sch11]. The LBBs conversely provide the syntax, semantics, and notation that facilitate the method. The MBBs therefore do not enclose the language primitives themselves,

but carry references to corresponding variables that are bound by appropriate LBBs during the configuration of the EA management function. In the context of BEAMS, two types of LBBs are defined. *Information Model Building Blocks* (IBBs) contain a practice-proven information model that reflects the particular EA concern. It thus provides the syntax and the semantics of the EA description language. A *Viewpoint Building Block* (VBB) in contrast defines the notation. This Building Block specifies the presentation of the backing information model to the stakeholder. Compared with the EA patterns, the three types of Building Blocks extend their counterparts with information about their integration and a concise theoretical foundation.

In light of the continuous evolution of the enterprise architecture, and with regard to multiple information models, covering the current and various planned states, the research group introduces modeling concepts to capture the data in a unified model repository. In several papers [Buc+08a; Buc+09a; Buc+10b], they therefore propose three "time-related" dimensions to allow for an identification and a traceability of model variants. The first dimension comprises the time an architecture is *planned for*. This dimension allows to distinguish between the current as-is model, and various future planned states. A second dimension captures the time an architecture has been *modeled at*. This enables a traceability based on different versions of the architecture models. The third dimension distinguishes between several *variants* of model states. These allow for different scenarios to be planned in parallel, each consisting of multiple versions of different transition states.

Overall, the group covers a multitude of aspects regarding the management of enterprise architectures. With BEAMS, they propose a flexible approach to build a dynamic EA management function based on best-practice. This approach furthermore allows for an integration with other EA management approaches [Buc+09b]. With the three "time-related" modeling concepts, the group further augmented the flexibility regarding the collaborative planning of an EA.

2.2.6. St. Gallen

At least since 2007, the university of St. Gallen is into the topic of EA management. At this time, Winter and Fischer [WF07] published their interpretation of essential layers of an enterprise architecture. Since then, the group has broadened their EA-related research and developed approaches covering many aspects of EA management.

Diagnosing a lack of descriptions respective the design and evolution of goals and processes, Hafner and Winter [HW08] propose a "consolidated process reference model for the managed evolution of the application architecture". They thereby focus on the application layer, as they describe it as being of particular importance due to its interfaces to the business- and IT-oriented levels. Based on the analysis of existing approaches found in literature, and the EA management processes at *Credit*

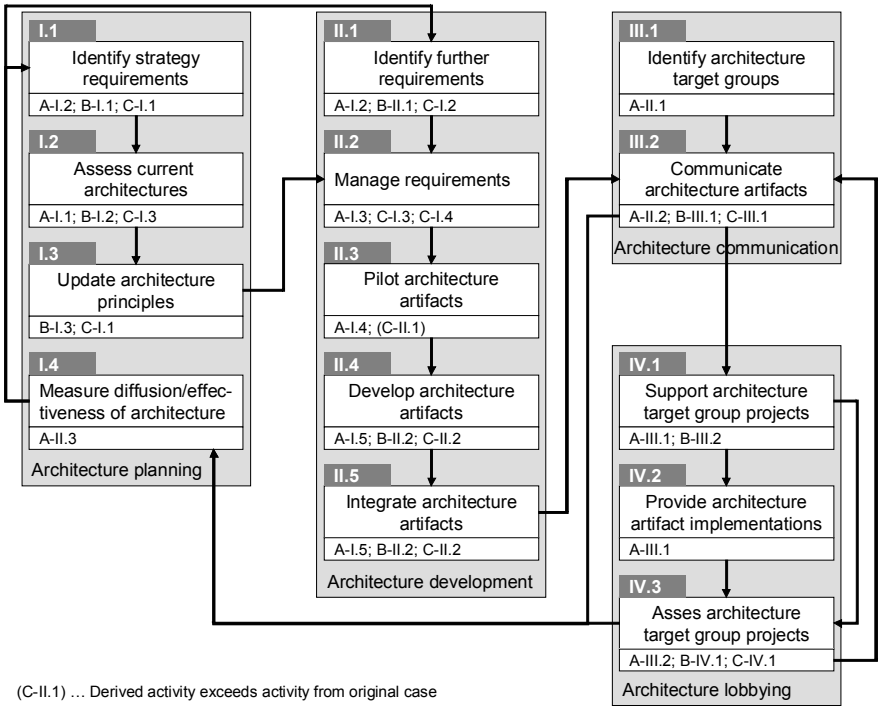


Figure 2.7.: St. Gallen process model for architecture management [HWo8]

Suisse, *Die Mobiliar*, and *HypoVereinsbank*, the authors derive an own consolidated process model. This model is depicted in figure 2.7. The process consists of 4 phases, that each contains several process steps. The *architecture planning* phase addresses strategic requirements and evaluates existing architecture models for any adaption requirements. The resulting architecture principles are then used for the further development of architecture artifacts. The *architecture development* phase covers the continuous identification and management of strategic and operational requirements from the entire enterprise. Architecture artifacts can subsequently be developed and integrated into the entirety of architecture artifacts. In the *architecture communication* phase, target groups are identified and supplied with information on the relevant architecture artifacts. Finally, the *architecture lobbying* phase provides assistance for strategic and operational projects that are relevant for the enterprise architecture. In this phase, standardized tools and method components are provided to the projects. In the project assessment activity, unavoidable inconsistencies are managed, which may produce new strategic or operational requirements, that need to be handled by the architecture management.

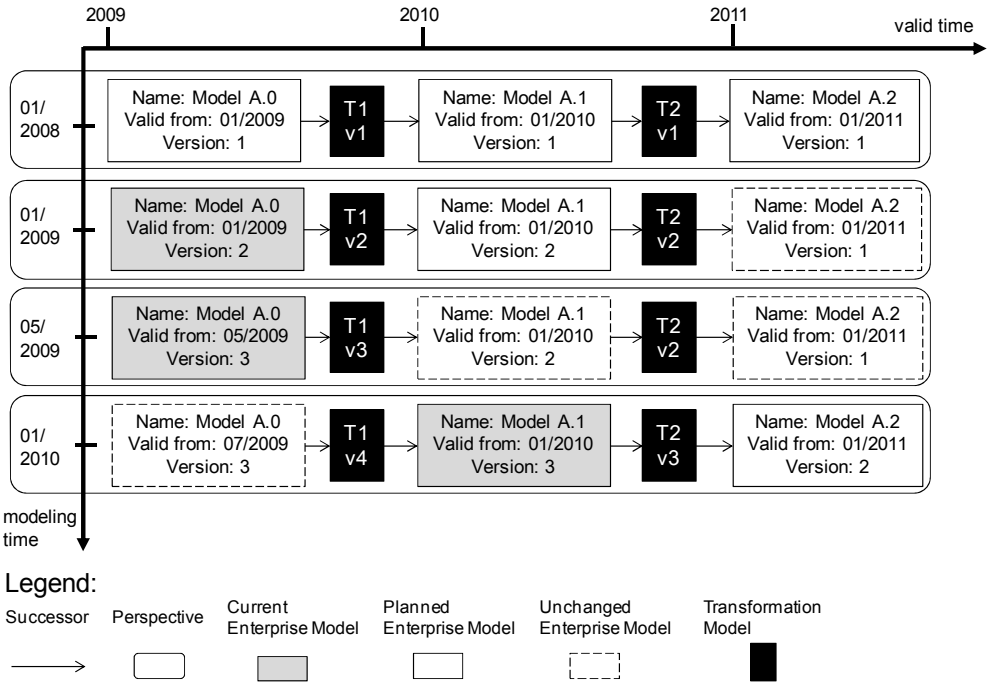


Figure 2.8.: St. Gallen model transformation macro level [AG10c]

Further illuminating the architecture development activity of the EA process, the St. Gallen EAM approach proposes to use models to capture the dynamics in enterprise transformation. Aier et al. [Aie+09] detail the process activities that document the current architecture, the desired target state, and thereupon the transformation steps. The authors incorporate dynamic aspects, which they capture in different levels of dynamic complexity. Besides various planned models for different points in time and alternative transformation plans, these levels also incorporate the requirement to support a deviation between the models and the reality. To facilitate these "unplanned shifts", the authors extend the typical notion of *as-is* and *to-be* models, representing the current and the planned state of an enterprise architecture, with *will-be* models. These are based on the corresponding planned state, but contain adjustments to respond to the unplanned deviations.

Based on these considerations, Aier and Gleichauf [AG10c] describe a mechanism to handle these requirements. To examine the dynamic transformation from two different viewpoints, the authors distinguish between the following two perspectives:

The *macro level* perspective, as shown in figure 2.8, focuses on whole models. The models are thereby mounted between two time dimensions: a *modeling time* and

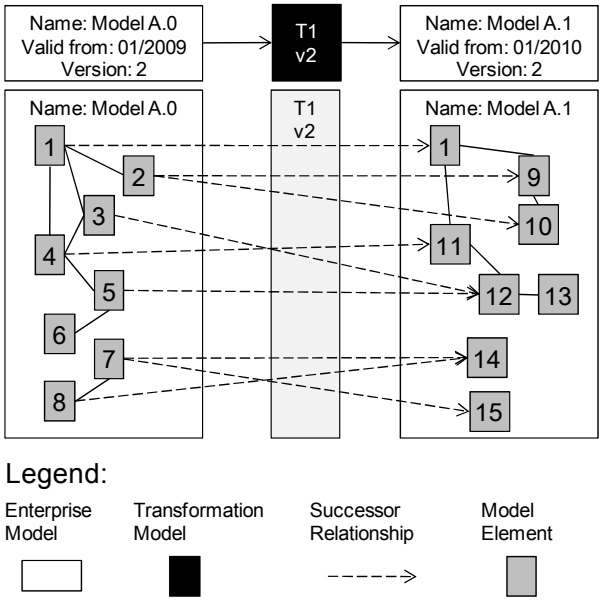


Figure 2.9.: St. Gallen model transformation micro level [AG10c]

a *valid time*. The former describes the point in time this model had been created, whereas the latter indicates at which time this model will be valid. The modeling time dimension can thus be used to capture versions of the same model, whereas the valid time dimension can be used to reconstruct the architectures evolution. As such, the valid time follows a fixed release cycle, whereas the modeling time is shown in irregular intervals, as changes can happen unexpectedly.

In contrast, the *micro level* perspective focuses on the the transformation aspect and visualizes the elements of temporal relationship between to models (cf. figure 2.9). Using this perspective, a current and a target state can be compared, and the transformation procedure be planned. This perspective allows to analyze which model elements have to be introduced, phased out, or migrated, and which elements stay untouched. Based on such an analysis, the authors propose to sequence individual transformation steps and apply some scheduling. This results in a procedure model, which can act as the basis for a realization of the planned EA.

Summarizing this, the research group at the St. Gallen university provides approaches covering various aspects of EA management. Regarding the strategic and collaborative planning with EA models, their reference process model and especially their approach concerning the modeling under the influence of dynamics provides valuable insights.

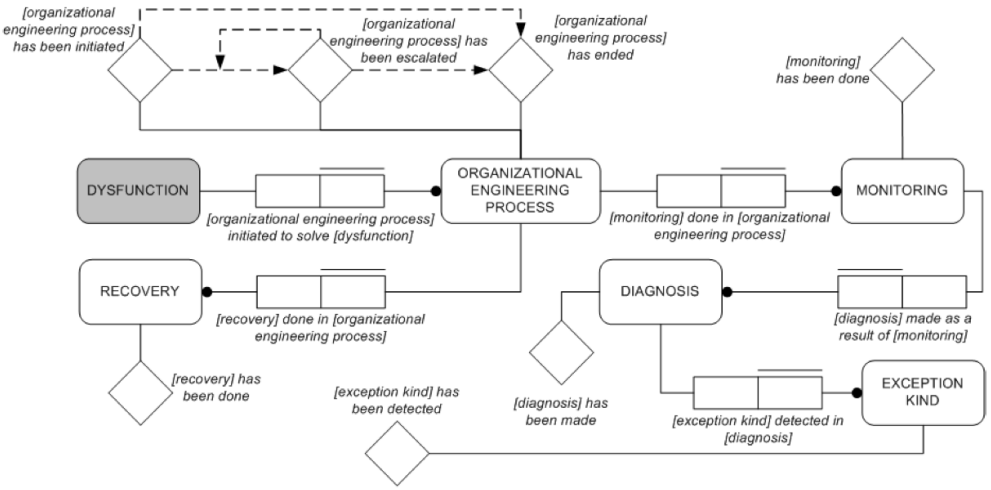


Figure 2.10.: TU Lisbon GOD Model [AST10a]

2.2.7. TU Lisbon

The research group of José Tribolet at the TU Lisbon deals with enterprise architectures since at least 2001. The first approaches thereby focus mainly on the modeling aspects of the *Information Systems Architecture* (ISA). The ISA is described as an intermediate level between the enterprise architecture and the software architecture, and "addresses the representation of the IS components structure, its relationships, principles, and directives, with the main purpose of supporting business" [VST03]. Based on this definition, the group extends the *CEO Framework* [Vas+01] — a basic meta-model to represent business processes, resources and goals — to support further elements in the context of information systems [VST03]. The authors use the existing concept of UML profiles to describe this framework.

Whereas this early work abstains from detailing actual steps to perform, the groups attention shifted in the more recent work from the modeling aspects to the method related aspects of EA management [AST10a; AST10b]. Adopting a functional perspective regarding an enterprise, the group defines a deviation from the normally expected values of an organizational system as a *dysfunction*, that can possibly compromise its viability. Consequently, an exception enforces a reaction in order to eliminate or circumvent the determined cause of dysfunction. The authors distinguish between two main types of change dynamics: *resilience* and *microgenesis*. Whereas the former describes the application of an already defined resilience strategy to react to certain known exceptions, the latter describes a change to the organizational artifacts in order to handle unexpected dysfunctions.

Regarding the microgenesis aspect in [AST10a], the authors diagnose a lack of concepts and methods concerning the explicit capture and management of information concerning exceptions and their handling. Such a description, capturing resilience- and microgenesis-dynamics, could thereby act as a reference regarding new resilience strategies and organizational artifacts to solve new exceptions. Based on these considerations, the authors propose to extend the *Design and Engineering Methodology for Organizations* (DEMO) [Die06; Die99], a comprehensive operational engineering approach, with modeling constructs and a method for continuous update of organizational models. In the context of this *(re)Generation, Operationalization and Discontinuation* (GOD) approach, a model is introduced that covers the monitoring, diagnosis, and recovery of an organizational exception. This model then acts as a starting point for the research of unknown dsyfunctions. The model is depicted in figure 2.10.

