Exemplifying a Framework for Interrelating Enterprise Architecture Concerns

Sabine Buckl, Florian Matthes, Christopher Schulz and Christian M. Schweda

Chair for Software Engineering of Business Information Systems (sebis),
Technische Universität München,
Boltzmannstr. 3, 85748 Garching, Germany
{sabine.buckl,matthes,christopher.schulz,schweda}@in.tum.de
http://wwwmatthes.in.tum.de

Abstract. In recent years, enterprise architecture (EA) management has gained increasing attention as means to support enterprises in adapting to changing markets and in seizing new business opportunities. A multitude of approaches and frameworks making prescriptions on how to document the different states of the EA have been developed. These approaches target different purposes and correspondingly different concerns (areas of interest) in the architecture. Hence, an enterprise seeking to develop or evolve an organization-specific EA documentation technique most likely runs into difficulties to understand the interdependencies between different frameworks and approaches.
The paper addresses the aforementioned challenge by presenting a framework for EA concern interrelations, which can be used to systematically analyze the concern relationships of different approaches. The applicability of the framework is demonstrated by means of a case study from the automotive industry, in which the framework is used for the development and enhancement of an EA description for risk management.

1 Introduction and motivation

The increasing frequency of change, modern enterprises face in today's globalized and competitive environments, leads to a rising internal complexity of the socio-technical system *enterprise*. A promising and commonly accepted instrument to deal with this complexity and to foster business-IT-alignment is enterprise architecture (EA) management [1, 2, 3]. Originating from the field of information systems architecture (cf. [4]), EA management takes a holistic perspective targeting all areas of an enterprise from business and organizational via application and information to infrastructure and data aspects. EA is thereby in the sense of the ISO Standard 42010 understood as the "fundamental organization of a system [enterprise] embodied in its components, their relationships to each other, and to the environment, and the principles guiding its design and evolution" [5]. A multitude of methods to EA management has been developed by researchers (cf. [6, 7, 2, 8, 3]), practitioners (cf. [9, 10, 11]), and standardization bodies (cf. [12]). Although differing in respect to the selected scope, reach, and focus, the proposed methods usually distinguish the following activities of the EA management function: a) document and maintain the current state of the EA, b) develop and describe planned and target states of the EA, c) enact and communicate planned EAs and architectural principles, and d) analyze and evaluate architectures.

A central challenge arising during the aforementioned typical activities of EA management is *stakeholder involvement* [12, 13]. To address this challenge the architectural documentations referring to the current, planned, or future states of the EA, need to represent the corresponding concerns of the stakeholders. Thereby, concerns are defined in accordance with the ISO Standard 42010 as "those [areas of] interests which pertain to the system's [enterprise's] development, its operation or any other aspect that are critical or otherwise important to one or more stakeholders" [5]. Put in other words, a concern can be understood as the area of the enterprise that the respective stakeholder is interested in. Whereas, a product manager for instance is interested in the performance of the business process and the services responsible for creating the product, an application manager is concerned with the standard conformity of the business applications.

In the holistic perspective of EA management the two aforementioned concerns are clearly interrelated. Crosspoints are the terms service and business application. The thereby denoted concepts are most likely related – a service is provided by business applications – or may even be identical – two terms referring to the same concept. The differences in terminology can be explained by the different language communities the two stakeholders belong to. To develop a comprehensive EA description, covering the different concerns of the stakeholders, the relations between concerns should be made explicit. Furthermore, these relations between concerns could be utilized to decide on a maturity roadmap for the evolution of the EA management function. In an initial step an enterprise might decide to stick to a more simple concern, while after first results have been achieved the concern is expanded to support a more detailed investigation. This directly yields the research questions that our paper addresses:

- *How does a framework for interrelating EA concerns look alike?*
- *What types of relationships between EA concerns exist?*
- *How can the framework be utilized in developing and evolving EA models?*

In the remainder of the article above research questions are answered by developing a conceptual framework for EA concern relationships. The framework is derived and presented in a step-wise manner (see Section 2). A real world case study form the automotive industry is utilized to illustrate the applicability of the framework in answering the aforementioned research questions in Section 3. Complementingly, prominent approaches from literature to EA management in general and EA modeling in special are revisited to ensure comprehensiveness of the approach (see Section 4). In Section 5, we conclude with a critical reflection of the achieved results and an outlook on future research directions.

