How to make your enterprise architecture management endeavor fail!

Sabine Buckl, Alexander M. Ernst, Florian Matthes, Christian M. Schweda
Chair for Informatics 19
Technische Universität München
eMail: {buckls, ernst, matthes, schweda}@in.tum.de

August 6, 2009

Abstract

Enterprise architecture (EA) management is one of the major challenges of modern enterprises. It aims at aligning business and IT in order to optimize their interaction. In contrast to other EAM patterns, which document proven-practices concerning methodologies (M-Pattern), viewpoints (V-Pattern), or information models (I-Pattern), this article includes two *anti patterns* for EA management. Anti-patterns detail on typical mistakes in EA management, and present revised solutions, which help pattern users to prevent these pitfalls. While the first anti pattern Oversized Information Model deals with problems arising from the usage of a giant information model, the second anti pattern Missing Legend shows why every visualization should provide a legend.

1 Introduction and Overview

Modern enterprises face the challenge to survive in an ever changing environment. In order to support the transformation of the enterprise according to new business demands, legal regulations, or changing market situations a holistic perspective on the enterprise architecture (EA) has to be taken. Thereby, EA is understood as the *fundamental conception of the system [enterprise] in its environment embodied in its elements, their relationships to each other and to its environment, and the principles guiding its design and evolution* [Int07]. Managing the EA is one of the major challenges of modern enterprises. It aims at aligning business and IT in order to optimize their interaction.
Documenting and managing the EA is an advanced topic, as the application landscape, which is part of the EA, often includes a few hundreds up to a few thousand business applications and their interconnections. Thereby, managing the EA is a task, that has to be executed as the need for a flexible IT is an integral concern of most companies. Another reason for the importance of EA management are regulations like e.g. the *Sarbanes Oxley Act* (SOX) [tC02], which determine the information a company has to have available about its EA.
This article includes patterns which are part of the *EAM Pattern Catalog*, a *pattern language for enterprise architecture management* [BEL+07b, BELM08, BEL+08, Ern08], which uses a pattern based approach to EA management. The complete *EAM Pattern Catalog* is available online at http://eampc-wiki.systemcartography.info/ [CfIs09] and currently includes 164

EAM patterns[1]. The intention behind this article is to further extend the existing *EAM Pattern Catalog* by two anti patterns in order to advance the EAM pattern language.
The *EAM Pattern Catalog* introduces four different types of patterns:

M-Patterns specify a methodology to address management problems in a stepwise manner. The procedures defined by the M-Pattern can be very different, ranging from visualizations and group discussions to more formal techniques as e.g. metrics calculations [LS08]. M-Patterns explicate the methodologies in order to complement activities carried out in an ad-hoc manner or relying on implicit knowledge with activities carried out more systematically.

V-Patterns provide visualizations like diagrams, reports, etc., which are practically proven to be adequate to address problems in EA management. The data required to produce the visualization is documented in one or more I-Patterns.

I-Patterns supply best-practice information model fragments, including definitions and descriptions of the used concepts, which can be used to collect information to address a certain problem in EA management.

Anti patterns document typical mistakes made in the context of EA management, and provide revised solutions in order to support the pattern user to prevent these pitfalls.

The term *information model* is used throughout this paper in accordance with [BEL+07a] for the meta-model for modeling the EA.
Relationships are an important aspect in a pattern language. To accommodate this relationships between all four EAM pattern types are possible. For example, Oversized Information Model refers to I-Patterns in its revised solution, or a V-Pattern references one or more I-Patterns documenting the information required to create a view according to the V-Pattern, etc.
The patterns included in the *EAM Pattern Catalog* [CfIs09] follow a template for pattern documentation similar to Buschmann et al. [BMR+96]. Anti patterns have a slightly different structure, which is inspired by [BMM98]. Table 1 shows the pattern form in detail.
In contrast to typical pattern formats, the chosen anti pattern form does not include a *know uses* section, because it is hard to find companies, which agree to be named for a not working practical example.
In addition to the information presented in Table 1 every EAM pattern includes an identifier, versioning information, and status information for managing the patterns. This information is omitted for this article. The same is true for acknowledgments. In this article a separate section (see Section 3.1) includes the acknowledgments for all anti patterns.
The rest of this section list some remarks to writer's workshop participants, gives a short overview about the intended audience, and an overview about included EAM patterns.