Aveiro, Silva, and Tribolet [AST10b] further enrich the GOD approach with an ontological foundation and a viability model. In light of this, an abstract notation for the description of measurements is introduced. These measurements are then combined with a notation of conditions, which together allows for an evaluation of measures according to a given frequency. If the evaluation results in the conclusion of a dysfunction, this might be caused by the occurrence of an already known exception, for which a resilience strategy exists. For an effective management, the authors propose to keep track of object in *Object History Lists* and *Object Responsibility Lists*. The former thereby presents a chronologically ordered list containing the several versions of a certain object and the transactions causing each change. This list can then be used in the occurrence of an unknown exception to diagnose the cause of the new dysfunction on the basis of the last performed changes. In addition, the Object Responsibility List contains a chronologically ordered list of the several versions of an object along with the actor roles and respective human agents that were responsible for a change. This list can then provide information on the people that were involved in the past and might have additional information to solve the problem.

Overall, the research group provides approaches covering diverse aspects of EAs. In light of the strategic and collaborative planning, especially the GOD approach contributes new considerations regarding EA management.

2.2.8. Hanschke

A practitioners approach regarding the strategic management of enterprise architectures is presented by Hanschke [Han10a; Han10b]. Diagnosing a lack of practical orientation in research literature, the author focuses on providing a best-practice "toolkit" derived from customer projects. The book covers various aspects regarding the EA management, ranging from business planning to IT operations. Neverthe-

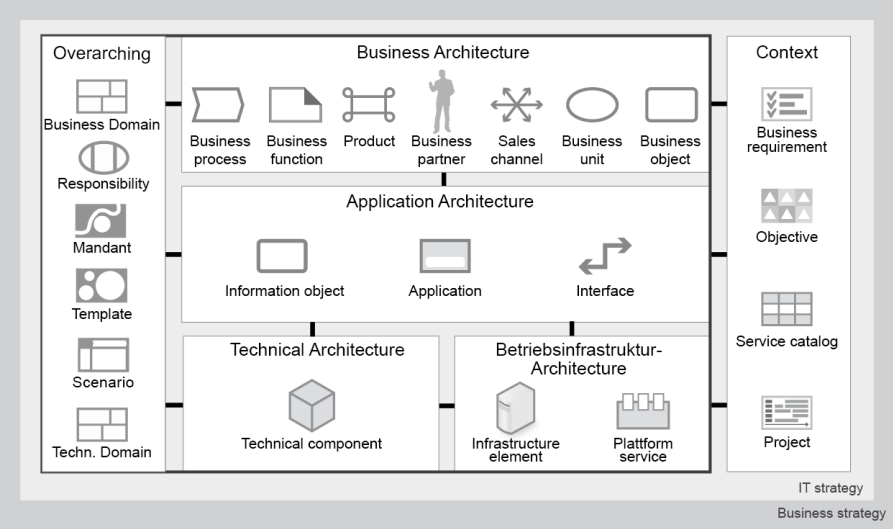


Figure 2.11.: Hanschke Best-Practice Enterprise Architecture [Han10b]

less, the focus lays on the enterprise architecture and the corresponding strategic management activity.

Regarding the theoretical foundations, Hanschke provides a *Best-Practice Enterprise Architecture* composed of four architectural layers, that each describes a different view on the EA. This architecture is depicted in figure 2.11. The *Business Architecture* describes the fundamental functional units constituting the business. In the *Application Architecture*, the supporting application systems, their interfaces and data is documented. The *Technical Architecture* describes the technical software components, i.e. databases, frameworks, and technologies, constituting the applications. Finally, the *Infrastructure Architecture* covers the hardware devices that run the applications and their components.

Based on this understanding, a process is outlined concerning the planning of the target and the planned states in each architectural layer. The process starts with the definition of the organizational context. Subsequently, the target architectures of the adjacent architectural layers are defined. They act as a point of reference for the future changes in the focused layer. The current state of the EA in context of the relevant layer is then analyzed for problems and possible improvements. Additionally, existing solutions and target states are examined for dependencies and implications. Possible solutions can then be developed based on common patterns and evaluated in context of the problem space. Target scenarios are determined by the consolidation of solution approaches. These scenarios are then refined to comprehend information about

the feasibility, duration, risks, and costs of the advance. The different scenarios are subsequently evaluated based on organization-specific criteria, and a recommendation is prepared. Based on this information, the EA management board decides about a mandatory target state. Finally, this state is documented and communicated, and the process repeated. As the gap between the current and the target state prevalently exceeds the scope of a planning cycle, transition states have to be developed that stepwise realize the target state. This process starts with the definition of the scope of the planning cycle. A delta-analysis is then performed to identify the gap between the current and the target state. Possible deltas that might appear include the installation of new elements and the change or phasing out of existing elements of the EA. Out of the collection of deltas, alternative scenarios are developed that target the transition between the current and the target state. In each scenario, transition measurements are defined and aligned with the operative necessities. These measurements are then consolidated into planning scenarios, that are subsequently evaluated and selected. Finally, a recommendation is documented and communicated to the project portfolio management function.

In line with the emphasis on the practical aspects, this detailed method is facilitated with an EA information model comprising the relevant elements of the respective architectures. Based on this enterprise information model, the open-source EA management tool *iteraplan*[1] is developed, that implements the prescriptions and guidelines, especially concerning the modeling perspective.

Overall, the approach as promoted by Hanschke provides an extensive description of the EA management function from a practical perspective. Regarding the collaborative strategical planning, especially the planning process, as outlined in detail in [Han10b], provides relevant insights.

2.3. Terminological mapping

In the preceding literature analysis, the terminological plurality regarding enterprise architecture management in general, and in the field of strategic EA planning in particular, became apparent. The various approaches covered the the analysis therefore use different terms for similar or equal concepts. As a foundation for the comparability of approaches and to facilitate cross-approach discussions, a mapping of the terms used in the context of EA transformation planning is provided. The presented requirements in the following chapter (cf. section 3.1) build upon this mapping and establish relationships between the key concepts.

The mapping is provided in table 2.2. The leftmost column presents the terminology used throughout the rest of this thesis. In the analyzed literature, not all approaches

[1] URL: http://www.iteraplan.de

Table 2.2.: Mapping of terms used for enterprise transformation planning

	Approaches							
Terminology	TOGAF	ArchiMate	GERAM	SEAM	BEAMS	St. Gallen	TU Lisbon	Hanschke
State / Scenario								planning scenarios
Current state	baseline architecture		particular architecture	current state	current state	as-is	current state	as-is landscape
Target state	target architecture		reference architecture	anticipated state	target state	to-be	norm	to-be landscape
Planned state	roadmap / planned changes			planned changes	planned state	to-be		planned landscapes
Difference analysis	gap analysis	gap		gap	delta analysis			gap
Revision / Version		plateau	life-history				object history list / version	
Duplication		split						
Consolida-tion		join						
Approval								appraised
Part of a state			view					sub-architecture
Access control							Access control and security	

use distinct terms for the concepts distilled in the mapping. Furthermore, certain concepts are only sparsely populated by matching terms from the analyzed approaches. Whereas the terms introduced in the upper part of the table have matching concepts in most approaches, the lower part of the table refers to concepts only few approaches cover. Concepts that are only implicitly handled in the approaches to date, and have no matching unambiguous terms, such as the *rescheduling* of plans, have therefore been left out in this mapping. The requirements proposal presented in the following chapter, as well as the subsequent framework, therefore seek to establish and integrate concepts that are not yet popular in the publications of this field, but nevertheless demanded by the community of practitioners (cf. section 3.4).

3. Framework for the strategic EA planning

In the previous chapter, approaches covering the field of EA transformation planning have been analyzed. Based on the analysis, this chapter presents a framework targeting the strategic EA planning. In section 3.1, requirements regarding strategic EA planning are defined and subsequently described. Section 3.2 builds upon these requirements and presents a framework for the strategic EA planning. This framework is exemplarily applied in section 3.3. In the final section 3.4, the justification of the requirements and the evaluation of the framework is presented.

3.1. Requirements for the strategic EA planning

Based on the analysis of relevant approaches regarding the strategic EA planning in section 2.2, a mapping has been presented in section 2.3, that relates the various terms used for the common concepts. These concepts are related in the following as requirements that constitute techniques needed to support strategic planning activities in a collaborative process. This set of requirements furthermore establishes constraints for a framework targeting the EA transformation planning. These 13 requirements are defined and described in the following.

> **R1** The framework must provide a mechanism to describe multiple *states of the EA*. This description of a state must contain all relevant elements, their relationships to each other, and their properties.

This first requirement asks for a mechanism to create an artifact that represents a state of the EA. This state must contain all organizational elements, relationships, and properties that are relevant for the execution of the EA management function in the context of the organization. The elements might be conceptually structured in architectural layers. An example for relevant elements are business processes that are supported by information systems, that in turn are executed on servers. The artifacts that describe a state of the EA might in practice be object-oriented models of the architecture, but can also be other structured information or even textual descriptions.

> **R2** The framework must provide a mechanism to designate a state as the *current state* of the EA. This state reflects the present situation of the enterprise.

As shown in the literature review (cf. chapter 2), existing EAM approaches distinguish between three distinct classifications of EA state descriptions. The present situation of an enterprise architecture is described as the current state. The framework has to provide a method to mark a certain state description as a representation of this present situation.

> **R3** The framework must provide a mechanism to designate a state as being the intended future *target state* of the EA. This state represents the unscheduled long-term vision of the architecture.

A second classification of EA states according to literature describes the envisioned target state of the EA. This state is derived after analysis of the current state and becomes manifest in the description of the target state. It describes the long-term goal of the architecture transition and provides guidelines for more concrete implementation plans. A target state could e.g. describe the migration from a mixed Microsoft .NET and Java architecture to a consolidated enterprise-wide Java platform.

> **R4** The framework must provide a mechanism to designate a *planned state* as being intended to take effect at a given future point in time. There may be multiple planned states scheduled to be realized at the same time. These planned states guide the EA evolution from the current to the target state.

The third distinct EA state described in literature is the planned state. This state represents a transition step on the path between the current and the target state. There might be multiple planned states defined, that lay out consecutive steps towards the target state. Each planned state should be realized until a certain specified point in time, that is assigned with this state. Furthermore, there might exist multiple planned states scheduled for the same time, that represent alternative steps towards the target architecture.

> **R5** The framework must provide a mechanism to *reschedule a planned state*. Therefore, the envisioned point of realization must be adaptable.

As the implementation of planned changes does not always succeed for the originally intended dates, planned states might need to be assigned to a different date. The framework therefore has to provide a mechanism to change the point in time, the state is intended to have effect.

R6 The framework must provide a mechanism to *revise a state*. The result must be represented as a *new version* of the same state. Each version must be accessible independently.

Due to the dynamic nature of enterprise architectures, possible iterations in the process of EA management, and the fact that changes between two states might be extensive, it must be possible to revise a once defined state. This allows an Enterprise Architect to e.g. retrace changes of the real architecture in its representation as an EA state, adjust an already planned state according to input from other architects, or perform a major modification in multiple steps. This leads to multiple versions of the same state, that represent the respective intermediate results. To make the current condition reproducible and its evolution understandable, each of these versions has to be accessible and analyzable. This allows to trace the changes that led to a given condition.

R7 The framework must provide a mechanism to create a *duplicate* of (a part of) a state. Each duplicate must be represented as an own state.

As different states commonly build on each other, there is a need for an existing state to build the foundation for the definition of a new state. A planned state could e.g. be defined based on the already described current state, with only minor changes. The framework therefore has to provide a mechanism to create a copy of a state, that is then treated as an own state, where subsequent changes can be applied independently from the original. This allows an Enterprise Architect to e.g. duplicate an existing current state, and then apply changes that lead to a final representation of a planned state. Furthermore, there might be the need to duplicate only a part of a state, if certain sections of the overall architecture should be handled independent of the other. This allows to create distinct planned states for each department in an organization.

R8 The framework must provide a mechanism to *consolidate* the descriptions of any two states into a new version of one of these states. The framework must support the selection of the state comprising the consolidation.

As changes, that are applied in one state, might need to be adopted in other states, the framework has to support a consolidation of two states. Thereby, the changes that constitute the differences between two states must be able to be adopted into a new version of the selected one of these states. This can e.g. be used to consolidate the distinct planned state for a department into a state comprising the planned state for the whole organization. Another use case can be to update a planned state with changes performed in the current state during the planning phase. Likewise, planned changes can be taken over into the current state upon achievement.

> **R9** The framework must provide a mechanism to *reject* a state. Consequently, it must ensure that rejected states can not be changed anymore.

This requirement addresses the need to handle abandoned states. States might e.g. be abandoned as they represented a competing alternative to a state, or they implemented complementing changes that have already been consolidated into a mainline state. These abandoned states can then be rejected to elude them from further changes.

> **R10** The framework must provide a mechanism to *access any version* of a state description for the purpose of analysis.

To enable a traceability of the evolution of a state, the distinct steps that have been performed in this state and ultimately led to the present condition have to be captured in various versions. It must be possible to use one arbitrary version, i.e. the most recent or one of the preceding versions, as the basis for further work. Therefore, each version has to be accessible and analyzable.

> **R11** The framework must provide a mechanism to *determine differences* between any two versions. Consequently, it must distinguish between new, revised, and rejected elements, relationships, and properties.

As the state descriptions might be extensive, and the differences between two arbitrary versions therefore not directly clear, the framework has to provide a mechanism that allows for an analysis of the differences between two versions. Based on a selection of two versions, this mechanism has to present all elements, relationships, and properties that have been added to or rejected from one version compared to the other. Furthermore, all revised elements, relationships, and properties must be presented. A possible use-case would be to compare the last version of the current and the target state to extract the elements that have to be changed in transition. This analysis can then be used as the basis for the development of the planned states, that realize the steps towards the target state.

> **R12** The framework must provide a mechanism to mark certain versions as *approved*. There might be multiple approved versions at the same time.

Due to the dynamic nature of an enterprise architecture, states can be target of constant changes. In contrast, the EA function arises the need for an approval process, that documents states that have been discussed with certain stakeholders and finally approved by the management. The framework therefore has to provide a mechanism that allows for selected versions to be marked as approved. An approved version can

then e.g. be used as the basis for further planning, whereas the state that contains the approved version itself can be refined without affecting the parallel planning. When these refinements reach a stable status, the updated version can again be discussed and approved by the management.

> **R13** The framework must provide a mechanism to specify *user access rights*. Consequently, the mechanism must distinguish between reading and writing access, the duplication of states, their rejection and approval.

As multiple enterprise architects and stakeholders might be part of the planning process, there might be the need to constrain the access of certain users to selected actions. The mechanism has to be able to constrain the reading and writing of states, which can e.g. be used to hide planning scenarios in early stages from non-authorized users, or to prevent enterprise architects to adjust scenarios of other architects. Furthermore, there must be access rights that constrain the duplication of states as well as its rejection. This might e.g. be useful to enforce a certain structure and to prevent any architect to create its own scenarios without agreement. Finally, only the management might have the permission to approve a version, which can be expressed with corresponding access rules.