2 A framework for EA concern relationships

In line with the ISO 42010 [5], descriptions of architectures are comprised of viewpoints that conversely reflect areas of interest in the corresponding architecture. These areas of interest, called *architectural concerns* in the terms of the ISO 42010, not only define the part of the architecture that should be considered, but also bring along a conceptualization of the EA. Put in other words, a concern builds and centrally employs an underlying domain ontology targeting a specific part of the overall EA. Considering two or more different architectural concerns, it may be beneficial to understand, in which ways the corresponding areas of interest relate. In order to facilitate in-depth discussions on concern relationships, we subsequently present are more precise definition of what an architectural concern means. A concern can be described by two fundamental properties, namely

– a **selection** of relevant architectural elements, i.e. of *things* and their *properties* in an ontological sense, that are in the area of interest, and
– a **conceptualization** describing the architectural *types* and *property types* abstracted from the selected (relevant) architectural elements.

Illustrating above definition of an architectural concern in the EA context, we formulate a Concern A that describes the

> *business applications in relationship to their responsible organizational units* (conceptualization) of an enterprise, *restricted to organizational units located in Germany* (selection).

In the light of the above definition, we can derive different types of relationships between concerns. A basic relationship expresses the fact that two concerns have at least one type in common, i.e. that their conceptualizations share at least one type. This entails a more sophisticated discussion on the question of how to understand 'share' in this context. In more detail, two ways to understand the term exist: *syntactic* and *semantic*. While sharing in the latter sense means that two concerns have a type with similar meaning in common, syntactic sharing describes that two concerns are related via a concept with a common name. To qualify these two different understandings of the term 'share', we use the term *intersection* to denote syntactic sharing.

Based on the intersection-relationship, we establish the notion *compatibility* as a relationship between two concerns, indicating that all intersecting types and property types of two concerns are also semantically shared. If in contrast at least one intersecting type or property type is assigned distinct meanings in both concerns, the concerns are called *incompatible*. Introducing a Concern B describing

> *business applications and their standardization level* (conceptualization) of an enterprise,

we can exemplify the aforementioned concern relationships in the following. Both concerns are clearly intersecting, as they both employ the type 'business application'. At this point nothing can be said on the compatibility of these concerns.

To answer this question, one would have to delve into the contexts, in which these concerns were raised. With concerns being described as sentences in real language, such investigation would employ analyses of the linguistic communities (cf. Kamlah and Lorenzen [14]) from which the concern descriptions originate. We abstain here from going into the details of interpersonal testing techniques, but call for an intuitive understanding of *equivalent meanings*, based on the assumption that each type described in the concerns was backed by a definitory sentence in a glossary or an ontologically richer structure. Based on the glossary, equivalence of meaning can be understood as stakeholder consensus in accordance to Kamlah and Lorenzen [14]. Two types thereby share a common meaning, if "any informed member of [both] linguistic communities would [...] say so". Given the fact that both concerns referred to a single definitory glossary, explaining what a business application is, we can state that the two concerns are compatible.

Grounded on the basic relationship of compatibility, we establish a *super*-relationship between architectural concerns. One concern is superconcern of another, if the concerns are compatible and all types and property types of the second are intersecting ones. Put more colloquially, the conceptualization of the first concern completely covers the conceptualization of the second one. Introducing another Concern C that describes the

> *business applications in relationship to their responsible organizational units and their supported business processes* (conceptualization) of an enterprise.

We further assume that Concern A and Concern C are *maximally* compatible, i.e. that all shared concepts each have an equivalent meaning. Under these premises, Concern C completely covers Concern A, exemplifying superconcern relationship with Concern C being super to Concern A.

