Complementing The Open Group Architecture Framework with Best Practice Solution Building Blocks

Sabine Buckl, Thomas Dierl, Florian Matthes, Christian M. Schweda
Chair for Software Engineering for Business Information Systems (sebis)
Technische Universität München, 85748 Garching, Germany
{sabine.buckl, dierl, matthes, schweda}@in.tum.de

Abstract

The design of an enterprise architecture (EA) management function suitable for an organization is no easy to accomplish task. Various frameworks as well as EA management tools exist, which promise to deliver guidance for performing EA management. Nevertheless, the approaches presented stay either on a level too abstract to provide realization support or are far too general neglecting organization-specific EA related goals, concerns, and contexts.

In this article, we propose a combined approach, which complements the abstract architecture development method of TOGAF with problem-specific best practice solution building blocks of the Enterprise Architecture Management Pattern Catalog of TU München. Increasing homogeneity of applications is used as an expository example to illustrate the approach.

1. Introduction and motivation

In the last decade several approaches to enterprise architecture (EA) management have been developed of which some provide a holistic and general view on the subject, while others make an in-depth dive into specific properties of EA management. An organization willing to use the prescriptions of these approaches to design an organization-specific EA management function is hence likely to run into difficulties, caused by the dichotomy of abstraction levels. Put in other words, while the embracing and often monolithic approaches provide a big picture of what EA management is meant to consist of remaining on an abstract level of description and presentation, the detailed approaches actually provide solutions for specific EA management problems and can often be used directly to address these problems but at the same time more often than not fail to provide an overall picture of what EA management consist of. The inherent complexity of EA management as reflected in the management subject as well as in the plurality of EA-relevant goals and stakeholders nevertheless calls for a structured approach for designing, introducing, and evolving an organization-specific EA management function.

In this article, we describe such an approach, which proposes to combine the prescriptions of two prominent EA management approaches, namely The Open Group Architecture Framework (TOGAF) 9 [27] and the enterprise architecture management pattern catalog [2]. Preparing our exposition of how patterns can be used to complement TOGAF with appropriate and building blocks, we describe TOGAF and the pattern catalog in Section 2. Section 3 shows how these two approaches can be combined to complement each other and Section 4 exemplifies this combination by an application example for standardization management. The article is concluded by a critical reflection and a short outlook in Section 5.

2. Prominent approaches to EA management

Various approaches to EA management have been proposed in the last years, originating from governmental institutions (see e.g. [9], [10]), standardization bodies (see e.g. [27]), academia (see [2], [21], [23]), and practitioners (see e.g. [11], [12], [19]). According to [6] two of the most promising approaches are the TOGAF framework [27] and the EAM pattern approach [2]. The latter two approaches to EA management are shortly introduced in the following to lay the basis for their integration, while in depth considerations and enhancement potentials are postponed to Section 3.

The Open Group Architecture Framework (TOGAF)

In February 2009 version 9 of TOGAF [27] superseded version 8.1.1. The new version introduced some additional features to support the design of organization-specific EA management approaches,

such as a **modular structure** to improve usability and incremental adoption, a **content framework** for improving consistency throughout the created deliverables, **extended guidance**, as well as **architectural styles**.

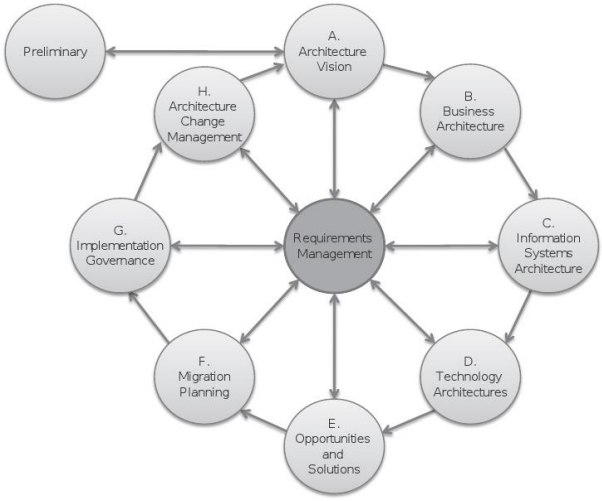

Figure 1: The architecture development method

Central constituent of TOGAF is the *architecture development method* (*ADM)*, which describes nine different phases of EA development (see Figure 1). These phases constitute the generic reference method for performing EA management and should – according to the prescriptions of the framework – be adapted to different *process styles* as well as to the *organization levels*, they are intended to be applied at. The corresponding adaptation mechanisms are part of the *ADM guidelines & techniques* section of the framework, although the therein presented mechanisms are limited in several ways, as e.g. different ways to organize the organization's IT function [26] are not alluded to.