The presented 13 requirements specify the foundations for a framework targeting the strategic EA planning and describe the relations between the various concepts. As the requirements are based on an analysis of relevant literature (cf. chapter 2), a mapping between the analyzed approaches from and the presented requirements is given in table 3.1. In contrast to the mapping of terms in section 2.3, this table also includes concepts that are only implicitly covered in the selected approaches. One can see that the basic requirements such as the support of current, planned, and target states are covered by most approaches, whereas the more practical requirements such as access control and approval mechanisms are not dealt with in the literature.

Table 3.1.: Requirements support by selected approaches

Requirements		Approaches							
		TOGAF	ArchiMate	GERAM	SEAM	BEAMS	St. Gallen	TU Lisbon	Hanschke
R1	State description	✓	✓	✓	✓	✓	✓	✓	✓
R2	Current state	✓	✓	✓	✓	✓	✓	✓	✓
R3	Target state	✓	✓	✓	✓	✓	✓	✓	✓
R4	Planned state	✓	✓	✓	✓	✓	✓	–	✓
R5	Rescheduling	–	–	–	–	–	–	–	✓
R6	Revision	–	✓	✓	✓	✓	–	✓	–
R7	Duplication	–	✓	✓	–	–	–	–	✓
R8	Consolidation	–	✓	–	–	–	✓	–	✓
R9	Rejection	–	–	–	–	–	–	–	✓
R10	Arbitrary access	–	–	✓	–	✓	–	✓	–
R11	Difference analysis	✓	✓	–	✓	✓	✓	–	✓
R12	Approval	–	–	–	–	✓	–	–	✓
R13	Access control	–	–	–	–	–	–	✓	–

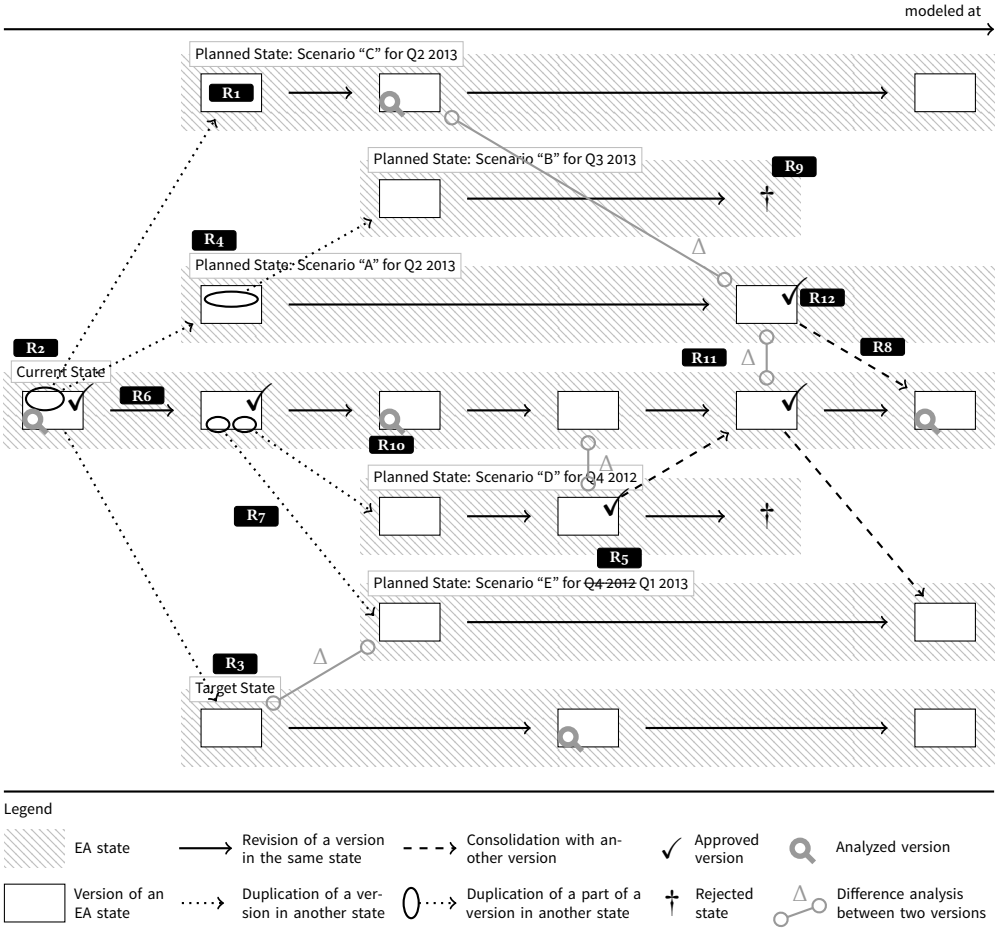


Figure 3.1.: Example for the application of the framework

3.2. Structure of the framework

Based on the requirements introduced in the preceding section, a framework for the strategic EA planning has been developed. This framework is illustrated in figure 3.1 and described in the following. The figure contains references to the requirements at the places their implementation is illustrated.

The overall structure of the framework resembles the basic structure of the various *version control systems* (VCS) known from the software engineering discipline. These systems manage files and directories, and the changes made to them, over time [CFP11]. A version control system is defined as:

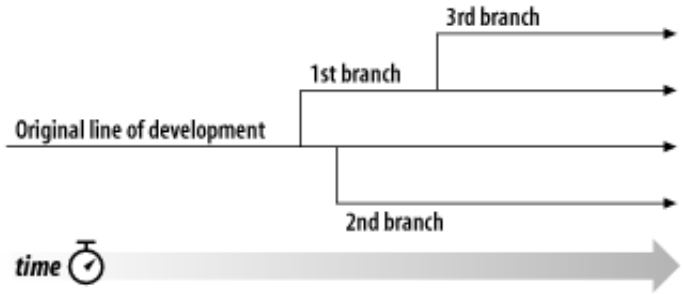


Figure 3.2.: Branches of development [CFP11]

> A version control system (or revision control system) is a system that tracks incremental versions (or revisions) of files and, in some cases, directories over time. Of course, merely tracking the various versions of a user's (or group of users') files and directories isn't very interesting in itself. What makes a version control system useful is the fact that it allows you to explore the changes which resulted in each of those versions and facilitates the arbitrary recall of the same. [CFP11]

As of today, there exist various version control systems, each having different strengths and weaknesses. Some of the popular open-source systems are the Concurrent Versions System (CVS)[1], Apache Subversion (SVN)[2], Bazaar[3], Mercurial[4] and Git[5]. Common to each of those systems are the concepts of *branching* and *merging*, that provide a method to handle alternative developments and their consolidation. A branch is thereby defined as follows:

> [...] This is the basic concept of a branch — namely, a line of development that exists independently of another line, yet still shares a common history if you look far enough back in time. A branch always begins life as a copy of something, and moves on from there, generating its own history. [CFP11]

The branching concept is illustrated in figure 3.2. The merging concept builds upon the branching and describes the reintegration of developments done in one branch into an other branch. The proposed framework is structured based on these concepts.

[1] URL: http://www.nongnu.org/cvs/
[2] URL: http://subversion.apache.org
[3] URL: http://bazaar-vcs.org
[4] URL: http://mercurial.selenic.com
[5] URL: http://git-scm.com

As these definitions already cover some of the requirements raised on section 3.1, these concepts have been used as the foundation for the development of the framework. The main deviation between the implementations of the concepts being the framework's handling of descriptions of the EA instead of files and directories as in typical version control systems. This covers requirement R1.

As illustrated in figure 3.1, an EA state can be seen as a branch in the sense of a VCS. In the illustration, one main branch exists, that represents the current state of the EA (R2), with various planned states (R3) and one target state (R4) being branched off this mainline ("trunk" in the context of VCS). In the illustration, an EA state is represented with a light-grey ruled background. Additionally to the type of state, i.e. current, planned, or target, the annotations of the planned states also cover the intended date of implementation. As a consequence of requirement R5, this date can be changed, which is shown in "Scenario E" in the illustration.

Within each state, rectangles are shown, that represent the various versions (R6) of the EA description. The continuous arrow in between two versions of one state represents the revision of a state and therefore the transition between the versions. The dotted arrows between two versions of different states represent a duplication of one state in a certain version into a new state (R7). This is in line with the definition of a branch, which states that a new branch begins its life as a copy of an existing branch. This copy can also contain only a part of an existing EA description, which is illustrated as a dotted arrow with an ellipse at its beginning, that covers part of the version. The consolidation of two states (R8) is implemented as a merge, that adopts the changes done in the context of one state into a new version of another state. In the illustration, this consolidation is represented as a dashed arrow. As shown with "Scenario D", the implementation of the consolidation as a merge can be used to reintegrate developments done in one scenario back into the master, or to update the scenario with changes done in an other state, as shown with the planned state "Scenario E". A rejection of an EA state as per requirement R9 is illustrated as a typographic dagger (†).

A further requirement that is supported through the imitation of a version control system is the access of arbitrary versions (R10). As depicted with a magnifying glass in the illustration, any version can be accessed, independent of newer versions existing in the same EA state. Beside being able to analyze a single version, the framework also allows to analyze the differences between two distinct versions. This corresponds to requirement R11. The difference analysis is represented with a line between the versions and the Greek letter delta (Δ). This difference analysis is possible between two versions in distinct EA states or within the same state. Finally, a checkmark is used to represent the approval of versions (R12). Thereby, multiple versions can be marked as approved at the same time. As the illustration visualizes an exemplary planning model, the technical and organizational aspects of access control and user access

rights claimed in requirement R13 are not covered in the figure. They nevertheless should be contained in a technical implementation of this framework.

The presented framework provides one possible design satisfying the requirements. With its adaption of concepts of the neighboring software engineering discipline, it builds upon existing knowledge and proven techniques. The illustration furthermore depicts the interpretation of requirements R1 to R12 in the context of the framework.

3.3. Exemplary application of the framework

In section 3.2, the proposed framework for the strategic EA planning has been presented. In the following section, the use of the framework and applicability in practice is shown based on exemplary use-cases presented in [Mat+08]. The requirements, which build the foundation of the framework, are thereby noted at the corresponding places in the text.

The fictitious department store *SoCaStore* wants to introduce an EA management function. The top management therefore establishes an EA board, where major decisions should be made, and assigns the responsibility for the performance of the management function to the IT-department of the headquarter. Together with the IT-department, the EA board decides to use an object-oriented tool that supports the presented framework (cf. requirement R1). To gain some experience in the area of enterprise architecture and to produce quick results, the EA board furthermore decides to start with only modeling organizational units, processes and software applications. After having configured the tool accordingly, the enterprise architects Alice and Bob of the IT-department start to gather the needed information about the headquarter, the three subsidiary stores and the warehouse. This data is subsequently modeled as the current state in the framework (R2). This current architecture is illustrated as a process support map in figure 3.3. Having completed the collection of data, the architects let the organizational units cross-check their respective architecture with an export of the information (R10). After all organizational units confirmed the correctness of the data, the EA board approves the respective version of the current state (R12). As the board noticed that many different applications are used for the same process steps throughout the organizational units, they decide to start developing a target state (R3), with the goal to homogenize the application landscape and increase the vertical integration[6].

The architects therefore duplicate the current state (R7), which is used as the foundation for the development of the target state. They also adjust the access

[6] An increase of vertical integration in this context symbolizes that an application system is used by more organizational units to support a specific process; in contrast, an increase of horizontal integration means that an application system supports more processes at a specific organizational unit.

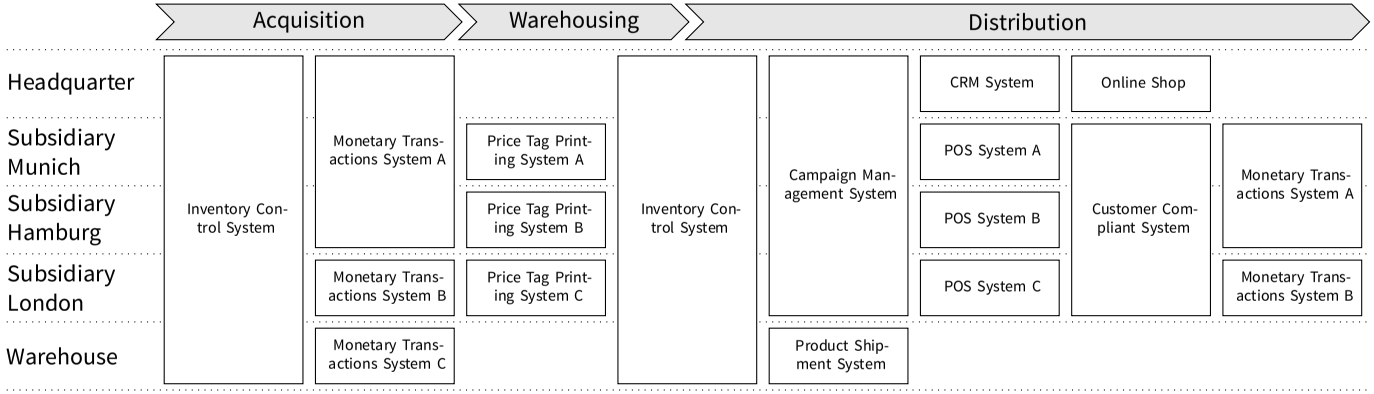


Figure 3.3.: Process support map showing the current state of the architecture

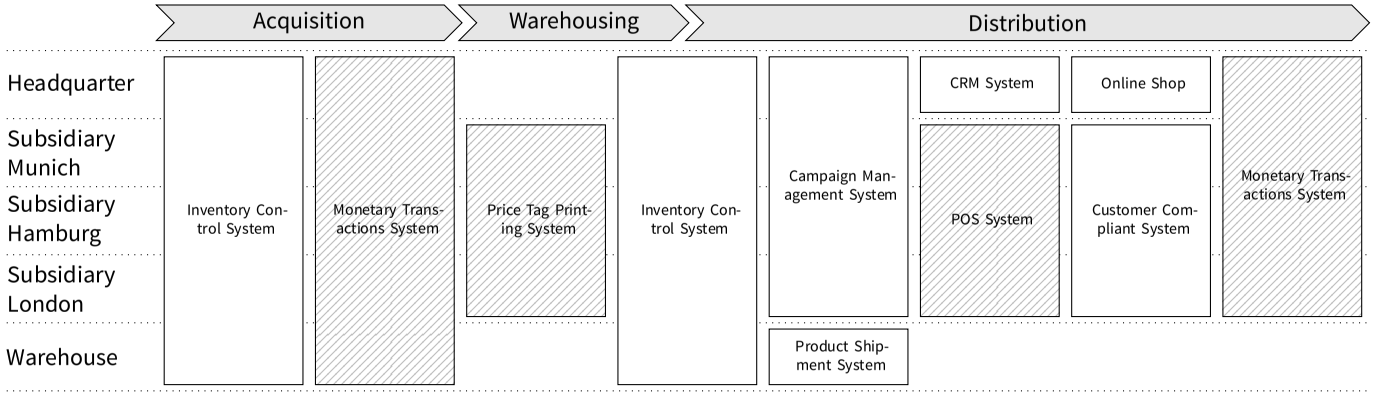


Figure 3.4.: Process support map showing the target state of the architecture

rights for this new state (R13), that only the board and the architects themselves can access the state. Within this new target state, the architects first homogenize the *Monetary Transaction Systems*, that are used throughout the *Acquisition* and *Distribution* process in each organizational unit. Furthermore, the architects model to use the *Price Tag Printing System*, that is being used in the subsidiary store in Munich, throughout the two other stores in London and Hamburg. After feedback from the EA board, the architects furthermore homogenize the *Point of Sale System* in the three subsidiaries. This target architecture is illustrated as a process support map in figure 3.4. In the meantime, the warehouse updated its *Product Shipment System* which was subsequently entered into to the model representing the current state (R6). After consolidation of this revised current state into an updated target state (R8), the EA board approves the final version and unlocks the target state for the organizational units. Being satisfied with the work of the two enterprise architects, the board asks Alice and Bob to create proposals for the transition from the current towards the target state.