Above, we introduced three types of concern relationships based on an intuitive understanding of the concern's underlying conceptualization. In the context of EA management such conceptualizations are frequently described as object-oriented meta-models, see e.g. Buckl et al. in [15], Johnson and Ekstedt in [16], and Österle et al. in [17]. Subsequently, we shall call these meta-models in accordance with Buckl et al. *information models.* Based on such more formally described conceptualizations the aforementioned types of concern relationships can be defined more concisely and described in more detail. We nevertheless abstain from discussing the subtleties of such definitions, an application of the concern relationships along exemplary information models is given in Section 3. Some additional remarks on concern relationships based on their corresponding conceptualizations are described by Buckl et al. in [18].

Concluding the exposition of the concern relationship framework in this section, we discuss on relationship types that ground in the selection that constitutes a concern. The first relationship type to devise, is the one of *instance level intersection.* Two concerns are related in this way, if they – applied to the same EA – cover at least one common element, i.e. thing, from the architecture. Refraining the examples from above, we display Concern D that describes the *organizational units* (conceptualization) of an enterprise.

Assuming that both Concern A and Concern D have a common understanding what the term "organizational unit" means, the concerns are intersecting on instance level. Complementing, we can define a *non-intersection* relationship. In addition, we can derive an *instance level superconcern* relationship, expressing that one concern – applied to the same EA – covers all elements that another concern – applied to this EA – covers.

We complement the above framework with a discussion on the utilization of the relationships in the context of EA management. Refraining an argument put forward by Aier et al. in [19], an enterprise-specific EA description technique must match the requirements of the using enterprise in two directions, namely in "width" and "depth". Width in this context means, that the technique must cover the relevant parts of the EA embracingly, i.e. must be able to answer questions of all relevant EA stakeholders. Complementing, the criterion of depth accounts for the fact that for answering certain EA questions more detailed information about the EA may be needed. While in the context of initially devising an EA description technique, multiple different approaches, also ones not using concern relationships, might be helpful, we expect that the relationships especially simplify the evolution of the EA description technique in response to changing demands of the EA stakeholders. To illustrate how this can be achieved, we build on the EA management pattern catalog of Technische Universität München [20, 21]. This catalog expatiates a set of practice-proven EA relevant concerns that are complemented with information models reifying the underlying information demands. If the concerns were related using the aforementioned relationship types, an enterprise willing to evolve its own EA description technique based on the information models would have the following advantages:

- Feeling the need to "widen" the scope of the EA description technique, the enterprise could search the set of concerns that are *compatible* with the currently employed ones. These concerns and their complementing information models should easily integrate into the current description technique.
- Feeling the need to "deepen" the scope of the EA description technique, the enterprise could traverse the *superconcern*-relationships, in order to determine concerns that employ the currently described concepts, but cover additional information.

Summarizing the above, concern relationships provide decision support for developing an enterprise-specific EA description technique. Illustrating this fact, we subsequently provide an application example from a practice case, where we applied concern relationships to prepare decision making on the appropriate EA description technique for addressing an EA challenge in both 'right width and depth'.

3 Application Example

An international manufacturer from the automotive industry intends to construct a new overseas production plant. Thereby, the local market should be served in a customer friendlier manner, while currency fluctuations have to be mitigated in order to remain competitive. Part of this large-scale project is the deployment and roll-out of approximately 170 partly interconnected applications supporting a modern and up to date assembling of vehicles. For the sake of profitability, existing applications are duplicated and slightly adjusted to the local conditions before being deployed and tested. These applications are typically referred to as *group applications*, meaning they are used in a slightly adapted manner at different production plants. Thereby, those new applications are also connected with their counterparts on the other side of the globe, enabling a company-wide business intelligence, monitoring, and controlling capability.

The local production and sales of vehicles is constrained by country-specific laws and regulations. In the case of our international car manufacturer, the disclosure of company-internal information is regulated by the *antitrust law*. Once a piece of information is revealed to one supplier or customer, it must be automatically made accessible to other third parties. Contrastingly, if a *class action suits* succeeds, hence an objection raised by several plaintiffs is sustained, lawyers will have the possibility to access different company-internal information as well.