The *content framework* of TOGAF introduces the content meta-model, i.e. an information model, for describing the EA of a using organization. This information model contains a large number of concepts, e.g. business application or goal, which should be documented. These concepts are further complemented with properties and interconnected via a set of predefined relationships. Accounting for a modular introduction of TOGAF, the framework further describes a set of *extensions* for the content framework, e.g. for introducing goal modeling into the information model. Nevertheless, the prescriptions of the content framework stay fairly abstract and do neither provide information on model constraints, as multiplicities nor on datatypes.

Enterprise Architecture Management Pattern Catalog (EAMPC)

Documenting proven-practice solutions to recurring problems in a specific context by so called patterns has initially been introduced by [1] in the field of architecture. Computer science has adopted this approach in the field of software engineering [15] and software architecture [8]. EAM patterns, as proposed in [14], provide general, reusable solutions to common problems in EA management, in a given context, identifying driving forces, denoting known usages, and consequences. Thus, EAM patterns are descriptions of a real world solution gained from observation.

The pattern-based approach to EA management has been developed to address typical problems of existing EA management approaches like too abstract guidelines, which lack appropriate guidance to be used in practice, or monolithic approaches pursuing an all or nothing approach neglecting the specific demands of an enterprise. An initial set of pattern has been collected from literature and practice, and has been evaluated in an extensive survey, resulting in version 1.0 of the EA management pattern catalog containing 120 EAM patterns.

Four types of patterns for EA management have been identified. Methodology Patterns (M-Pattern) define steps to be taken in order to address a given problem (also known as concern). Viewpoint Patterns (V-Pattern) provide a language used by one or more M-Patterns and thus propose ways to present data stored according to one or more information model patterns. Information Model Patterns (I-Pattern) supply an underlying model for the data visualized in one or more V-Patterns. In contrast to those three EAM patterns types, Anti-Pattern for EA management document solutions, which have proven not to work in order to prevent blind alleys. Patterns of all the aforementioned types form a pattern language for EA management, which is continuously improved and extended.

The EAM patterns thereby expatiate a twofold nature: firstly, they describe reusable solutions for common EA management problems, observed in practice. Secondly, they form building blocks of an organization-specific EA management function. To facilitate the design of such a management function, the pattern catalog is grounded in a consistent terminology and the contained information is structured and organized to simplify the selection, adaptation, and integration of patterns. The overall pattern approach focuses on addressing specific concerns and does not build an all embracing model that is meant to be suitable for every organization. In this way giant and monolithic process models as well as information models are avoided and replaced by problem-specific building blocks, only containing the information that is needed for the specific purpose.

This reduces the effort that has to be invested in collecting and maintaining the information to initially fill the repository and reduce the maintenance effort in the future.

TOGAF and the EAM pattern catalog both provide means to develop organization-specific EA management functions. As TOGAF explicitly states the ADM "complements and can be used in conjunction with other frameworks" [27], the subsequent section describes how the two approaches can be combined to complement the benefits of the ADM providing an overall picture with the advantages of the EAMPC, i.e. the problem specificity tailored to the needs of the using organization.

3. Complementing TOGAF with the EAMPC

The EAMPC provides best-practices for addressing typical EA management problems. Each best-practice brings along a specific process description on how to be performed, whereas the EAMPC does not provide a comprehensive EA management process model. Due to this specificity of the approach, the EAMPC does not provide an overall process description. In contrast, TOGAF provides a general process model - the ADM, which details on a sequence of phases to develop and evolve an EA. The distinct phases of the ADM are further detailed with a description of the *objectives*, an overview about the pursued approach, the required *inputs*, the abstract *steps* to be executed, and the resulting *outputs* of the phase, which might serve as input for the next phase of the ADM cycle.

Subsequently, we detail on the single phases of the ADM cycle, give a short overview on the abstract steps conducted in each phase, and provide indications how the EAMPC can be used to complement the general ADM of TOGAF with best-practice solutions gathered from practice and academia. In the subsequent discussions, we will see that not for all phases of the ADM best-practice solutions have been documented yet. While this at first sight may seem as a drawback of the presented approach, it in fact is a direct consequence of the evolutionary nature of the EAMPC. The catalog is continually improved and extended with practice-proven solutions. In combining the two approaches, we further have to bridge a gap in respect to the used terminology. Although both approaches, TOGAF and the EAMPC, centrally rely on the terminological basis of the ISO Std 42010 [17], the used terminology differs.