Alice starts by duplicating the current state into a planned state (R4), which she calls *Scenario A1*. She therein models to homogenize the *Monetary Transaction Systems* throughout all organizational units and all affected processes, first. She then continues to branch off a second planned state, which only includes the architecture of the three subsidiary stores (R7). She calls this new planned state *Scenario A2*. Within this state, the *Price Tag Printing Systems* in the three subsidiary stores are harmonized. After a difference analysis between *Scenario A2* and the approved target state (R11), she notices that she forgot to harmonize the *Point of Sale Systems*. Alice therefore revises *Scenario A2* to incorporate the respective standardization of the three systems in the subsidiaries. Alice finally schedules the two plans to be realized within two subsequent half-year periods.

At the same time, Bob creates a *Scenario B1* based on the current state. Within this planned state, he models to harmonize the *Monetary Transaction Systems* used in the *Acquisition* process in all organizational units, as well as the *Price Tag Printing System* used in the *Warehousing* process throughout the three subsidiary stores. Bob schedules this first plan to be realized within a year. He then creates a second planned state based on the first one to harmonize the *Monetary Transaction Systems* for the *Distribution* process. He names this second branch *Scenario B2*. Finally, he creates an other planned state based on *Scenario B1*, to incorporate the standardization of the *Point of Sale Systems*. This state is named *Scenario B3*. In a last step, Bob schedules the planned states *Scenario B2* and *Scenario B3* to be implemented in parallel after completion of *Scenario B1*.

The architects finally present their plans to the EA board, which discusses both proposals and compares them with the current and target states (R11). The board finally decides to start with the proposal from Alice and therefore approves *Scenario*

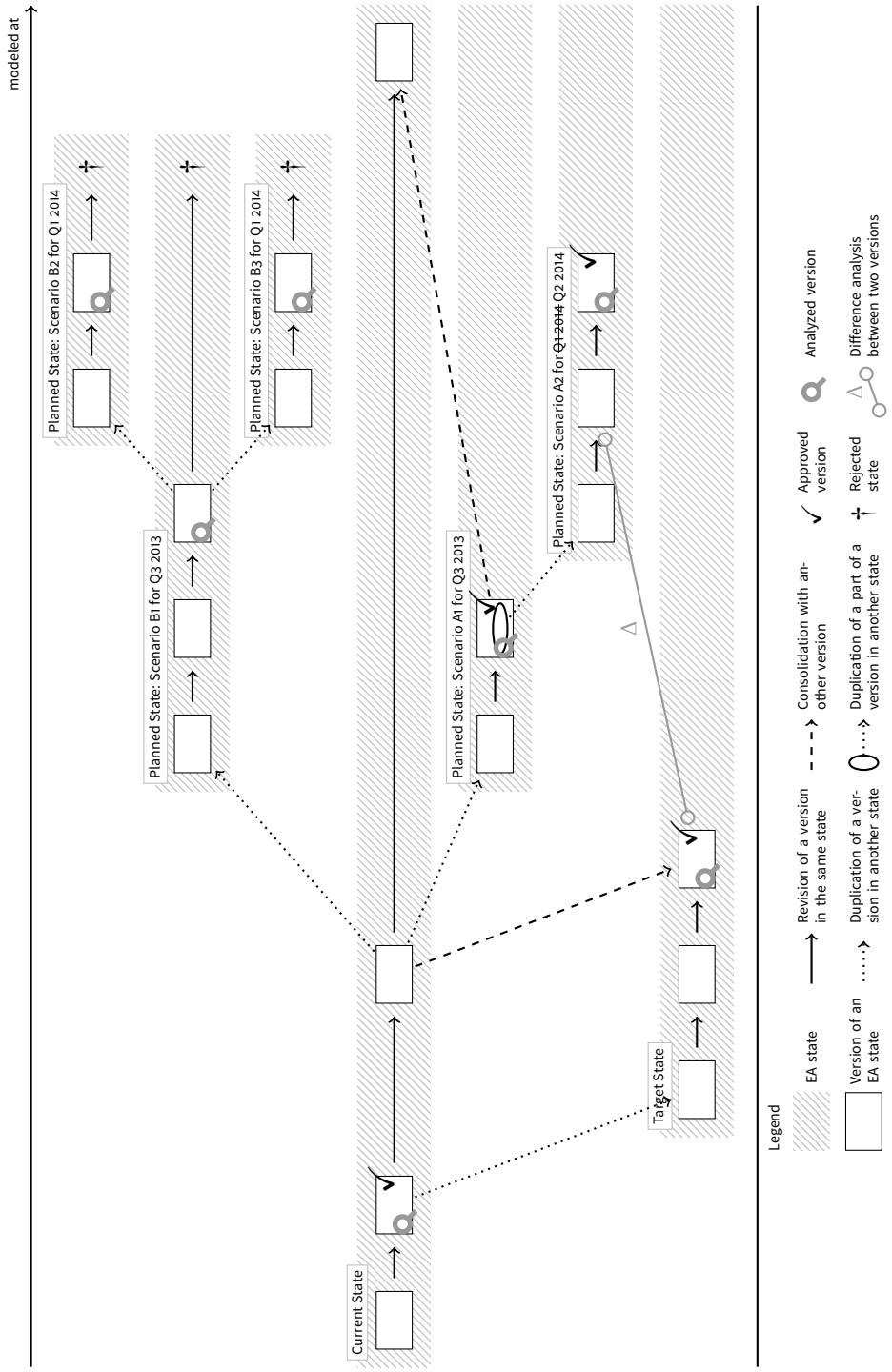


Figure 3.5: Schematic illustration of the exemplary application

A1 and *A2* and officially rejects Bob's plans (R9). The first approved scenario is then handed over to a project team, which is asked to implement the harmonization of the *Monetary Transaction System* throughout the organization. Due to the complexity of the system, the final implementation is delayed by one month, whereupon the enterprise architects reschedule their second plan (R5). After project completion, the architects update the EA documentation in the framework by consolidating the changes planned in *Scenario A1* into the current state. A schematic illustration of this application of the framework is shown in figure 3.5.

Being invited by the top management to present the progress so far, the EA board asks the architects to export the data about the architecture at the beginning of the EA initiative and the most recent version of the current state (R10). The board then finally presents this to the top management, which now plans to extend the EA management efforts.

3.4. Evaluation of the requirements

The evaluation of the requirements and the conceptual framework has been approached in two steps. In the first step, the requirements have been justified and refined based on expert interviews (cf. section 3.4.1). In the second step, an online survey has been conducted to evaluate the importance and completeness of the requirements and to gain input regarding the framework (cf. section 3.4.2).

3.4.1. Requirements justification and refinement

To validate the requirements for enterprise transformation (cf. section 3.1) and the corresponding framework (cf. section 3.2), three expert interviews have been conducted. The interview partners have been selected from university and industry contacts and all have multiple years of in-depth experience with the field of EAM. The interviews took place in October and November 2012. In each interview, we first provided a short introduction of our research goal. We then continued with a discussion of each single requirement, covering the application of its central concept in practice, its importance for the expert, as well as its description in detail. Finally, we asked for the completeness of the resulting requirement set and gained input regarding the conceptual implementation in the proposed framework.

The first interview has been performed on-site at a German organization on October 26, 2012 in a face-to-face meeting. It took about 1.5 hours. Although the overall set of requirements was confirmed during the talk, the interview provided input regarding several aspects: The interviewee highlighted the importance of an approval mechanism, which helps to verify and release certain descriptions of an EA state (cf. requirement R12), as well as the access control feature (cf. requirement R13), to reflect

the accountability of stakeholders via views. The interview partner furthermore reaffirmed the need for a distinct concept of a target state (cf. requirement R3), which we discussed to unify with the concept of a planned state. A new requirement that was brought to our attention during the talk was the need to be able to model parts of the overall landscape. The interviewee exemplified this with distinct departments that should be enabled to develop their own architecture further, without affecting the other departments. These distinct parts of the overall architecture can then be consolidated into a common organization-wide architecture, that can then act as a basis for consolidation efforts. This then new requirement has subsequently been incorporated into requirement R7.

The second interview was conducted on November 5, 2012 with an expert of a Swiss organization via a telephone conference. The call took about 45 minutes. The interview partner confirmed the need to model parts of the architecture, which has been brought up in the first interview (cf. requirement R7). He furthermore welcomed the intent to support multiple alternative planning scenarios (cf. requirement R3) and being able to compare and consolidate developments (cf. requirements R8 and R11), as there were a lack in the flexibility of this feature in current EA tools. The expert furthermore verified the overall requirement set.

The third interview was held via a video conference on November 9, 2012 with an expert of an Austrian organization. The interview took about one hour. During the consultation, the expert confirmed the overall set of requirements and strengthened the need to model "fragments", i.e. parts, of architectural models (cf. requirement R7). He additionally highlighted the importance of a history in the current state to enable a traceability of the realized changes (cf. requirement R6). Against the background of an "insufficient support" of strategic planning in existing approaches and tools, the interviewee welcomed the incorporation of the branching and merging concepts in the conceptual framework (cf. requirements R7 and R8), as they would allow for a "dynamic" planning. The expert repeated the mention of the need to control the access to the models and being able to release certain parts of the architecture to selected architects, which has also been addressed in interview one. This persuaded us to reflect this importance with its incorporation into a new distinct requirement (R13).

Overall, the experts provided valuable input regarding the justification and refinement of the basic concepts and their formulation as requirements. The interviews furthermore helped to validate the applicability of the conceptual framework in practice. These insights have then been used as the basis for the following online survey.

3.4.2. Requirements importance and completeness assessment

After the requirements justification and refinement, we conducted an online survey to validate the importance and completeness of the resulting requirement set. The survey contained 28 mostly closed questions in three groups that could be completed in about 10 minutes. The full survey and its results are provided in appendix A. The survey was accessible to a hand-picked audience of experts from various companies that have been selected from current and past research projects at the chair for *Software Engineering for Business Information Systems* (sebis). The survey started on Monday, November 26, 2012 and was officially open until Wednesday, December 5, 2012. The timespan contained 10 weekdays and 8 business days. An invitation to the survey has been sent out via e-mail on the first day and a remainder on December 3, 2012. A total of 63 experts have been invited of whom we received 3 incomplete and 5 complete answers. Only the completed answers are considered in the following evaluation.

The first group (cf. appendix A.2) consisted of 10 questions and gathered background information about the experts. The participants all work in different industry branches in Germany. Among the industries are manufacturing, telecommunication, consulting and government. 60% of the experts stated to work as an enterprise architect, 20% as an IT architect and 20% as a consultant. 40% of the participants have between 1 and 5 years of experience in the area of EAM, 40% between 6 and 10 years, and 20% over 10 years of experience. Most of the experts have an equal time of experience in strategic EA planning as in EAM in general, with only one participant being involved for less than 1 year in the area of strategic planning.

60% of the participants organizations apply an EAM framework, of whom 66% use more than one framework. All of the organizations that use a framework apply TOGAF (cf. section 2.2.1). One organization additionally used ARIS and NAF, and one other organization additionally applies the strategic IT management framework of Hanschke (cf. section 2.2.8). 60% of the organizations also use an EA tool to support the application of EAM, with one of the organizations using multiple tools. Among the tools used are ARIS, Troux and iteraplan. 100% of the organizations consider the current state in their EA efforts. Furthermore, 80% consider planned states and 60% the target state.

In the second part of the survey (cf. appendix A.3), the participants were asked to evaluate the importance of the identified requirements. This section contained 14 questions — one for each requirement and an additional question that allowed the participants to completely rule out one or more requirements as dispensable in the context of enterprise transformation planning. However, none of the experts opted to designate a requirement as dispensable. The evaluation of the 13 requirements was performed based on a five-point Likert scale [Lik32] to assess the importance from "strongly agree that this requirement is important" to "strongly disagree that this

Table 3.2.: Requirements importance

Requirement is important		Strongly agree 2	Agree 1	Neutral 0	Disagree −1	Strongly disagree −2		Total [−10; 10]
R1	State description	1	4	0	0	0	▪▮___	6
R2	Current state	2	3	0	0	0	▪▮___	7
R3	Target state	0	4	1	0	0	_▮▪__	4
R4	Planned state	0	4	1	0	0	_▮▪__	4
R5	Rescheduling	1	3	0	1	0	▪▮_▪_	4
R6	Revision	2	3	0	0	0	▪▮___	7
R7	Duplication	0	3	2	0	0	_▮▪__	3
R8	Consolidation	0	4	1	0	0	_▮▪__	4
R9	Rejection	0	2	2	1	0	_▪▪▪_	1
R10	Arbitrary access	0	2	0	2	1	_▪_▪▪	−2
R11	Difference analysis	0	5	0	0	0	_▮___	5
R12	Approval	0	3	1	1	0	_▮▪▪_	2
R13	Access control	0	2	1	1	1	_▪▪▪▪	−1

requirement is important". In the following analysis, the five possible answers are given a value in the range between 2 (strongly agree) and −2 (strongly disagree) to analyze the feedback numerically. Table 3.2 summarizes the respective answers. It shows the number of votes for each of the 13 requirements regarding all assessment values and a total in the range between −10 and 10 based on the five single answers. This importance is compared graphically in figure 3.6.

Based on this evaluation, experts agree that most of the requirements are important. With a summed importance of 7 each, requirements R2 and R6 can be considered the most important requirements, whereas requirements R10 and R13 are considered least important, with values of −2 and −1. Nevertheless, none of these least important aspects are considered dispensable.