As a matter of fact, information is mostly stored within applications nowadays allowing an effective and efficient data organization and retrieval. These applications directly and indirectly support the execution of different business processes (e.g. car construction process, car sales process, etc.). Due to above alluded rules and regulations, external lawyers and auditors would have unimpeded access to those applications as well as their contained data. As second consequence, the attorneys were also allowed to retrieve information not only stored locally on those applications, but also on other applications which are connected via network technology. Hence, the real set of information being also transitively accessible through the 170 applications would be much broader than anticipated before, representing a danger for corporate secrets.

To respond to this challenge, the international manufacturer launched a project targeting at modifying those applications also granting access to data, which is not explicitly needed for the production plant. Initially, 35 applications as well as their exported and imported interfaces providing access on data not relating to the overseas factory where identified and marked as inherently critical. The classification thereby was mostly based on the experience application owners and users had with those applications. Subsequently, the 35 applications classified as critical will undergo time-consuming analysis and complex functional modifications aiming at restricting their data scope to the information relevant for the local context. The remaining 135 applications not classified yet, will be analyzed in a two step approach. To reduce workload, a quick but comprehensive interface analysis of the 135 applications will be performed in a first step to identify those applications, which need to be considered in more detail. In a second step, the applications classified as critical will be analyzed as described before.

When being confronted with the application interface analysis, the car manufacturer asked us for help. The problem description of our cooperation reads as follows:

> How can criticality of application systems interfaces be determined? Which concepts and relationships are decisive for criticality of interfaces exported by applications.

After thoroughly studying a list of factors all determining if an interface is critical, we suggested three different concerns, which are detailed in the following.

As a first step, we generally defined an interface as being critical whenever it has no authorization mechanism available. Hence, once an interface possesses such a mechanism, its criticality is set to low and it needs no further investigation. Figure 1 illustrates the underlying information model I-1 consisting only of one single entity named `interface` and the attributes `hasAuthorization`, `hasAuthentication`, and `criticality`. Thereby, the attribute `criticality` is a derived attribute, whose value is influenced by the two other attributes. This fact is illustrated utilizing the notation of *feature dependency relationship* as defined by Buckl et al. in [22].

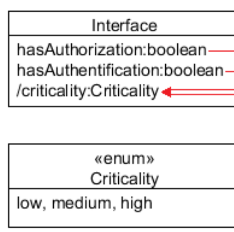

Fig. 1. Interface criticality defined by an available authentication mechanism

An alternative way to assign a specific criticality to an application interface is to closely examine the security type of the data the interface exchanges. This can be expressed through a second concern, which is described in the information model I-2 depicted in Figure 2 by showing two entities, `interface` and `data`. Thereby, an `interface` is regarded as being low critical whenever it is used for exchanging public data only. If the data is either internal, confidential, or secret, the interface's criticality shifts to medium or high. Again the notation of the *feature dependency relationship* (cf. [22]) is used.

Compared to our aforementioned framework for interrelating EA concerns, both models are compatible. The entity `interface` is semantically and syntactically equal within the two concerns, both models could complement each other.

After further consideration, we proposed a third concern whose information model I-3 also addresses the goal of determining critical and non-critical inter-

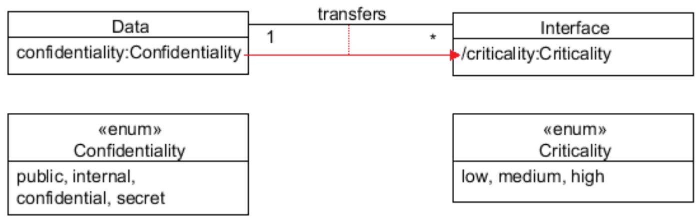

Fig. 2. Interface criticality contingent on the security type of exchanged data

faces. Thereby, an interface may only be deemed as being critical when it is from the type 'exporting' with regards to the interface's underlying application. In addition, the interface has to transfer non-public data, thus data which is either internal, confidential, or secret.