Preliminary phase: The TOGAF ADM cycle starts with the *Preliminary* phase, in which the EA management is prepared and initialized. Typical tasks executed in this phase include the definition and establishment of the EA management team, the selection and roll-out of supporting tools, as well as the definition of architecture guidelines and principles.

Architecture vision phase: After the preparation and initialization activities are performed, the scope of the EA management endeavor is defined. A core objective of this phase is to identify relevant stakeholders and their concerns. While TOGAF details on the management of stakeholders and makes categories of stakeholders explicit, e.g. *executives*, *line manager*, and *business process experts for the project organization*, no procedure how to identify relevant concerns is given. Based on the stakeholders and concerns identified a high-level architecture vision of the enterprise is developed.

In this phase the EAMPC can be used to support the identification of relevant concerns. It explicitly lists typical concerns in the context of EA management. These concerns can be provided to the stakeholders identified according to TOGAF. The stakeholders then select and prioritize the concerns to be addressed.

Business architecture, information systems architecture and technology architecture phase: Based on the architecture vision designed in the preceding phase, the corresponding business, information systems, and technology architectures are developed in three subsequent steps. The fundamental method used is thereby very similar: Initially, the current state of the architecture on the corresponding architectural level (business, information systems, or technology) is described. Based on this state, a target state of the architecture is developed. Complementing, gap analyses are performed to evaluate the differences between the current and the target states of the EA.

These phases build generic processes of architecture development and documentation and can be operationalized using EAM patterns. Thereby, the phases are made more concrete in respect to the specific requirements of the using organization as follows:

- *Identify EA concepts to be documented* – TOGAF does not explicitly account for the architecture information that should be documented in the current state or be designed in the context of the target state development. More precisely, TOGAF's content metamodel only gives a general overview on possibly relevant information, while in contrast the importance of gathering only the necessary information to avoid gratuitous effort is only referred to in the following advice: *Gather and analyze only that information that allows informed decisions to be made relevant to the scope of the architecture effort.* [27]. Put in

other words, TOGAF describes the need to adapt the content metamodel to the specific concerns of the using organization but does not detail on how to do this. The *EAMPC* can be used here to derive the required information from the concerns identified in the *Architecture Vision* phase. To do so, the I-Patterns related to the selected concerns are revisited and integrated into a concern-specific EA information model. The terminology provided by TOGAF's content metamodel may be used to further adapt this information model.

- *Determine overall documentation and development process* – Different methods for documenting the current state of the EA as well as for developing a target state architecture are used in practice. Quite a couple of such methods are documented in the M-Patterns of the EAMPC, of which an organization can choose the most appropriate ones. Exemplarily, an organization can decide to use the M-Patterns contributed by Moser et al. [25] for documenting the current state of the EA.
- *Identify required visualizations* – The importance of visualizations for communicating the EA to the corresponding stakeholders is alluded to as part of TOGAF. In response, the framework textually describes different types of visualizations but abstains from detailing the information that is necessary to create these visualizations. The V-Patterns contained in the pattern catalog describe practice-proven architectural visualizations, i.e. their underlying viewpoints, and explicitly reference to the corresponding information models and architectural concerns.

Opportunities and Solutions phase: Based on the identified gaps between current and target state of the EA, projects and programs to transform the architecture are derived. Along these projects also intermediary planned states of the EA are devised. As part of this multi-project planning, dependencies between the different projects are analyzed and overall consistency of the architectural roadmap is ensured.

Manifold patterns in the EAMPC target the topics of roadmapping and dependency analysis. To name just a prominent example, the pattern presented by of Ernst and Schneider [13] should be named. This pattern describes a practice-proven method for analyzing architecture-mediated project dependencies to avoid project interferences in EA transformation.

Migration Planning phase: The intermediary planned states for the EA are consolidated into a migration plan, which is complemented with a business value assignment for each project as well as a prioritization of the projects.

The EAMPC can support this phase with visualizations showing the aspect of time-dependency arising in the context of migration planning. This leads to certain demands regarding an information model and the visualizations used in this phase [3,13].

Implementation Governance phase: The projects selected for realization are executed and their execution is complemented with additional guidance from an EA management perspective. Especially architectural reviews are means to be employed in this phase to assure that project execution contributes to a consolidated architectural state.