The last part of the survey (cf. appendix A.3) consisted of 4 questions and asked for comments and feedback regarding the requirement set and the framework. All participants stated that an implementation of our framework in an EA management tool (accounting all 13 requirements) would be useful for their work in the area of

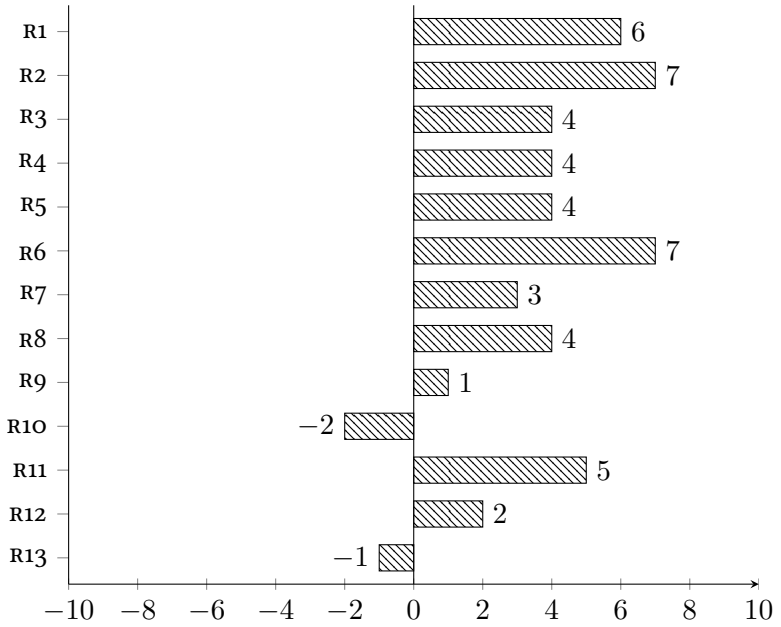


Figure 3.6.: Average requirements importance

strategic EA planning. Asked for any missing requirements, one participant noticed the need to link the proposed strategic EA planning framework with strategic business planning frameworks as well as frameworks covering the operational part of the enterprise. A second expert stated: "All the requirements are worthless without the right processes and tools to manage the resulting models."

Overall, the survey verifies the importance and completeness of the identified requirement set, with no requirements being dispensable and no requirements within the scope of the strategic EA planning being missing. Furthermore, the experts confirmed the importance of the overall framework.

4. Technical foundation

In the previous chapter, a framework for the strategic planning of enterprise architectures has been presented. In this chapter, possible technical foundations for an implementation of this framework are evaluated, to foster the realization of the framework in an EA management tool. As described in section 3.2, the proposed framework for the strategic EA planning resembles a *version control system* (VCS). The selection of possible technical foundations takes up on this fact and considers two different technical frameworks, that follow distinct approaches for model versioning. Whereas the typical VCSs known from the software engineering discipline are geared towards textual artifacts in files and folders (cf. section 3.2), the two selected frameworks feature the versioning of graph-structured models in their repositories. As such, they are better suited for the handling of object-oriented enterprise models [Bro+12], as typically used in common EA management tools. Both frameworks are developed in different contexts: The first framework, EMFStore, is a popular open source project. It is described and evaluated in section 4.1. In contrast, the second framework, MoVE, can be seen as a research project. An assessment of this framework is presented in section 4.2. The results of both evaluations are compared in section 4.3.

4.1. EMFStore

The first selected framework is described in the following. In section 4.1.1, general information about the framework is presented. An evaluation of the framework regarding the proposed requirements is provided subsequently in section 4.1.2.

4.1.1. General information

EMFStore[1] [KH10] is an open-source model repository, developed in the context of the Eclipse project[2]. It is implemented in the Java programming language and published under the *Eclipse Public License* (EPL). EMFStore is implemented as several components, consisting of a server, a client, and a UI, based on the *EMF Client Platform* (EMFCP). The components can be used together in an Eclipse-based product, but can also be exchanged, removed, or modified as needed. The EMFStore server is

[1] URL: `http://eclipse.org/emfstore/`

[2] URL: `http://eclipse.org`

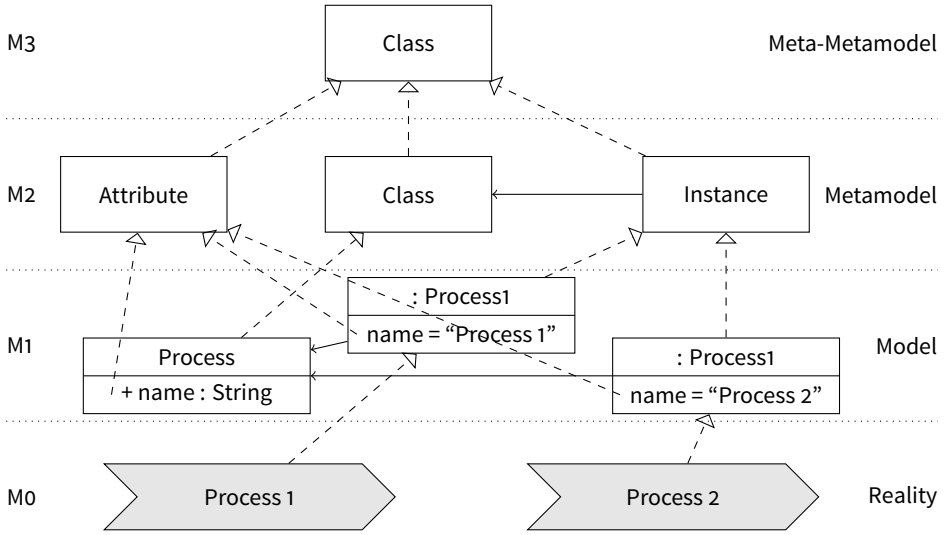


Figure 4.1.: MOF architecture

designed as a headless application and provides a well-defined API to be used by the provided or a custom client application [Hel12; KH12].

Regarding the model support, EMFStore builds on the *Eclipse Modeling Framework* (EMF), a Java-based implementation of the *Meta Object Facility* (MOF) [OMG11]. MOF is specified by the *Object Management Group* (OMG) and describes a generic model architecture. This architecture consists of four layers, whereas the the lowest layer represents the *reality*, i.e. the section of the real world that should be represented as a model. In case of EA management, this could be the the processes *Acquisition* and *Distribution*. This lowest layer is called M0. The layer above, called M1, stands for the actual *model* that abstracts this reality. In our example, this could be an UML class for the *process*, together with two instances for representing the real world objects. The third layer, M2, contains a *metamodel* that describes the concepts available in M1. It thus sets the constraints for what can be modeled. In this example case, it would contain the UML specification, in particular the concepts of a UML class and a UML instance. Finally, the topmost layer, called M3, contains the concepts that build the foundation for the UML specification. These concepts are called *meta-metamodel*. This layer therefore contains the concepts described in MOF. Furthermore, a meta-metamodel is able to describe itself, i.e. there are no more layers necessary to describe the concepts in this layer [OMG11; ISO12a]. This MOF architecture with the exemplified instances is illustrated in figure 4.1. Therein, the dashed lines denote the derivates over the distinct layers.

EMF provides an implementation of this model architecture, with *ECore* as the meta-metamodel (M3). Being based on this generic architecture, EMFStore supports custom EMF metamodels (M2) to be loaded in the repository. As this metamodel constrains the possible model elements, it can be used as the basis for a description of an enterprise architecture.

Regarding the versioning of the model elements, the concept of a *delta* describes the differences between two versions. Multiple approaches for the representation of the deltas have been proposed: In the *state-based* approach, only the state representation of the distinct versions is stored [KH10]. Deltas thus need to be derived based on a differencing algorithm that compares two state representations, i.e. a version and its successor. Using this approach, it can be impossible to recalculate the sequence of changes that resulted in the successor version. This is especially a problem when the changes of one operation are partially or completely masked by those of a later operation. The *change-based* approach in contrast records the changes on the model while they occur. As the sequence of changes is stored, the differences between two versions can easily be determined. EMFStore employs the *operation-based* approach [KHS09]. This approach builds on the change-based approach and represents the changes as transformation operations on a state. With this approach, a single transformation that results in multiple changes on the model, e.g. a refactoring, can be recorded as a single operation. The original semantic context of the change is therefore preserved. This has additional benefits, if two states of the model should be merged, and the operations recorded on one state therefore have to be replayed on the other state. The drawback of this approach is that the operations are dependent on the editor used, i.e. the versioning system has to be hooked into the editor actions. This results in a coupling of the editor with the versioning system and is therefore not applicable if arbitrary editors should be supported [Kög08]. Howewer, that is typically not the case in the context of EA management.

4.1.2. Evaluation

Evaluating EMFStore regarding its suitability as the technical foundation for the proposed EA-planning framework, the following requirements are supported.

Requirement R1 describes the need for a description language for states of the EA. This requirement is supported by EMFStore through its generic support of custom EMF metamodels and corresponding models. The needed concepts can therefore be described in the metamodel and the elements being saved as instances in the model. The requirements R2 to R4 demand support to designate a single state to either represent the current, the target, or a planned state of the EA. These three requirements can be realized as named branches in EMFStore, a feature that was added recently [Kög12]. The scheduling and rescheduling of planned states, as specified

in requirements R4 and R5 can either be implemented as a further information in the name of a branch, or can also be part of a custom metamodel. The revision of a state, as requested in requirement R6, is supported natively by the version control system. Requirement R7 specifies the need to be able to duplicate a whole and a part of an existing state in a new state. The duplication of a whole state is supported natively in EMFStore by the use of branching. To duplicate parts of states, a custom implementation is needed, e.g. with an appropriate metamodel design that allows to hide certain elements of the EA from the user. The consolidation of states according to requirement R8 is supported by the framework with the concept of a merge. No information could be found about the rejection of states, i.e. the deletion of a branch, described in requirement R9. This mechanism could nevertheless be part of the custom implementation on top of this technical framework. The arbitrary access specified in requirement R10 is supported natively by the version control system. The difference analysis, requested in requirement R11, is likewise supported. As EMFStore does not contain a tagging or a similar mechanism, the approval of versions is not natively supported. This requirement, specified in requirement R12, can nevertheless be implemented on top of the framework, or as a feature of the metamodel. Access control is only partially supported: EMFStore allows to restrict the access to the versioning server with an access control system that allows to specify users that are granted access to the server. However, distinct access rights as specified in requirement R13, are not featured and must be implemented on top of EMFStore.

An overview about this evaluation is given in table 4.1. Overall, EMFStore provides a broad support for the requirements defined in section 3.1. The requirements that could not be directly mapped to a feature of EMFStore can easily be implemented in the context of a custom client. EMFStore therefore is a suitable candidate for the technical foundation of the framework, proposed in chapter 3.

Table 4.1.: Support of requirements in EMFStore

Requirements		Supported	
R1	State description	✓	Custom EMF metamodel supported
R2	Current state	✓	Named branch
R3	Target state	✓	Named branch
R4	Planned state	✓	Named branch
R5	Rescheduling	✓	Named branch or custom metamodel
R6	Revision	✓	Version control
R7	Duplication	~	Parts of states not supported
R8	Consolidation	✓	Merging of branches
R9	Rejection	?	No information found
R10	Arbitrary access	✓	Version control
R11	Difference analysis	✓	Version control
R12	Approval	—	Custom implementation
R13	Access control	~	Only repository access

✓ full support, ~ partially support, — no support, ? unknown support

4.2. Model Versioning and Evolution

The second technical framework which might act as the foundation for an implementation of the EA planning framework presented in section 3.2 is described in the following. General information about the framework is given in section 4.2.1. In section 4.2.2, an evaluation regarding the requirements is provided.

4.2.1. General information

The *Model Versioning and Evolution* (MoVE)[3] project proposes a novel approach for model versioning and brings together various results of different model research areas. The project is homed at the *Quality Engineering Lab* of the University of Innsbruck. It presents multiple concepts for a model infrastructure and provides a prototypic framework as a realization.

The project proposes a *Living Models* paradigm [Bre10] to bring together dynamic evolving models in the area of software development, IT management, and system operations. Breu [Bre10] proposes ten principles, which the group considers crucial for a Living Models environment. These principles include *stakeholder-centric modeling environments*, which states the Living Models environment should support models

[3] URL: http://move.q-e.at

for various stakeholders with distinct goals and different requirements regarding the level of abstraction in the model. Each stakeholder can be assigned the *responsibility* to a domain of model elements. This principle requires various models of distinct areas to be connected, and the environment to provide specific views with regard to the individual concerns of the stakeholders. For this to work, the various models are required to be related by a *common system view* [BBL10]. According to this second principle, each model maintained in the repository is considered part of a common system model. This model is defined by a holistic *system metamodel* comprising the abstract metamodel elements and their relationships. This metamodel can be enriched by plugins to provide non-functional aspects, such as security related informations. Furthermore, the single model elements in the common system model can have *states*, that reflect certain aspects in its lifecycle. These states can be restricted by a state machine to only allow specific predefined evolutions. Changes to the model elements are perceived as events, which in turn can activate consecutive operations. This *change propagation* can e.g. be used to trigger state changes or to send a notification to the stakeholders [Tro10; TBL10].

A prototypic implementation of this Living Models paradigm is provided as an open-source project under the *Eclipse Public License* (EPL). MoVE is written in the Java programming language and is structured in multiple components inheriting a client/server-architecture. The system metamodel and the derived models are structured according to the MOF specification (cf. section 4.1.1) and are built upon the *Eclipse Modeling Framework* (EMF) and its *ECore* meta-metamodel. MoVE implements a state-based approach for change tracking in which, the model merging, conflict resolution, and difference analysis is based on EMFCompare[4]. The central versioning component of MoVE builds upon *Apache Subversion* (SVN)[5], an open-source file and folder oriented version control system. The integration into the MoVE components is implemented based on SVN hooks, which allow to trigger external programs and propagate information [BBL11; Kal10]. As SVN is in widespread use, client connectors for various platforms exist. A MoVE client can therefore build upon existing SVN connectors, but needs to extend this generic interface with custom logic for the use with the MoVE backend.

4.2.2. Evaluation

Regarding the suitability of MoVE as a foundation for the proposed EA-planning framework, the following requirements are supported:

The mechanism to describe a state of the EA, according to requirement R1, is provided by MoVE through its support of custom metamodels based on EMF. The

[4] URL: http://www.eclipse.org/emf/compare/

[5] URL: http://subversion.apache.org

requirements R2 to R4 describe the need to designate a state to either represent the current, a planned, or the target architecture. This requirement is supported by MoVE through its incorporation of the SVN system. With the help of SVN, a branch can be created and given a custom name. The (re-)scheduling of planned states, as requested in requirement R5, can equally be achieved as a part of the branch's name, or as SVN metadata on an element in the branch. Requirement R6 specifies the need of a versioning mechanism for the EA states. As the core feature, it is natively provided by the version control system. The duplication of states can be achieved by branching existing data. The stakeholder-specific views can be used to implement the mechanism to duplicate a part of a state. This requirement is specified as R7. The consolidation of distinct EA states, as requested in requirement R8, is only partially supported by MoVE: Although the system allows to merge branches, only state-based change tracking and thus merging is implemented. As described in section 4.1.1, this approach has drawbacks with graph-based data. Requirement R9 specifies the rejection of EA states. This requirements is supported through the ability of SVN to delete branches. The arbitrary access to versions is requested in requirement R10 and supported by MoVE through its versioning functionality. A method for the difference analysis is provided by MoVE through its incorporation of EMFCompare. This functionality is requested by requirement R11. One possibility to realize requirement R12, the approval of versions, in MoVE, is to use the SVN tagging mechanism. Using this mechanism, one can assign labels to branches. The second possibility is to use the state-mechanism of MoVE, to represent an approved-state in the lifecycle of the affected model elements. The access control feature specified in the last requirement (R13) is also supported: MoVE allows to specify roles with responsibilities for selected model domains, and also specify permissions considering change operations.