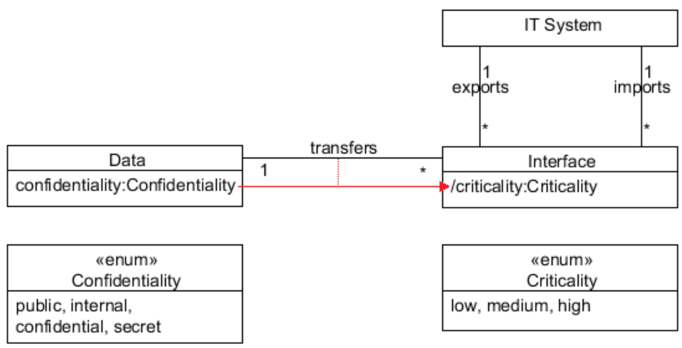

Fig. 3. Interface criticality depending on interface and exchanged data security type

As expatiated on in Section 2, Figure 3 represents a super concern with regards to the concern specified in Figure 2. Additional relationships between information model I-1 and I-2, respectively I-3 exist. Each of the above pair describes two compatible concerns. The interrelations between the different concerns worked out so far, are depicted in the map presented in Figure 4. In the map, we further added an "artificial" concern I-3', as the integration of I-1 and I-3.

When discussing the results with the international car manufacturer, two further fields of work arose. Firstly, we identified additional attributes, which may be regarded as being relevant for the criticality of an application and its interfaces, i.a. multi-client capability on different application layers, concrete data content (i.e. business objects and their criticality), configurability of the

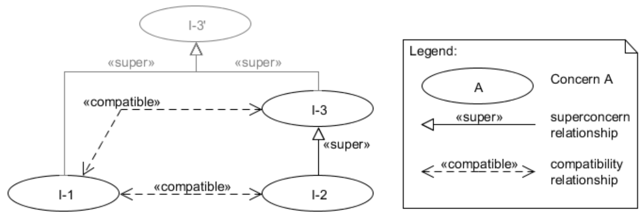

Fig. 4. Concern map for the exemplary concerns

interface, etc. Those aspects could be either directly incorporated in existing information models or new concerns had to be created. Secondly, we had to find out which of the required information addressing the concerns of application interface criticality presented above is already stored in existing EA management tools. If the specific date is not available it has to be gathered, e.g. through interviews with the according application owners or an online survey. In all, the goal of determining the criticality of application interfaces was addressed through several interrelating EA concerns, summarized by a concern relationship map, which follows the presented framework.

4 Concern relationship in the state-of-the-art in EA descriptions

In addition to the first successful application of the framework for EA concern relationship in practice as discussed in the preceding section, we analyzed existing literature on EA management. EA management is a research topic with an increasing number of publications [23], in which especially the fields of EA modeling and EA analysis are heavily researched. Central to both activities is the notion of the architectural description reflecting the corresponding architectural concerns. Subsequently, we explore the state-of-the-art in EA modeling and analysis with a special emphasis on the EA descriptions used thereby as well as their underlying concerns in order to validate the framework for interrelating EA concerns as presented in Section 2.

The approach of *multi-perspective enterprise modeling (MEMO)* was initially presented by Frank in 2002 [6]. Therein, Frank outlines a modeling framework, which is based on an extendable set of special purpose modeling languages. The special purpose modeling languages correspond to the different language communities, which typically exist in an enterprise, e.g. salespersons or project managers. By further providing a common meta-language, the *MEMO meta modelling language (MML)* [24], which the special purpose languages rely on, the integration of the different languages can be facilitated. Examples for special purpose languages are e.g. the *strategy modelling language (MEMO-SML)* [25] and the *organization modelling language (MEMO-OrgML)* [26]. Concepts of two different languages can be associated, i.e. are common to both languages. Therefore, the different concerns addressed by the MEMO special purpose languages can be regarded as being *intersected* and *compatible*.

The communication challenge already mentioned in the approach presented by Frank, is further discussed by Buckl et al. in [27], Schelp and Winter in [28], and Schönherr in [13]. While the first publication refers to the communication challenge within an enterprise, the latter two publications discuss the different language communities mirroring the academic groups conducting research in the area of EA management. Due to the absence of a standardized terminology and a commonly-accepted description language for EA management, the different approaches and communities have developed their own terms, leading to *incompatibility* issues, when different approaches are combined.