Buckl et al. [4] proposed a method for performing project surveillance in order to determine, which projects are in need of in-depth architectural reviews and assistance during implementation. While no actually formulated using the typical form of an M-pattern, the method nevertheless describes a practice proven technique to selectively provide architectural guidance, where necessary. Further, implementation aspects of concerns are addressed in the EAMPC as part of the M-patterns, which give indications on implementation details, opportunities and impediments.

Architecture Change Management phase: Changes made to the architecture are assessed and the achievement of the goals pursued the EA management is analyzed. Key tasks thereby are the deployment of monitoring techniques for EA management, the development of change requirements to meet performance targets, and the execution of the governance process for EA management.

Above, we expatiated how patterns from the EAMPC can be used to complement TOGAF's ADM. The concern-driven approach of the EAMPC is thereby used to operationalize the general phases of the ADM in an organization-specific manner. Especially the possibility to derive the information model from the concerns risen by the stakeholder provides an extension to the ADM. Furthermore, the best-practice visualizations of the EAMPC can be used to augment the deliverables defined within TOGAF. Subsequently the approach discussed above is exemplified along typical application examples.

4. Exemplifying the approach

The managed evolution of the application landscape is commonly regarded a focal point of EA management

endeavors [7], [16], [18], [20], [28]. Hence, application landscape management is used as an application example for the approach presented above. The application landscape can be seen as a juncture of business and IT as its central concepts are the business

Prior to developing an integrated view on the EA, the ADM cycle starts with the documentation of the *business architecture* (phase *B*), *information systems architecture* (phase *C*), and *technology architecture* (phase *D*) of the enterprise. The EAMPC can be used

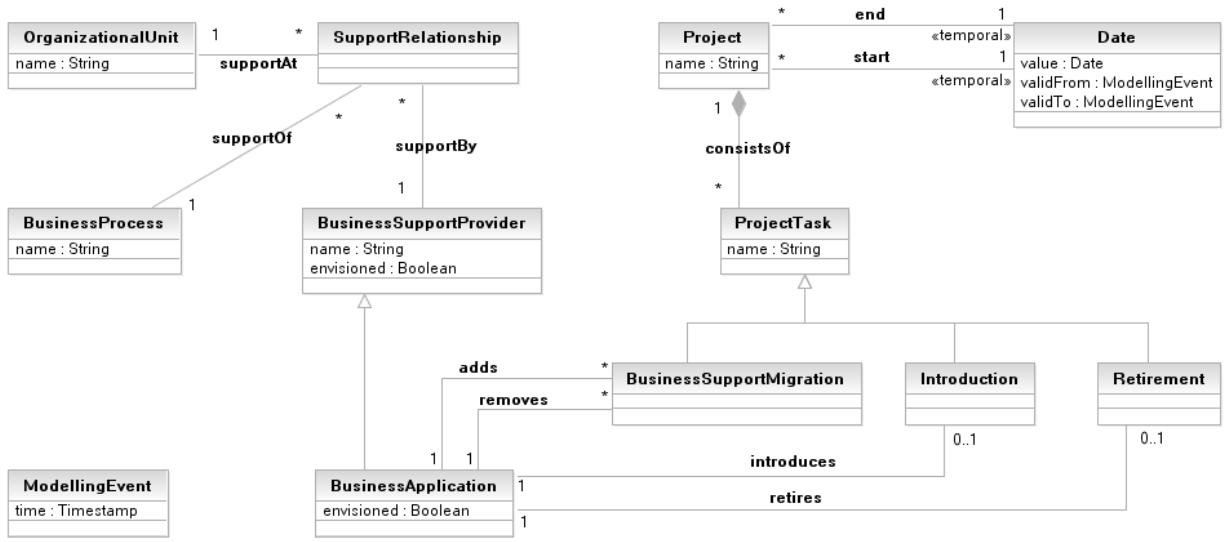

Figure 2: Information model of an I-Pattern

applications, which on the one hand provide support for business processes and therefore exchange information via certain interfaces and on the other hand run on certain IT

Following the ADM cycle, the stakeholders of the application scenario are identified in phase *A* - Architecture Vision. Exemplary stakeholders, which are concerned about the managed evolution of the EA are e.g. CxOs, the Program Management Office, and Executives. The concerns of these stakeholders need to be defined in order to ensure their commitment to the EA management endeavor and keep them satisfied. The list of EA-related concerns as contained in the EAMPC can be used to facilitate the discussions during identification of stakeholders' concerns. The following concern was selected to motivate the application example:

> *How can we ensure a managed evolution of the application landscape? Thereby, future planning needs to be supported and traceability must be ensured of management decisions whereas business as well as technical aspects need to be taken into account.*

to derive information about the data that needs to be gathered in order to address the above stated concern. The corresponding information model of an I-Pattern is shown in Figure 2.