Table 4.2 summarizes this evaluation. Overall, MoVE features nearly every requirement without the need for much customization. The sole requirement that is marked as only partially supported is requirement R8, the consolidation of states. Although the corresponding functionality is technically supported by the tool, its implementation might rise a problem in the context of the models used in the discipline of EA management. MoVE is nevertheless a valid candidate for an implementation of the proposed framework.

Table 4.2.: Support of requirements in MoVE

Requirements		Supported	
R1	State description	✓	Custom EMF metamodel supported
R2	Current state	✓	Named branch
R3	Target state	✓	Named branch
R4	Planned state	✓	Named branch
R5	Rescheduling	✓	Branch metadata
R6	Revision	✓	Version control
R7	Duplication	✓	Stakeholder-specific views
R8	Consolidation	~	Merging of branches
R9	Rejection	✓	Deletion of branches
R10	Arbitrary access	✓	Version control
R11	Difference analysis	✓	Version control
R12	Approval	✓	Tagging
R13	Access control	✓	Domains and responsibilites

✓ full support, ~ partially support, — no support, ? unknown support

4.3. Comparison of results

In the preceding sections 4.1 and 4.2, two selected technical frameworks have been presented, that could serve as the foundation for an implementation of the proposed EA-planning framework. Both projects have been evaluated as suitable candidates for a potential implementation. Nevertheless are there considerable differences between the two projects.

First, there are organizational as well as popularity differences: The first presented technical framework, EMFStore, is a popular open-source model repository which has established itself outside of its original roots in Unicase [Brü+08] — integrated in the Eclipse development community and with commercial support available. MoVE in contrast is a relatively young project which has yet outgrow its university background. As such, there are significant developments around the tool, with the major migration from a REST-based interface [Fie00] to an SVN-backend only two years ago [BBL10; Kal10]. The stability of the tool regarding the implementation in a EA management software has therefore to be evaluated.

Second, there are technical differences: MoVE makes use of existing tools and libraries, such as EMFCompare and SVN, and integrates them in a customized package. EMFStore in contrast provides an integrated solution, covering the client and the server. Whether the flexibility of MoVE or the homogenous environment of EMFStore

is more suitable has to be decided depending in the context of the intended use.

Finally, both projects offer some unique advantages: The operation-based change recording in EMFStore can be a valuable feature if the need for complex merges is foreseeable. MoVE, on the other hand, provides a more extensive foundation with the Living Models paradigm: The triggering and change propagation mechanism as well as the stakeholder-specific views might both help to comprise neighbored functionalities not directly targeted by the proposed EA-planning framework.

Altogether, both tools might provide a suitable foundation for an implementation of the proposed EA-planning framework. A conclusive evaluation will have to include the concrete context of application to result in a final analysis.

5. Conclusion

To conclude this work, the central results are recapitulated and related questions, that arose during the research, are presented. In section 5.1, the precedent chapters are summarized and the central contributions are summed up. Finally, in section 5.2, the contributions of this work are critically reflected and a brief outlook on possible directions for future research is given.

5.1. Summary of results

In the preceding four chapters, the main objective of this work was approached stepwise. Each chapter contributed to the overall goal, defined as follows:

> **Research objective:** Design and describe a framework for the strategic planning of enterprise architectures.

The individual steps taken towards this research objective are described subsequently. Thereby, the single chapters are summarized and the key contributions are discussed.

Chapter 1 introduced the main motivation for this research and described the goal that was approached. In section 1.1, a brief introduction into the field of EA management was given and the research objective was motivated. In section 1.2, the research objective was defined and the research approach as well as the targeted questions were presented. In section 1.3, the transcription of the approach into the outline of the work was covered.

Chapter 2 gave an overview about related literature and discussed the key concepts introduced in several approaches. In section 2.1, the basis for the selection was presented and the selection of relevant approaches was described. In section 2.2, the selected eight approaches were introduced and the central concepts were described. In section 2.3, a terminological mapping was presented, that relates the common terms. Overall, this chapter contributed the answer to research question Q1, which is about the concepts regarding the strategic planning with EA models, found in state-of-the-art EA management approaches.

Chapter 3 presented the core contribution of this work. It introduced a novel framework for the strategic EA transformation. In section 3.1, the requirements

for the strategic planning with enterprise architectures were defined and discussed. Furthermore, a mapping of the identified 13 requirements with the analyzed EA management approaches was given. Following that, in section 3.2, the proposed framework for the strategic EA management was presented. In section 3.3, the applicability of the framework in practice was demonstrated based on exemplary use-cases. In section 3.4, the evaluation of the specified requirements and the proposed framework was described, based on expert interviews and an online survey. Overall, this chapter contributed to the research questions Q2 and Q3. The first part of this chapter answered which requirements regarding the planning with EA models exist, whereas the second part described a framework for the strategic planning of enterprise architectures.

Chapter 4 laid the foundation for subsequent research and evaluated technical frameworks regarding their suitability for an implementation of the proposed planning-framework. In section 4.1, the model repository *EMFStore* was discussed and subsequently evaluated. In section 4.2, the *MoVE* approach was analyzed. In the subsequent section 4.3, the results of both evaluations were compared and related to the field of EA management.

5.2. Critical reflection and future research

With the analysis of the selected state-of-the-art EA management approaches, the foundation for the following research has been established. In the context of this work, only eight approaches could be analyzed and only one literature overview could be selected as the foundation for the selection. In future research, one could base this analysis on multiple literature overviews and extend the number of selected approaches. In this way, the coverage of existing approaches could be improved and the expressiveness of the work could therefore be increased. Likewise, the mapping of terms would provide a more extensive overview about the terminological plurality in this field.

The justification of the requirements and the evaluation of the proposed framework have been provided through expert interviews and an online survey. Due to the time-constraints of this work, only three experts could be interviewed. Furthermore, only five valid responses could be received from practitioners in the online survey. As this limited feedback reduces the expressiveness of the evaluation, future research could approach a more extensive number of responses in both, the expert interviews and the online survey.

With the analysis of possible technical foundations for an implementation of the proposed planning-framework, an initial step towards an implementation of the framework in a software tool has been taken. As this evaluation was only partly

related with the core objective of this work — and its subdivision in the three research questions — only two frameworks could be evaluated. To extend this groundwork for a later implementation, a more comprehensive analysis of technical frameworks could be made.

Taking up on this preliminary work, a next step in the development of the proposed framework could be an implementation as a tool. Referencing the statement in one of the expert responses (cf. section 3.4.2), the framework only provides advantage in practice when integrated in a software tool. Besides practice, future research could also build on that to evaluate and refine the framework in real application.

A. Appendix: Survey

In this appendix, the online expert survey, conducted between November 26, 2012 and December 5, 2012, is presented. A description of this survey can be found in section 3.4.2. In the following, every question is presented, together with the consolidated answers.

A.1. Cover page

Collaborative strategic EA planning — a requirements prioritization survey

Dear industry expert,

Academia and practitioners consider Enterprise Architecture (EA) management to be a powerful method to align business and IT in order to gain strategic advantage over competitors. Therefore, it is essential to describe the status-quo and the intended target state of the EA, which allows for various planned (transition) states to be derived. These EA state descriptions build the foundation for the planning, communication and analysis of changes.

With the growing importance of EA management and it's strategic planning in recent years, the handling of different EA states is getting more and more important and sparked interest not only among practitioners and tool vendors, but also among academia. In our research, we first analyzed eight EA approaches and derived 13 requirements for a framework targeting the handling of states in the area of strategic EA planning. We then evaluated and refined these requirements based on expert interviews. Now, we need your valuable input to help us understand *your prioritization of these requirements for your EA planning activities.*

With your participation in this survey, you will gain an insight into our current research results on the one hand. On the other hand, you could influence through your prioritization the road map of EA management tool vendors.

The survey consists of 28 mostly closed questions. Its *completion takes less than 13 minutes.* Please fill out the survey until 5th of December, 2012. For any questions do not hesitate to contact us: philip.achenbach@tum.de.

Thank you for your interest and support in advance!

With kind regards,

Philip Achenbach

Technical University Munich (TUM)
Chair for Software Engineering for Business Information Systems (sebis)

A.2. Background information

Q1 In which country are you working?

5 × Germany

Q2 What industry branch are you working in?

1 × Manufacturing (e.g. automotive) 1 × Telecommunication 1 × Consulting 1 × Government 1 × Other: —

Q3 Which of the following describes your current professional occupation best?

3 × Enterprise architect 1 × Consultant 1 × IT architect

Q4 For how long have you already been working in the area of EAM?

2 × 1–5 years 2 × 6–10 years 1 × More than 10 years

Q5 For how long have you already been working in the area of strategic EA planning?

2 × 6–10 years 1 × Less than 1 year 1 × 1–5 years 1 × More than 10 years

Q6 Does your company apply an EAM framework?

3 × Yes 2 × No

Q7 Which EAM frameworks are applied?

1 × Strategic IT management of Hanschke, TOGAF 1 × TOGAF 2 × — 1 × ARIS, TOGAF, Other: NAF

Q8 Does your company use a EA tool?

4 × Yes 2 × No

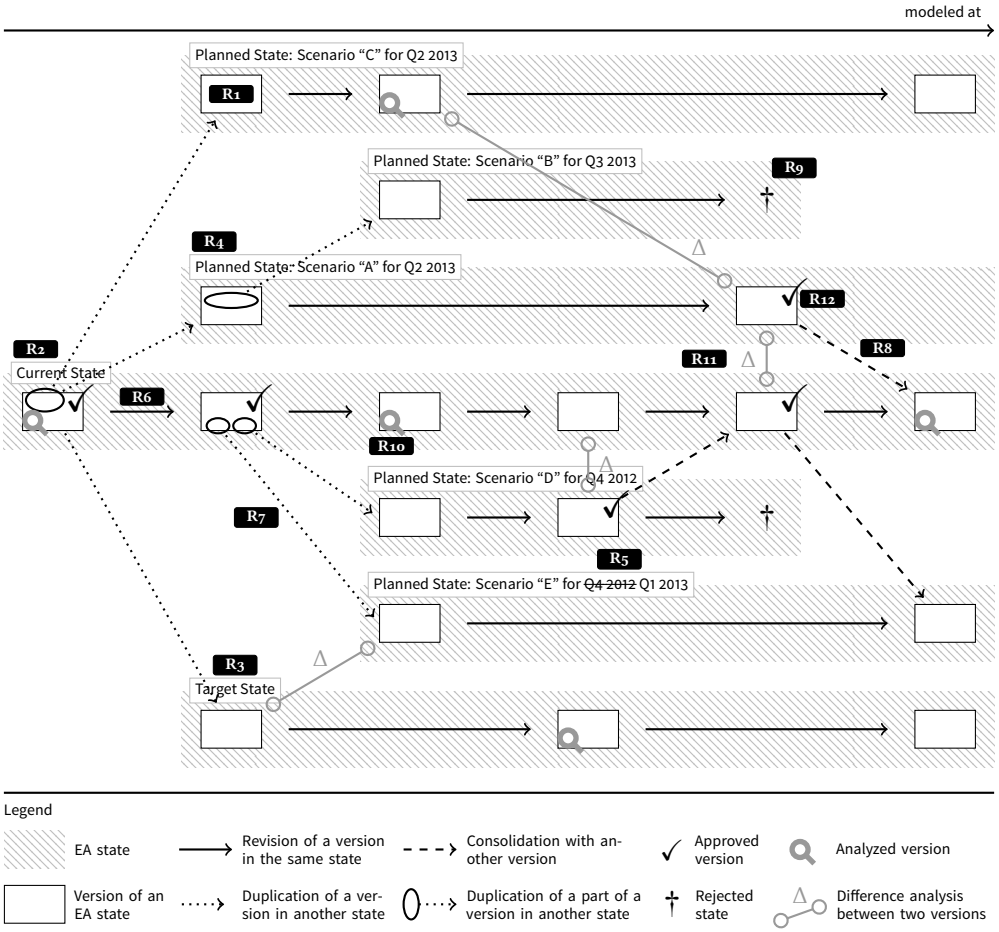
Q9 Which EA tools does your company use? Please provide the names of these tools.

1 × Troux, ARIS 1 × iteraplan 1 × RIS 2 × —

Q10 Which of the following states are considered in your companies EA planning?

2 × current, planned, target 2 × current, planned 1 × current, target

A.3. Requirements Prioritization



Q11 R1: The framework must provide a mechanism to describe multiple states of the EA. This description must contain all relevant elements, their relationships to each other, and their properties.

Do you agree that this requirement is important?

1 × Strongly agree 4 × Agree 0 × Neutral 0 × Disagree 0 × Strongly disagree

Q12 R2: The framework must provide a mechanism to mark a single description of a state as representing the current state of the EA. This state reflects the present situation of the enterprise.

Do you agree that this requirement is important?

2 × Strongly agree 3 × Agree 0 × Neutral 0 × Disagree 0 × Strongly disagree

Q13 R3: The framework must provide a mechanism to mark a single description of a state as being the intended future target state of the EA. This state represents the unscheduled long-term vision of the architecture.

Do you agree that this requirement is important?

0 × Strongly agree 4 × Agree 1 × Neutral 0 × Disagree 0 × Strongly disagree

Q14 R4: The framework must provide a mechanism to mark a description of a planned state as being intended to take effect at a given future point in time. There may be multiple planned states scheduled to be realized at the same time. These planned states guide the EA evolution from the current to the target state.

Do you agree that this requirement is important?

0 × Strongly agree 4 × Agree 1 × Neutral 0 × Disagree 0 × Strongly disagree

Q15 R5: The framework must provide a mechanism to reschedule a planned state. Therefore, the envisioned point of realization must be adaptable.

Do you agree that this requirement is important?

1 × Strongly agree 3 × Agree 0 × Neutral 1 × Disagree 0 × Strongly disagree

Q16 R6: The framework must provide a mechanism to revise a state. The result must be represented as a new version of the same state. Each version must be accessible independently.

Do you agree that this requirement is important?

2 × Strongly agree 3 × Agree 0 × Neutral 0 × Disagree 0 × Strongly disagree

Q17 R7: The framework must provide a mechanism to create a duplicate of (a part of) a state. Each duplicate must be represented as an own state.

Do you agree that this requirement is important?

0 × Strongly agree 3 × Agree 2 × Neutral 0 × Disagree 0 × Strongly disagree

Q18 R8: The framework must provide a mechanism to consolidate the descriptions of any two states into a new version of one of these states. The framework must support the selection of the state comprising the consolidation.

Do you agree that this requirement is important?