A systemic perspective on EA modeling is presented by Wegmann et al. in [29]. They provide a method and a tool to formalize the alignment of the multiple levels that constitute an EA. In particular, they propose to organize the different concepts that constitute the EA in *organizational* and *functional levels*. Thereby, the functional levels represent behavioral and the organizational levels the constructional hierarchy. Within each organizational level, two different viewpoints are available, the *information viewpoint* – a black box view on the respective concern – and the *computational viewpoint* – a white box specification of the concern. These viewpoints can be refined, which results in a hierarchy of viewpoints and the underlying information, respectively. Based on this understanding, a *super*- and *sub-relationship* between different viewpoints and concerns can be identified. Furthermore, Wegman et al. point out that a "vocabulary mapping" [29] between related concepts on different organizational levels has to be performed, which can be ascribed to the already mentioned *compatibility* question.

In [7] Kurpjuweit and Winter present a *systemic approach to meta model engineering*, which consists of five steps: 1) Identification of relevant concerns, 2) requirements elicitation, 3) viewpoint relationship overview, 4) meta model[1] fragment selection or design, and 5) meta model fragment integration. For the identification of relationships between concerns, especially the last step is of importance. Kurpjuweit and Winter allude to two different types of relationships between concerns as mirrored in the corresponding meta model fragments. Firstly, concerns can be *intersected*, which rises the challenge that the terminology must be adjusted. Secondly, a generalization or specialization may be necessary, which is caused by a *superconcern* or *subconcern* relationship between different concerns.

In [16] Johnson and Ekstedt discuss an approach to EA decision making based on EA models and analyses. Based on the current documentation of the EA, future EA scenarios are derived and assessed in respect to selected quality attributes, as e.g. performance, interoperability, availability, security, or usability. For each of these quality attributes, an *influence diagram* is presented by the authors, which details on causal dependencies between architectural properties.

[1] The term *meta model* thereby corresponds to the term *information model* as used in this paper.

If for instance availability is considered, the reliability or recoverability of the system under consideration influences the overall availability. Thus, the concerns included in an influence diagram have a *subconcern/superconcern* relationship to each other, while concerns of different influence diagrams may be *intersected*.

Central ideas for a language for describing EAs are outlined by Jonkers et al. in [30] and further elaborated in [31]. They discuss two requirements that relate to the notion of the concern, namely *meta model flexibility* and *integration of heterogeneous models*. Meta model flexibility is thereby meant to demand that general EA description concepts can be refined to organization-specific concepts and standards. Conversely, a language for describing EAs must facilitate the integration of heterogeneous models, as special purpose languages for specific parts of the EA exist, whose concepts can be 'translated' into or 'associated' with concepts for EA description. Jonkers et al. further discuss in [30] the distinct levels of detail, on which the corresponding descriptions may act. In this sense, one or more concerns from special purpose modeling languages might be 'aggregated' into a single concern at the EA description level. In [31] Jonkers et al. provide a core meta-model for architectural descriptions, more precisely behavioral aspects thereof, based on a dichotomy of *service-* and *implementation-aspects*. The corresponding basic concepts *service* and *behavior element* are refined to specific *business*, *application* and *infrastructure-layer* concepts, respectively. Nevertheless, the basic modeling language allows to omit concepts on some of these layers, if they are not of interest for the specific modeling purpose. Put in the terminology of our concern-discussion, the language implicitly allows for the derivation of *subconcerns*.

Summarizing the state-of-the-art in EA concern relationship modeling it can be stated, that the framework presented in Section 2 can be used to classify the different relationships as explicitly or only implicitly proposed in current publications. Furthermore, it provides a framework for conceptual interrelation as well as proposes the *concern map* for making the relations explicit.

5 Conclusion and Outlook

In this paper, we discussed the concept *concern* in the context of EA modeling and EA management, respectively. Concerns thereby represent distinct information demands in respect to the described architecture and can hence be reflected by information models that provide a 'schema' for the information to be modeled and collected during the corresponding EA management process. Based on this model-centric understanding of concern, we developed a framework for interrelating EA concerns and showed its applicability in a real world case study from the automotive industry. Thereby, especially the utility for evolving modeling techniques was discussed. Complementingly, we revisited the state-of-the-art in EA modeling and analysis.