The EA management pattern catalog provides a glossary of the terms used in the I-Pattern to ensure a common understanding of the involved stakeholders regarding the concepts in the information model:

- **Business application** is a software system, which is part of a business information system of an organization. A business application thus provides support for at least one business process, i.e. infrastructure systems are not considered business applications in this context.
- **Business process:** A business process is defined as a sequence of logical, individual functions with connections in between. A process here should not be identified with a single process step, as found e.g. in an event driven process chain (EPC). It should be considered a coarse grained process at a level similar to the one used in value chains, i.e. partially ordered, linear sequences of activities. Additionally, a process maintains relationships to the business applications,

which support it at the different organizational units. As in application landscape management, the business processes are considered to be fixed, i.e. they are not transformed by projects.

- **Organizational unit:** An organizational unit represents a subdivision of the organization according to its internal structure. An organizational unit is a node of a hierarchical organization structure, e.g. a department or a branch. In application landscape management, organizational units are considered fixed - thus, they are not transformed by projects.

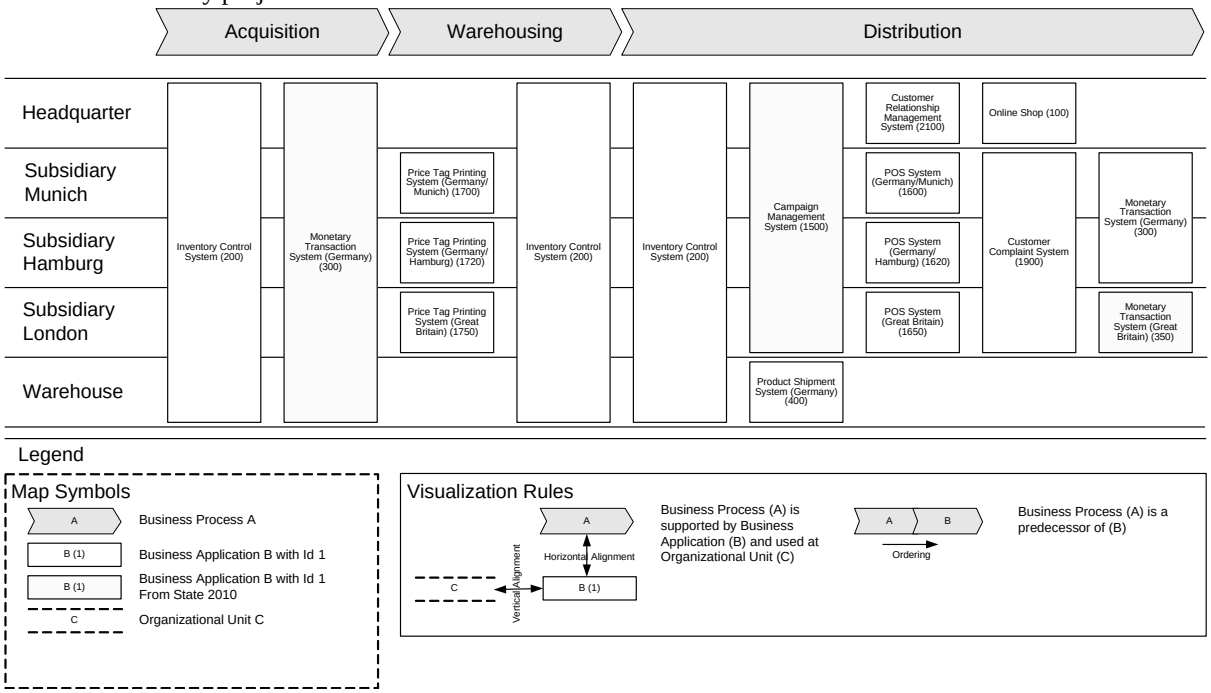

Figure 3: Example on a process support map

- **Business Support Migration** represents a project task migrating the provision of a specific business support from a source business application to a target one. The business support is considered fully migrated, once the date specified in endsAt has passed.
- **Business support provider:** A business support provider is a constituent of an application landscape, used to indicate that a related business process is supported at a distinct organizational unit, without giving a specification, which business application is likely to provide this support, if any. In spite of the similarities to the business application, the envisioned support provider is not affected by projects but has nevertheless a period of validity associated. Thereby, it references the point in time it has been modeled at and (optional) the point in time, the provider became invalid.