0 × Strongly agree 4 × Agree 1 × Neutral 0 × Disagree 0 × Strongly disagree

Q19 R9: The framework must provide a mechanism to reject a state. Consequently, it must ensure that rejected states can not be changed anymore.

Do you agree that this requirement is important?

0 × Strongly agree 2 × Agree 2 × Neutral 1 × Disagree 0 × Strongly disagree

Q20 R10: The framework must provide a mechanism to access any version for the purpose of analysis.

Do you agree that this requirement is important?

0 × Strongly agree 2 × Agree 0 × Neutral 2 × Disagree 1 × Strongly disagree

Q21 R11: The framework must provide a mechanism to determine differences between any two versions. Consequently, it must distinguish between new, revised, and rejected elements, relationships, and properties.

Do you agree that this requirement is important?

0 × Strongly agree 5 × Agree 0 × Neutral 0 × Disagree 0 × Strongly disagree

Q22 R12: The framework must provide a mechanism to mark certain versions as approved. There might be multiple approved versions at the same time.

Do you agree that this requirement is important?

0 × Strongly agree 3 × Agree 1 × Neutral 1 × Disagree 0 × Strongly disagree

Q23 R13: The framework must provide a mechanism to specify user access rights. Consequently, the mechanism must distinguish between reading and writing access, the duplication of states, their rejection and approval.

Do you agree that this requirement is important?

0 × Strongly agree 2 × Agree 1 × Neutral 1 × Disagree 1 × Strongly disagree

Q24 Are there any requirements mentioned that are dispensable regarding EA planning and should be removed? If yes, please provide a short justification for your decision.

5 × No

A.4. Comments and Feedback

Q25 Are any relevant requirements missing?

2 × Yes 3 × No

Q26 Please provide a comprehensive description of these missing requirements.

"All the requirements are worthless without the right processes and tools to manage the resulting models."

"I assume that when referring to EA there is a link to the strategy part of the enterprise (i.e. Biz–IT alignment need) as well as the link to the operational (and strategic) planning framework (i.e. portfolio management of services and projects) — in this bigger context EAM end up in providing max. value add"

Q27 Can you imagine that an implementation of this framework within an EA management tool would be useful for your work in the area of EA strategic planning?

5 × Yes

Q28 Are there any further thoughts you want to share regarding the strategic EA planning?

5 × —

Bibliography

[99] *GERAM: Generalised Enterprise Architecture Methodology*. IFIP–IFAC Task Force on Architectures for Enterprise Integration, Mar. 1999. URL: http://www.ict.griffith.edu.au/~bernus/taskforce/geram/versions/geram1-6-3/v1.6.3.html (visited on 09/24/2012).

[AG10a] Stephan Aier and Bettina Gleichauf. "Application of Enterprise Models for Engineering Enterprise Transformation." In: *Enterprise Modelling and Information Systems Architectures* 5 (1 July 2010): *Special Issue on Methodologies for Enterprise and Organisational Engineering*, pp. 58–75.

[AG10b] Stephan Aier and Bettina Gleichauf. "Applying Design Research Artifacts for Building Design Research Artifacts: A Process Model for Enterprise Architecture Planning." In: *Global Perspectives on Design Science Research*. Proceedings. 5th International Conference on Global Perspectives on Design Science Research (DESRIST) (St. Gallen, Switzerland, June 4–5, 2010). Ed. by Robert Winter, J. Leon Zhao, and Stephan Aier. Lecture Notes in Computer Science (LNCS) 6105. Berlin and Heidelberg, Germany: Springer, 2010, pp. 333–348. ISBN: 978-3-642-13335-0. DOI: 10.1007/978-3-642-13335-0_23.

[AG10c] Stephan Aier and Bettina Gleichauf. "Towards a Systematic Approach for Capturing Dynamic Transformation in Enterprise Models." In: *Proceedings of the 43rd Hawaii International Conference on System Sciences 2010*. 43rd Annual Hawaii International Conference on System Sciences (HICSS-43) (Koloa, Kauai, Hawaii, Jan. 5–8, 2010). Ed. by Ralph H. Sprague. Los Alamitos, CA, USA: IEEE Computer Society, Jan. 2010, pp. 1–10. ISBN: 978-0-7695-3869-3. DOI: 10.1109/HICSS.2010.404.

[Aie+09] Stephan Aier et al. "Complexity Levels of Representing Dynamics in EA Planning." In: *Advances in Enterprise Engineering III*. Proceedings of the 5th International Workshop CIAO! 2009 and 5th International Workshop EOMAS 2009. The 5th International Workshop on Cooperation & Interoperability – Architecture & Ontology (CIAO! 2009) in conjunction with the CAiSE'09 conference (Amsterdam, Netherlands, June 8–9, 2009). Ed.

by Antonia Albani et al. Lecture Notes in Business Information Processing (LNBIP) 34. Berlin and Heidelberg, Germany: Springer, 2009, pp. 55–69. ISBN: 978-3-642-01915-9. DOI: 10.1007/978-3-642-01915-9_5.

[AST10a] David Aveiro, A. Rito Silva, and José Tribolet. "Extending the Design and Engineering Methodology for Organizations with the Generation Operationalization and Discontinuation Organization." In: *Global Perspectives on Design Science Research*. 5th International Conference on Design Science Research in Information Systems and Technology (DESRIST 2010) (St. Gallen, Switzerland, June 4–5, 2010). Ed. by Robert Winter, J. Leon Zhao, and Stephan Aier. Lecture Notes in Computer Science (LNCS) 6105. Berlin and Heidelberg, Germany: Springer, 2010, pp. 226–241. ISBN: 978-3-642-13334-3. DOI: 10.1007/978-3-642-13335-0_16.

[AST10b] David Aveiro, A. Rito Silva, and José Tribolet. "Towards a GOD-theory for organizational engineering: continuously modeling the continuous (re)Generation, Operation and Deletion of the enterprise." In: *Proceedings of the 2010 ACM Symposium on Applied Computing*. 25th Symposium On Applied Computing (SAC'10) (Sierre, Switzerland, Mar. 23–26, 2010). New York, NY, USA: ACM, 2010, pp. 150–157. ISBN: 978-1-60558-639-7. DOI: 10.1145/1774088.1774118.

[BBL10] Michael Breu, Ruth Breu, and Sarah Löw. "Living on the MoVE: Towards an Architecture for a Living Models Infrastructure." In: *ICSEA 2010. The Fifth International Conference on Software Engineering Advances*. Proceedings. The Fifth International Conference on Software Engineering Advances (ICSEA) (Nice, France, Aug. 22–27, 2010). Ed. by Jon Hall et al. Los Alamitos, CA, USA: IEEE Computer Society, Nov. 1, 2010, pp. 290–295. ISBN: 978-0-7695-4144-0. DOI: 10.1109/ICSEA.2010.51.

[BBL11] Michael Breu, Ruth Breu, and Sarah Löw. "MoVEing Forward: Towards an Architecture and Processes for a Living Models Infrastructure." In: *International Journal On Advances in Life* 3.1 and 2 (Sept. 15, 2011), pp. 12–22. ISSN: 1942-2660.

[BNS03] Peter Bernus, Laszlo Nemes, and Günter Schmidt, eds. *Handbook on Enterprise Architecture*. International Handbooks on Information Systems. Berlin and Heidelberg, Germany: Springer, 2003. ISBN: 3-540-00343-6. DOI: 10.1007/978-3-540-24744-9.

[Bre10] Ruth Breu. "Ten Principles for Living Models — A Manifesto of Change-Driven Software Engineering." In: *CISIS 2010. Proceedings*. International Conference on Complex, Intelligent, and Software Intensive Systems. The 4th International Conference on Complex, Intelligent, and Software

Intensive Systems (CISIS) (Krakow, Poland, Feb. 15–18, 2010). Ed. by Leonard Barolli et al. Los Alamitos, CA, USA: IEEE Computer Society, Apr. 15, 2010, pp. 1–8. ISBN: 978-0-7695-3967-6. DOI: 10.1109/CISIS.2010.73.

[Bro+12] Petra Brosch et al. "An Introduction to Model Versioning." In: *Formal Methods for Model-Driven Engineering*. 12th International School on Formal Methods for the Design of Computer, Communication and Software Systems (SFM) (Bertinoro, Italy, June 18–23, 2012). Ed. by Marco Bernardo, Vittorio Cortellessa, and Alfonso Pierantonio. Lecture Notes in Computer Science (LNCS) 7320. Berlin and Heidelberg, Germany: Springer, 2012, pp. 336–398. ISBN: 978-3-642-30982-3. DOI: 10.1007/978-3-642-30982-3_10.

[Brü+08] Bernd Brügge et al. "Unicase – an Ecosystem for Unified Software Engineering Research Tools." In: 3rd IEEE International Conference on Global Software Engineering (ICGSE'08) (Bangalore, India, Aug. 17–20, 2008). 2008.

[BS11] Sabine Buckl and Christian M. Schweda. *On the State-of-the-Art in Enterprise Architecture Managemenet Literature*. Technical Report. Munich, Germany: Chair for Software Engineering of Business Information Systems (sebis), Technische Universität München, June 1, 2011.

[Buc+08a] Sabine Buckl et al. "An Information Model for Landscape Management – Discussing Temporality Aspects." In: *Service-Oriented Computing – ICSOC 2008 Workshops*. Third Workshop on Trends in Enterprise Architecture Research (TEAR 2008) (Sydney, Australia, Dec. 1, 2008). Ed. by George Feuerlicht and Winfried Lamersdorf. Lecture Notes in Computer Science (LNCS) 5472. Berlin and Heidelberg, Germany: Springer, 2008, pp. 363–374. ISBN: 978-3-642-01247-1. DOI: 10.1007/978-3-642-01247-1_37.

[Buc+08b] Sabine Buckl et al. *Enterprise Architecture Management Pattern Catalog*. Technical Report. Version 1.0. Munich, Germany: Chair for Software Engineering of Business Information Systems (sebis), Technische Universität München, Feb. 2008.

[Buc+08c] Sabine Buckl et al. "Enterprise Architecture Management Patterns – Exemplifying the Approach." In: *Twelfth IEEE International EDOC Enterprise Computing Conference*. Proceedings. 12th IEEE International Enterprise Distributed Object Computing Conference (EDOC'08) (Munich, Germany, Sept. 15–19, 2008). Los Alamitos, CA, USA: IEEE Computer So-

ciety, 2008, pp. 393–402. ISBN: 978-0-7695-3373-5. DOI: 10.1109/EDOC.2008.75.

[Buc+09a] Sabine Buckl et al. "An Information Model for Managed Application Landscape Evolution." In: *Journal of Enterprise Architecture (JEA)* (Feb. 2009), pp. 12–26.

[Buc+09b] Sabine Buckl et al. "Using Enterprise Architecture Management Patterns to complement TOGAF." In: *Thirteenth IEEE International EDOC Enterprise Computing Conference*. Proceedings. 13th IEEE International Enterprise Distributed Object Computing Conference (EDOC'09) (Auckland, New Zealand, Sept. 1–4, 2009). Los Alamitos, CA, USA: IEEE Computer Society, 2009, pp. 34–41. ISBN: 978-0-7695-3785-6. DOI: 10.1109/EDOC.2009.30.

[Buc+10a] Sabine Buckl et al. "Building Blocks for Enterprise Architecture Management Solutions." In: *Practice-Driven Research on Enterprise Transformation*. Second Working Conference, PRET 2010 (Delft, Netherlands, Nov. 11, 2010). Ed. by Frank Harmsen et al. Lecture Notes in Business Information Processing (LNBIP) 69. Berlin and Heidelberg, Germany: Springer, 2010, pp. 17–46. ISBN: 978-3-642-16770-6. DOI: 10.1007/978-3-642-16770-6_2.

[Buc+10b] Sabine Buckl et al. "Enterprise Architecture Management Patterns for Enterprise Architecture Visioning." In: *The Proceedings of EuroPLoP 2009*. 14th Annual European Conference on Pattern Languages of Programming (EuroPLoP 2009) (Irsee, Germany, July 8–12, 2009). Ed. by Allan Kelly and Michael Weiss. CEUR Workshop Proceedings 566. Mar. 2, 2010. URL: http://nbn-resolving.de/urn:nbn:de:0074-566-8.

[Buc+11] Sabine Buckl et al. "Modeling Enterprise Architecture Transformations." In: The International IFIP Working Conference on Enterprise Interoperability (IWEI 2011) (Stockholm, Schweden, Mar. 23–24, 2011). 2011.

[Buc11] Sabine Buckl. "Developing Organization-Specific Enterprise Architecture Management Functions Using a Method Base." Ph.D. thesis. Munich, Germany: Technische Universität München, May 10, 2011. URL: http://nbn-resolving.de/urn/resolver.pl?urn:nbn:de:bvb:91-diss-20110510-1069959-1-8.

[CFP11] Ben Collins-Sussman, Brian W. Fitzpatrick, and C. Michael Pilato. *Version Control with Subversion. The Standard in Open Source Version Control*. For Subversion 1.7. 2011. URL: http://svnbook.red-bean.com/en/1.7/index.html (visited on 12/27/2012).

[Die06] Jan L. G. Dietz. *Enterprise Ontology. Theory and Methodolody*. Berlin and Heidelberg, Germany: Springer, 2006. ISBN: 978-3-540-29169-5. DOI: 10.1007/3-540-33149-2.

[Die99] Jan L. G. Dietz. "Understanding and Modelling Business Processes with DEMO." In: *Conceptual Modeling – ER '99*. 18th International Conference on Conceptual Modeling (Paris, France, Nov. 15–18, 1999). Ed. by Jacky Akoka et al. Lecture Notes in Computer Science (LNCS) 1728. Berlin and Heidelberg, Germany: Springer, 1999, pp. 767–767. ISBN: 978-3-540-47866-9. DOI: 10.1007/3-540-47866-3_13.

[Ern08] Alexander M. Ernst. "Enterprise Architecture Management Patterns." In: *Proceedings of the 15th Conference on Pattern Languages of Programs*. 15th Conference on Pattern Languages of Programs (PLoP'08) (Nashville, Tennessee, USA, Oct. 18–20, 2008). New York, NY, USA: ACM, 2008. ISBN: 978-1-60558-151-4. DOI: 10.1145/1753196.1753205.

[Ern10] Alexander M. Ernst. "A Pattern-based Approach to Enterprise Architecture Management." Ph.D. thesis. Munich, Germany: Technische Universität München, Apr. 7, 2010. URL: http://nbn-resolving.de/urn/resolver.pl?urn:nbn:de:bvb:91-diss-20100407-808722-1-6.

[Fie00] Roy Thomas Fielding. "Architectural Styles and the Design of Network-based Software Architectures." Ph.D. thesis. Irvine, CA, USA: University of California, Irvine, 2000.

[Han10a] Inge Hanschke. *Strategic IT Management. A Toolkit for Enterprise Architecture Management*. Berlin and Heidelberg, Germany: Springer, 2010. ISBN: 978-3-642-05033-6. DOI: 10.1007/978-3-642-05034-3.