This paper presents itself as a research-in-progress paper, that outlines the idea of concern relationships. From this point different directions for further development and research are open. At first, the informal presentation of the relationship types in this paper should be complemented with a more formal understanding thereof. In this respect, the semantics of the different relationship types could be formally defined based on a formal understanding of architectural descriptions. This would then allow for more intricate considerations on the relationships between information models that reflect the corresponding concerns. Secondly, concern relationships might not only be an issue of theoretical interest, but may prove to be interesting for supporting maturity considerations for EA management functions. In this respect, the relationships could also be beneficial for evolving an already existing EA management function towards a 'broader' and 'deeper' coverage of the overall EA. This subject has yet not been investigated in practice, although the notion of the "related" concerns as outlined by Ernst in [32] and in [21] have shown to be beneficial during the process of establishing and evolving an EA management function.

6 Acknowledgement

The case study research presented in this paper was possible thanks to the support of Volkswagen AG, Wolfsburg. The authors of the document wish to acknowledge the industry partners for their help and important contributions.

References

1. Henderson, J.C., Venkatraman, N.: Strategic alignment: leveraging information technology for transforming organizations. IBM Systems Journal **32**(1) (1993) 472–484
2. Lankhorst, M.: Enterprise Architecture at Work: Modelling, Communication and Analysis. Springer, Berlin, Heidelberg, Germany (2005)
3. Ross, Jeanne, W., Weill, P., Robertson, David, C.: Enterprise Architecture as Strategy. Harvard Business School Press, Boston, MA, USA (2006)
4. Zachman, J.: Extending and formalising the framework for information systems architecture. IBM Systems Journal **31**(3) (1992)
5. International Organization for Standardization: Iso/iec 42010:2007 systems and software engineering – recommended practice for architectural description of software-intensive systems (2007)
6. Frank, U.: Multi-perspective enterprise modeling (memo) – conceptual framework and modeling languages. In: Proceedings of the 35^{th} Annual Hawaii International Conference on System Sciences (HICSS 2002), Washington, DC, USA (2002) 1258–1267
7. Kurpjuweit, S., Winter, R.: Viewpoint-based meta model engineering. In Reichert, M., Strecker, S., Turowski, K., eds.: Enterprise Modelling and Information Systems Architectures – Concepts and Applications , Proceedings of the 2^{nd} International Workshop on Enterprise Modelling and Information Systems Architectures (EMISA'07), St. Goar, Germany, October 8-9, 2007. LNI, Bonn, Germany, Gesellschaft für Informatik (2007) 143–161