- **Introduction** is a specific type of project task introducing a distinct business application. After the date specified in **endsAt**, the associated business application is considered to be in production.
- **Project:** Projects are implementers of organizational change. Therefore, adaptations of the application landscape are the result of a project being completed. Projects are scheduled activities and thus hold different types of temporal attributes, their **startDate** and **endDate** on the one hand. On the other hand, projects are **plannedAt** respectively removedAt certain points in time referring to the time of their creation or deletion. This effectively results in a period of validity, which is assigned to each project. In application landscape management, projects are considered to only affect business applications in general and their business support provided, in special. Projects do not affect business processes or organizational units in this model.
- **Project task** is the abstract base concept for the different accomplishments of projects as considered in this pattern. Each project task

spans a distinct period of time, enclosed by the two points in time **startsAt** and **endsAt**. The project tasks indicate the discrete events of change, connecting the different states of the EA to a chronological sequence.

visualizes which business application supports which business process at which organizational unit.

Viewpoints, like the process support map in Figure 3 can additionally be used to perform gap analyses between different states of the architecture

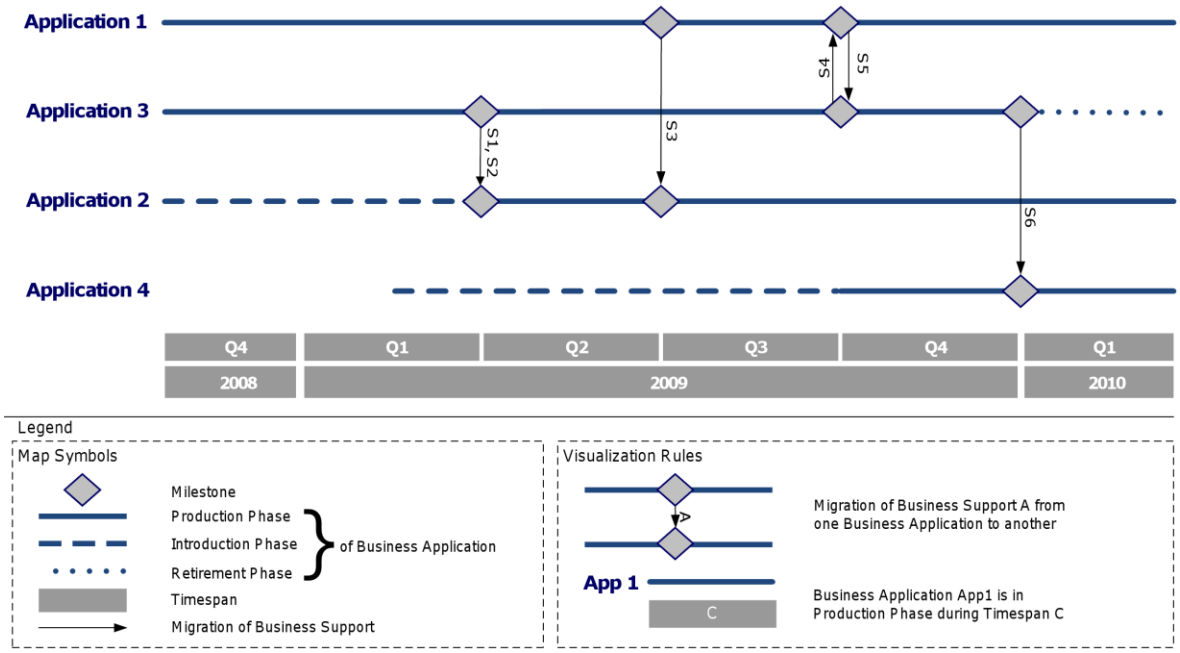

Figure 4: Business support migration plan

- **Retirement** is a specific type of project task retiring a distinct business application. After the date specified in **startsAt**, the associated business application is considered to be in retirement.
- **Support relationship**: represents the support of a specific business process by a specific business support provider at a specific organizational unit.

In order to gather data according to the given concern, different M-Patterns ranging from automatic gathering to decentralized manual data acquisition as described by Moser et al. [25] can be utilized. Furthermore, the EAMPC provides support for the identification of required viewpoints, which visualize the data gathered in the preceding step. A cluster map [29] can be utilized to describe the information systems architecture and to make the business applications of the enterprise and their responsible organizational units explicit.