[Han10b] Inge Hanschke. *Strategisches Management der IT-Landschaft. Ein praktischer Leitfaden für das Enterprise Architecture Management*. German. 2nd ed. München, Germany: Carl Hanser Verlag, July 2010. ISBN: 978-3-446-42257-5.

[Hel12] Jonas Helming. *EMFStore Release 0.9.0*. EclipseSource. May 29, 2012. URL: http://eclipsesource.com/blogs/2012/05/29/emfstore-release-0-9-0/ (visited on 01/10/2013).

[HW08] Martin Hafner and Robert Winter. "Processes for Enterprise Application Architecture Management." In: *Proceedings of the 41st Hawaii International Conference on System Sciences 2008*. 41st Hawaii International Conference on System Sciences (HICSS-41) (Waikoloa, Big Island, Hawaii,

Jan. 7–10, 2008). Los Alamitos, CA, USA: IEEE Computer Society, 2008. DOI: 10.1109/HICSS.2008.362.

[ISO] International Organization for Standardization. *ISO/DIS 19439*. Enterprise integration – Framework for enterprise modelling. Geneva, Switzerland: International Organization for Standardization.

[ISO00] International Organization for Standardization. *ISO 15704:2000*. Industrial automation systems – Requirements for enterprise-reference architectures and methodologies. Geneva, Switzerland: International Organization for Standardization, 2000.

[ISO07] International Organization for Standardization. *ISO/IEC 42010:2007*. Systems and software engineering – Recommended practice for architectural description of software-intensive systems. Geneva, Switzerland: International Organization for Standardization, 2007.

[ISO12a] International Organization for Standardization. *ISO/IEC 19505-1:2012. formal/2012-05-06*. Information technology – Object Management Group Unified Modeling Language (OMG UML), Infrastructure. Geneva, Switzerland: International Organization for Standardization, Apr. 2012.

[ISO12b] International Organization for Standardization. *ISO/IEC 19505-2:2012. formal/2012-05-07*. Information technology – Object Management Group Unified Modeling Language (OMG UML), Superstructure. Geneva, Switzerland: International Organization for Standardization, Apr. 2012.

[ISO98] International Organization for Standardization. *ISO/IEC 10746-1:1998*. Information technology – Open Distributed Processing – Reference model: Overview. Geneva, Switzerland: International Organization for Standardization, Dec. 15, 1998.

[Jon+03] Henk Jonkers et al. "Towards a Language for Coherent Enterprise Architecture Descriptions." In: *Seventh IEEE International Enterprise Distributed Object Computing Conference*. Proceedings. 7th IEEE International Enterprise Distributed Object Computing Conference (EDOC'03) (Brisbane, Queensland, Australia, Sept. 16–19, 2003). Los Alamitos, CA, USA: IEEE Computer Society, Sept. 2003, pp. 28–37. ISBN: 0-7695-1994-6. DOI: 10.1109/EDOC.2003.1233835.

[Jon+10] Henk Jonkers et al. *ArchiMate Extension for Modeling TOGAF's Implementation and Migration phases*. White Paper. San Francisco, CA, USA: The Open Group, Oct. 2010.

[JPT09a] Henk Jonkers, Erik Proper, and Mike Turner. *TOGAF 9 and ArchiMate 1.0*. White Paper. San Francisco, CA, USA: The Open Group, Nov. 2009.

[JPT09b] Henk Jonkers, Erik Proper, and Mike Turner. *TOGAF and ArchiMate: A Future Together. A Vision for Convergence & Co-Existence*. White Paper. San Francisco, CA, USA: The Open Group, Nov. 2009.

[Kal10] Philipp Kalb. "UML Model Merging in the Context of a Model Evolution Engine." Master's thesis. Innsbruck, Austria: Leopold-Franzens-University Innsbruck, Sept. 2010.

[KH10] Maximilian Kögel and Jonas Helming. "EMFStore: a model repository for EMF models." In: *Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering*. ACM/IEEE 32nd International Conference on Software Engineering (ICSE) (Cape Town, South Africa, May 2–8, 2010). Vol. 2. New York, NY, USA: ACM, 2010, pp. 307–308. ISBN: 978-1-60558-719-6. DOI: 10.1145/1810295.1810364.

[KH12] Maximilian Kögel and Jonas Helming. *Tutorial – EMFStore – A Model Repository. Share and version Java entities with EMF and EMFStore*. May 30, 2012. URL: http://jaxenter.com/tutorial-emfstore-a-model-repository-42956.html (visited on 01/10/2013).

[KHS09] Maximilian Kögel, Jonas Helming, and Stephan Seyboth. "Operation-based conflict detection and resolution." In: *Proceedings of the 2009 ICSE Workshop on Comparison and Versioning of Software Models*. International Workshop on Comparison and Versioning of Software Models (CVSM 2009) (Vancouver, Canada, May 17, 2009). Los Alamitos, CA, USA: IEEE Computer Society, 2009, pp. 43–48. ISBN: 978-1-4244-3714-6. DOI: 10.1109/CVSM.2009.5071721.

[Kög08] Maximilian Kögel. "Towards software configuration management for unified models." In: *Proceedings of the 2008 ICSE Workshop on Comparison and Versioning of Software Models*. International Workshop on Comparison and Versioning of Software Models (CVSM 2008) (Leipzig, Germany, May 17, 2008). New York, NY, USA: ACM, 2008, pp. 19–24. ISBN: 978-1-60558-045-6. DOI: 10.1145/1370152.1370158.

[Kög12] Maximilian Kögel. *Branching Support for the EMFStore Model Repository*. EclipseSource. July 23, 2012. URL: http://eclipsesource.com/blogs/2012/07/23/branching-support-for-the-emfstore-model-repository/ (visited on 01/10/2013).

[Lik32] Rensis Likert. "A technique for the measurement of attitudes." In: *Archives of Psychology* 22 (140 1932), pp. 1–55.

[LMW05a] Josef Lankes, Florian Matthes, and André Wittenburg. "Architekturbeschreibung von Anwendungslandschaften: Softwarekartographie und IEEE Std 1471-2000." German. In: *Software Engineering 2005*. (Essen, Germany, Mar. 8–11, 2005). Ed. by Peter Liggesmeyer, Klaus Pohl, and Michael Goedicke. Lecture Notes in Informatics (LNI) 64. Bonn, Germany: Gesellschaft für Informatik (GI), 2005, pp. 43–54. ISBN: 3-88579-393-8.

[LMW05b] Josef Lankes, Florian Matthes, and André Wittenburg. "Softwarekartographie: Systematische Darstellung von Anwendungslandschaften." German. In: *Wirtschaftsinformatik 2005. eEconomy, eGovernment, eSociety*. 7. Internationale Tagung Wirtschaftsinformatik (Bamberg, Germany, Mar. 22–25, 2005). Ed. by Otto K. Ferstl et al. Heidelberg: Physica Verlag, 2005, pp. 1443–1462. ISBN: 978-3-7908-1574-0. DOI: 10.1007/3-7908-1624-8_76.

[Lto4] Marc Lankhorst and the ArchiMate team. *ArchiMate Language Primer. Introduction to the ArchiMate Modelling Language for Enterprise Architecture*. Version 1.0. Enschede: Telematica Instituut, Aug. 26, 2004.

[LW04] Lam-Son Lê and Alain Wegmann. *Meta-model for Object-Oriented Hierarchical Systems*. Technical Report. Lausanne, Switzerland: Laboratory of Systemic Modeling, Swiss Federal Institute of Technology, Lausanne, 2004.

[LW05] Lam-Son Lê and Alain Wegmann. "Definition of an Object-Oriented Modeling Language for Enterprise Architecture." In: *Proceedings of the 38th Hawaii International Conference on System Sciences (HICSS-38)*. 38th Hawaii International Conference on System Sciences (HICSS-38) (Waikoloa, Hawaii, Jan. 3–6, 2005). Los Alamitos, CA, USA: IEEE Computer Society, Jan. 2005, 222a. ISBN: 0-7695-2268-8. DOI: 10.1109/HICSS.2005.186.

[LW06] Lam-Son Lê and Alain Wegmann. "SeamCAD: Object-Oriented Modeling Tool for Hierarchical Systems in Enterprise Architecture." In: *Proceedings of the 39th Hawaii International Conference on System Sciences (HICSS-39)*. 39th Hawaii International Conference on System Sciences (HICSS-39) (Koloa, Kauai, Hawaii, Jan. 4–7, 2006). Vol. 8. Los Alamitos, CA, USA: IEEE Computer Society, Jan. 2006, p. 179c. ISBN: 0-7695-2507-5. DOI: 10.1109/HICSS.2006.428.

[Mat+08] Florian Matthes et al. *Enterprise Architecture Management Tool Survey 2008*. Munich, Germany: Chair for Software Engineering of Business Information Systems (sebis), Technische Universität München, 2008.

[Mat+12] Florian Matthes et al. *EAM KPI Catalog*. Technical Report. Version 1.0. Munich, Germany: Chair for Software Engineering of Business Information Systems (sebis), Technische Universität München, 2012.

[MW04] Florian Matthes and André Wittenburg. "Softwarekartographie: Visualisierung von Anwendungslandschaften und ihrer Schnittstellen." German. In: *Informatik 2004. Informatik verbindet.* 34. Jahrestagung der Gesellschaft für Informatik e.V. (Ulm, Germany, Sept. 10–24, 2004). Ed. by Peter Dadam and Manfred Reichert. Vol. 2. Lecture Notes in Informatics (LNI) 51. Bonn, Germany: Gesellschaft für Informatik, 2004, pp. 71–75. ISBN: 3-88579-380-6.

[MWF08] Stephan Murer, Carl Worms, and Frank J. Furrer. "Managed Evolution. Nachhaltige Weiterentwicklung großer Systeme." German. In: *Informatik-Spektrum* 31 (6 2008): *Sonderheft: Management großer Systeme*, pp. 537–547. ISSN: 0170-6012. DOI: 10.1007/s00287-008-0290-9.

[Myk+11] Mariana Mykhashchuk et al. "Charting the landscape of enterprise architecture management. An extensive literature analysis." In: *Wirtschaftsinformatik Proceedings 2011.* 10. Internationale Tagung Wirtschaftsinformatik (Zürich, Switzerland, Feb. 16–18, 2011). 2011.

[OG09] The Open Group. *ArchiMate 1.0 Specification*. Reading, United Kingdom: The Open Group, Feb. 2009. URL: http://www.opengroup.org/archimate/doc/ts_archimate/ (visited on 09/24/2012).

[OG11] The Open Group. *TOGAF Version 9.1*. Reading, United Kingdom: The Open Group, 2011. URL: http://pubs.opengroup.org/architecture/togaf9-doc/arch/ (visited on 09/24/2012).

[OMG11] Object Management Group. *OMG Meta Object Facility (MOF) Core Specification*. Version 2.4.1. Needham, MA, USA: Object Management Group, Aug. 2011. URL: http://www.omg.org/spec/MOF/2.4.1.

[QEJ10] Dick Quartel, Wilco Engelsman, and Henk Jonkers. *ArchiMate Extension for Modeling and Managing Motivation, Principles, and Requirements in TOGAF*. White Paper. The Open Group, Oct. 2010.

[RW06] Irina Rychkova and Alain Wegmann. "A Method for Functional Alignment Verification in Hierarchical Enterprise Models." In: *Proceedings of the CAISE'06 Workshop on Business/IT Alignment and Interoperability (BUSITAL'06).* Business/IT Alignment and Interoperability (BUSITAL'06) (Luxemburg, June 5–9, 2006). Ed. by Yves Pigneur and Carson Woo. CEUR Workshop Proceedings 237. 2006. URL: http://nbn-resolving.de/urn:nbn:de:0074-237-2.

[RWB03] Irina Rychkova, Alain Wegmann, and Pavel Balabko. "Operational ASM Semantics behind Graphical SEAM Notation." In: *FMOODS/DAIS 2003 Student Workshop. Proceedings*. PhD Student Workshop on Formal Methods for Open Object-based Distributed Systems (FMOODS) and Distributed Applications and Interoperable Systems (DAIS) (Paris, France, Nov. 18, 2003). Paris, France, 2003, pp. 10–19.

[Sch11] Christian M. Schweda. "Development of Organization-Specific Enterprise Architecture Modeling Languages Using Building Blocks." Ph.D. thesis. Munich, Germany: Technische Universität München, July 28, 2011. URL: http://nbn-resolving.de/urn/resolver.pl?urn:nbn:de:bvb:91-diss-20110728-1072071-1-0.

[seb05] Chair for Software Engineering of Business Information Systems (sebis). *Enterprise Architecture Management Tool Survey 2005*. Technical Report. Munich, Germany: Chair for Software Engineering of Business Information Systems (sebis), Technische Universität München, 2005.

[TBL10] Thomas Trojer, Michael Breu, and Sarah Löw. "Change-driven Model Evolution for Living Models." In: 3rd Workshop on Model-Driven Tool & Process Integration (MDTPI) (Paris, France, June 16, 2010). 2010.

[Tro10] Thomas Trojer. "A Cooperative Platform supporting the Evolution of Heterogeneous Models." Master's thesis. Innsbruck, Austria: Leopold-Franzens-University Innsbruck, June 21, 2010.

[Vas+01] André Vasconcelos et al. "A framework for modeling strategy, business processes and information systems." In: *Fifth IEEE International Enterprise Distributed Object Computing Conference*. Proceedings. 5th IEEE International Enterprise Distributed Object Computing Conference (EDOC'01) (Seattle, WA, USA, Sept. 4–7, 2001). Los Alamitos, CA, USA: IEEE Computer Society, 2001, pp. 69–80. ISBN: 0-7695-1345-X. DOI: 10.1109/EDOC.2001.950424.

[VST03] André Vasconcelos, Pedro Sousa, and José Tribolet. "Information System Architectures: Representation, Planning and Evaluation." In: *Journal on Systemics, Cybernetics and Informatics (JSCI)* 1.6 (2003), pp. 78–84.

[Weg+07] Alain Wegmann et al. "Enterprise Modeling Using the Foundation Concepts of the RM-ODP ISO/ITU Standard." In: *Information Systems and E-Business Management* 5 (4 2007), pp. 397–413. ISSN: 1617-9846. DOI: 10.1007/s10257-007-0051-3.

[Weg03] Alain Wegmann. "On the Systemic Enterprise Architecture Methodology (SEAM)." In: *Proceedings of the 5th International Conference on Enterprise Information Systems*. Information Systems Analysis and Specification. 5th International Conference on Enterprise Information Systems (ICEIS 2003) (Angers, France, Apr. 23–26, 2003). Vol. 3. EPFL/LAMS & ICEIS, 2003, pp. 483–490.

[WF07] Robert Winter and Ronny Fischer. "Essential Layers, Artifacts, and Dependencies of Enterprise Architecture." In: *Journal of Enterprise Architecture* 2 (3 May 2007).