8. van der Raadt, B., van Vliet, H.: Designing the enterprise architecture function. In: 4$^{\text{th}}$ International Conference on the Quality of Software Architectures (QoSA2008), Karlsruhe, Germany (2008) 103–118
9. Dern, G.: Management von IT-Architekturen (Edition CIO). Vieweg, Wiesbaden, Germany (2006)
10. Keller, W.: IT-Unternehmensarchitektur. dpunkt.verlag, Heidelberg, Germany (2007)
11. Niemann, K.D.: From Enterprise Architecture to IT Governance – Elements of Effective IT Management. Vieweg+Teubner, Wiesbaden, Germany (2006)
12. The Open Group: TOGAF "Enterprise Edition" Version 9. `http://www.togaf.org` (cited 2010-02-25) (2009)
13. Schönherr, M.: Towards a common terminology in the discipline of enterprise architecture. In Aier, S., Johnson, P., Schelp, J., eds.: Pre-Proceedings of the 3$^{\text{rd}}$ Workshop on Trends in Enterprise Architecture Research, Sydney, Australia (2008) 107–123
14. Kamlah, W., Lorenzen, P.: Logische Propädeutik: Vorschule des vernünftigen Redens. 3$^{\text{rd}}$ edn. Metzler, Stuttgart, Germany (1996)
15. Buckl, S., Ernst, A.M., Lankes, J., Schneider, K., Schweda, C.M.: A pattern based approach for constructing enterprise architecture management information models. In: Wirtschaftsinformatik 2007, Karlsruhe, Germany, Universitätsverlag Karlsruhe (2007) 145–162
16. Johnson, P., Ekstedt, M.: Enterprise Architecture – Models and Analyses for Information Systems Decision Making. Studentlitteratur, Pozkal, Poland (2007)
17. Österle, H., Winter, R., Hoening, F., Kurpjuweit, S., Osl, P.: Der St. Galler Ansatz des Business Engineering: Das Core Business Metamodel. Wisu – Das Wirtschaftsstudium **2**(36) (2007) 191–194
18. Buckl, S., Matthes, F., Schweda, C.M.: Interrelating concerns in ea documentation – towards a conceptual framework of relationships. In: 2nd European Workshop on Patterns for Enterprise Architecture Management (PEAM2010), Paderborn, Germany (2010)
19. Aier, S., Kurpjuweit, S., Saat, J., Winter, R.: Enterprise Architecture Design as an Engineering Discipline. AIS Transactions on Enterprise Systems **1** (2009) 36–43
20. Buckl, S., Ernst, A.M., Lankes, J., Matthes, F.: Enterprise Architecture Management Pattern Catalog (Version 1.0, February 2008). Technical report, Chair for Informatics 19 (sebis), Technische Universität München, Munich, Germany (2008)
21. Chair for Informatics 19 (sebis), Technische Universität München: Eam pattern catalog wiki. http://eampc-wiki.systemcartography.info (cited 2010-02-25) (2010)
22. Buckl, S., Ernst, A.M., Schweda, C.M.: An extension to the essential meta-object facility (emof) for specifying and indicating dependencies between properties. Technical report, Technische Universität München (2008)
23. Langenberg, K., Wegmann, A.: Enterprise architecture: What aspect is current research targeting? Technical report, Laboratory of Systemic Modeling, Ecole Polytechnique Fédérale de Lausanne, Lausanne, Switzerland (2004)
24. Frank, U.: The memo meta modelling language (mml) and language architecture (icb-research report). Technical report, Institut für Informatik und Wirtschaftsinformatik, Duisburg-Essen, Germany (2009)
25. Frank, U.: Memo: Visual languages for enterprise modelling. Technical Report 18, Arbeitsberichte des Instituts für Wirtschaftsinformatik Koblenz (1999)
26. Frank, U., Jung, J.: The memo organisation modelling language (memo-orgml). Technical Report 26, Arbeitsberichte des Institut für Wirtschaftsinformatik der Universität Koblenz-Landau (2001)

27. Buckl, S., Ernst, A.M., Lankes, J., Matthes, F., Schweda, C.M.: State of the art in enterprise architecture management 2009. Technical report, Chair for Informatics 19 (sebis), Technische Universität München, Munich, Germany (2009)
28. Schelp, J., Winter, R.: Language communities in enterprise architecture research. In: DESRIST '09: Proceedings of the 4^{th} International Conference on Design Science Research in Information Systems and Technology, New York, NY, USA, ACM (2009) 1–10
29. Wegmann, A., Balabko, P., Lê, Regev, G., Rychkova, I.: A method and tool for business-it alignment in enterprise architecture. In: Proceedings of the CAiSE'05 Forum, Porto, Portugal (2005) 113–118
30. Jonkers, H., van Burren, R., Arbab, F., de Boer, F., Bonsangue, M., Bosma, H., ter Doest, H., Groenewegen, L., Scholten, J., Hoppenbrouwers, S., Iacob, M.E., Janssen, W., Lankhorst, M., van Leeuwen, D., Proper, E., Stam, A., van der Torre, L., van Zanten, G.: Towards a language for coherent enterprise architecture descriptions. In: 7^{th} International Enterprise Distributed Object Computing Conference (EDOC 2003), Brisbane, Australia, IEEE Computer Society (2003)
31. Jonkers, H., Goenewegen, L., Bonsangue, M., van Buuren, R.: A language for enterprise modelling. In Lankhorst, M., ed.: Enterprise Architecture at Work. Springer, Berlin, Heidelberg, New York (2005)
32. Ernst, A.M.: A Pattern-Based Approach to Enterprise Architecture Management. PhD thesis, Technische Universität München, München, Germany (2010)