The EAMPC supports the Opportunities and Solutions phase of TOGAF by creating an integrated view on the current and future states of the EA and allows comparisons between them. Therefore, a viewpoint according to the V-Pattern *process support map* can be utilized (cf. Figure 3). The viewpoint

[24]. For this differences between the states are highlighted on the viewpoint e.g. via shadowing. In order to derive intermediate planned states of the EA, M-Patterns as e.g. introduced in [5] or [22] can be utilized. Buckl et al. [5] for example explain a method how different planned states of the EA can be derived from a project portfolio selection explicit.

The developed planned states of the EA provide input for the migration planning phase of the ADM. Within this phase a roadmap for EA transformation is developed including milestones for the evolution of the EA. While TOGAF only details on viewpoints, which provide snapshots of the EA at a certain time, similar to the process support map introduced above, the EAMPC contains V-Patterns, which can be utilized to detail on transformation impacts of the provided business support of an enterprise [3]. Figure 4 provides a business support migration plan, which explicates the migration of business support during the transformation of the application landscape.

In order to document information according to the viewpoints from Figure 3 and Figure 4 an information model suitable to store time-related information as introduced above is necessary. Although, TOGAF contains an information model, time-related aspects are not referred to in the current version.

After the roadmap for the EA transformation is developed and decided upon in phase *Migration Planning*, the phase *Implementation Governance* realizes the transformation. M-Patterns of the EAMPC can provide input for the implementation of this phase. An M-Pattern, for example, provides information how to identify critical projects, e.g. due to high risks, business impact, or changes to critical business applications. Furthermore, a procedure how to establish quality gates to ensure the architecture-conform development of projects is explicated in an M-Pattern.

5. Resumee

In this paper, we discussed the ADM of TOGAF, which provides guidelines for executing an EA management. These guidelines nevertheless stay on a rather generic level and might hence not be directly applicable to a specific enterprise. Subsequently, the pattern-based approach to EA management as presented in the EA management pattern catalog was discussed [2]. This approach provides guidance for addressing specific EA-related concerns with methods, viewpoints, and information models. From this, we elaborated how the pattern-based EA management approach can be used to complement the ADM, in order to create an enterprise-specific EA management accounting especially for the enterprise's most important EA-related concerns. The idea was further exemplified in Section 4 showing how the two approaches can be combined to design an EA management.

The utilization of EA management patterns in this context gives rise to a research question for the further development of pattern-based EA management. The patterns currently documented in the EA management pattern catalog are very likely to span different phases of an EA management process, as they provide comprehensive best-practices for addressing a specific concern as a whole. It might nevertheless be interesting to investigate, if these patterns could be organized according to the phases of a typical EA management process, similar to the TOGAF ADM, in addition to their specialization on distinct concerns. Doing so, modularity of the EA management approach could be further promoted, allowing an enterprise to choose the patterns e.g. for documenting concern-specific information (cf. Phases B to D) independently from the patterns for enacting control over the projects (cf. Phase G), in which the according concern is addressed. Furthermore, the guidelines for integration of different pattern types should be enhanced to make the integration easier to comprehend or, if possible, fully automated.

6. References

[1] C. Alexander, S.Ishikawa, M. Silverstein, M. Jacobson, i. Fiksdahl-King, and S. Angel, A Pattern Language. New York: Oxford University Press, 1977.

[2] S. Buckl, A.M. Ernst, J. Lankes, and F. Matthes, "Enterprise architecture management pattern catalog (version 1.0, February 2008)". Chair for Informatics 19 (sebis), Technische Universität München, Munich, Tech. Rep., 2008. [Online]. Available: http://wwwmatthes.in.tum.de/wikis/eam-pattern-catalog/home

[3] S. Buckl, A.M. Ernst, F. Matthes, and C.M. Schweda,, "Visual Roadmaps for Enterprise Architecture Evolution". In The 1st International Workshop on Enterprise Architecture Challenges and Responses, Korea, 2009.

[4] S. Buckl, A.M. Ernst, F. Matthes, C. Schulz, and C.M. Schweda. "Constructing an Enterprise-specific Radar System for Assisted Project Surveillance". In: Workshop MDD, SOA und IT-Management (MSI 2009), Oldenburg, 2009.

[5] S. Buckl, A.M. Ernst, F. Matthes, and C.M. Schweda, "An Information Model for Managed Application Landscape Evolution". In: Journal of Enterprise Architecture (JEA), Volume 5, pages 12-26, 2009.

[6] S.Buckl, A.M. Ernst, J. Lankes, F.Matthes, and C.M. Schweda, „State of the art in enterprise architecture management 2009," Technische Universität München, Tech. Rep., 2009.

[7] C. Braun and R. Winter, "A comprehensive enterprise architecture meta-model," in Enterprise Modelling and Information Systems Architectures 2005, ser. LNI, J. Desel and U. Frank, Eds., vol. 75. GI, 2005, pp. 64-79.

[8] F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad, and M. Stal, Pattern-oriented software archtiecture: a system of patterns. New York, NY, USA: John Wiley & Sons, Inc., 1996.

[9] U. F. C. Council, "Federal enterprise architecture framework (feaf)," http://www.cio.gov/documents /fedarch1.pdf (cited 2009-02-25), 1999.

[10] U. D. of Defense, "Department of defense architecture framework (dodaf)," http://www.defenselink.mil/cio-nii/docs/DoDAF_Volume_I.pdf (cited2009-02-25), 2007.

[11] G. Dern, Management von IT-Architekturen (Edition CIO). Wiesbaden: Vieweg, 2006.

[12] G. Engels, A.Hess, B.Humm, O.Juwig, M. Lohmann, and J.-P. Richter, Quasar Enterprise – Anwendungslandschaften serviceorientiert gestalten. Heidelberg: dpunkt.verlag, 2008.

[13] A.M. Ernst and A.W. Schneider, "Roadmaps for Enterprise Architecture Evolution". In: 2nd European Workshop on Patterns for Enterprise Architecture Management (PEAM2010), Paderborn 2010.

[14] A. Ernst, "Enterprise architecture management patterns," in PLoP 08: Proceedings of the Pattern Languages of Programs Conference 2008, Nashville, 2008.

[15] E. Gamma, R.Helm, R.Johnson, and J. M. Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software (Addison-Wesley Professional Computing Series). Addison-Wesley Professional, 1994.

[16] A. Garg, R.Kazman, and H.-M. Chen, "Interface descriptions for enterprise architecture," science of Computer Programming, vol. 61, no. 1, pp. 4-15, 2006.

[17] International Organization for Standardization "ISO/IEC 42010:2007 Systems and Software Engineering - Recommended practice for architectural description of software-intensive systems", 2007.

[18] H. Jonkers, L. Goenewegen, M. Bonsangue, and R. van Buuren, „A language for enterprise modelling," in Enterprise Architecture at Work, M. Lankhorst, Ed. Berlin Heidelberg, New York: Springer, 2005.

[19] W. Keller, IT-Unternehmenarchitekur. Heidelberg: dpunkt.verlag, 2007.

[20] L. Kirchner, „Eine Methode zur Unterstützung des IT-Managements im Rahmen der Unternehmensmodellierung," Ph.D. dissertation, Universtität Duisburg-Essen, Berlin, 2008.

[21] S. Kurpjuweit and R. Winter, "Viewpoint-based meta model engineering," in EMISA, ser. LNI, M. Reichert, S. Strecker, and K. Turowski, Eds., vol. P-119. GI, 2007, pp. 143-161.

[22] J. Lankes, "Metrics for application landscapes – status quo, development, and a case study". Ph.D. dissertation, Technische Universität München, Fakultät für Informatik, Munich, Germany, 2008.

[23] M. Lankhorst, "Introduction to enterprise architecture," in Enterprise Architecture at Work. Berlin, Heidelberg, New York: Springer, 2005.

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

[25] C. Moser, S. Junginger, M. Brückmann, and K.-M. Schöne, "Some process patterns for Enterprise architecture management in Software Engineering 2009" In: Workshop on Patterns for Enterprise Architecture Management (PEAM2009), Kaiserslautern 2009.

[26] J.W. Ross, P. Weill, and D.C. Robertson, "Enterprise Architecture as Strategy". Boston, Massachusetts, USA: Harvard Business School Press, 2006.

[27] The Open Group. "TOGAF 'enterprise edition' version 9", 2009.

[28] L.W.N. van der Torre, M.M. Lankhorst, H.W.L. ter Doest, J.T.P. Campschroer, and F. Arbab, "Landscape maps for enterprise architectures," in Advanced Information Systems Engineering, 18th International Conference, CAiSE 2006, Luxembourg, Junde 5-9, 2006, Proceedings, ser. Lecture Notes in Computer Science, E. Dubois and K. Pohl, Eds., vol. 4001. Springer, 2006, pp. 351-366.

[29] A. Wittenburg, "Softwarekartographie: Modelle und Methoden zur systematischen Visualisierung von Anwendungslandschaften". Ph.D. Dissertation, Fakultät für Informatik, Technische Universität München, 2007.