[1] For a detailed explanation of the concept of EAM patterns refer to [Ern08].

Name of Section	Content of Section
Description	Short summary of the pattern to get a first look at its content.
Example	An example illustrating the problem to be addressed by the pattern. This example should be used by the other parts of the pattern. Text, which is specifically about the example is especially highlighted.
Context	The situations in which the pattern may apply.
Problem	The problem a pattern addresses, including a discussion about its associated forces. Only one problem per pattern. Forces are *goals* and *constraints*, which occur in the context.
General form	The recurring, not working solution found in practice.
Variants	A brief description of variants or specializations of a pattern if available.
Consequences	The liabilities of the solution documented in the general form.
Revised solution	A revised solution to the problems presented in the general form section.
See also	References to other patterns solving similar problems, and to patterns that help to refine the pattern under consideration.

Table 1: Anti pattern form

1.1 Intended Audience

This article and the herein included patterns are intended for people concerned with creating and maintaining information models and people who have to create and edit EA visualizations, which are used for communication purposes.

1.2 Included Anti Patterns

This article includes the Anti-Patterns Oversized Information Model (see page 4) and Missing Legend (see page 8), which are in this case not presented in form of a pattern map showing their relationships to other patterns, because they apply in more general cases. Oversized Information Model should be considered every time an information model is developed or changed. In contrast Missing Legend should be considered, whenever an EA visualization is created, which is meant to be used for communication between different people.

2 Anti Patterns

This section contains the two anti patterns Oversized Information Model (see page 4) and Missing Legend (see page 8), which are extensions to the *EAM Pattern Catalog* [CfIs09].

2.1 Oversized Information Model

Oversized Information Model shows, why it is not advisable to develop a giant information model for EA management.

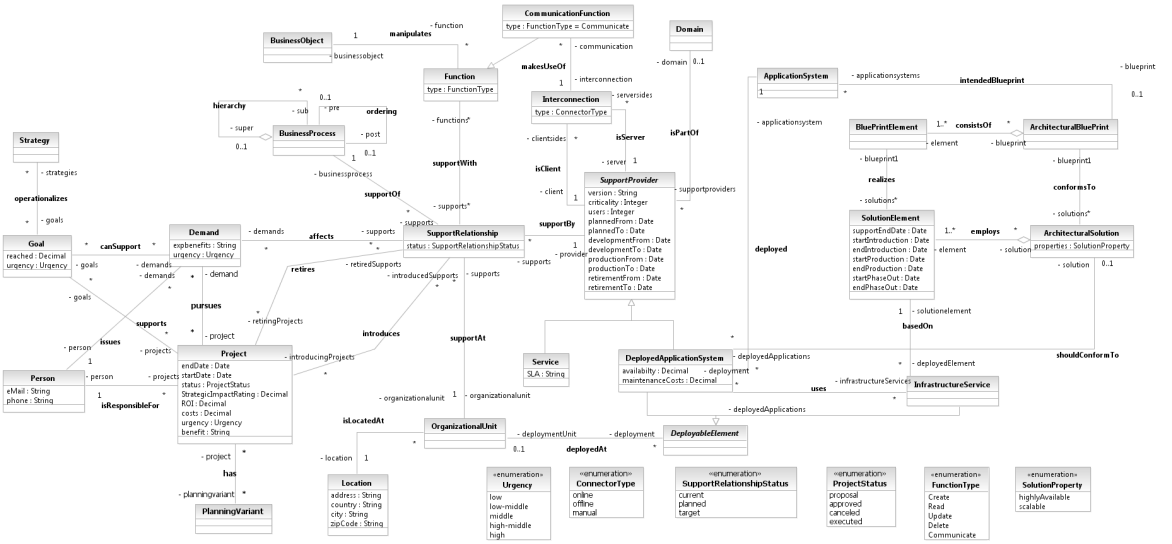

Figure 1: Exemplary Oversized Information Model

2.1.1 Example

The department store *SoCaStore* wants to start its EA management endeavor. An information model provides the basis for the EA management activities and defines which entities, attributes, and relationships should be documented. Because no commonly accepted standard information model is available, SoCaStore has to develop its own enterprise-specific one.

2.1.2 Context

An enterprise, which wants to develop an information model, which fits its specific EA management approach and does not want to rely on predefined ones, as those models do not specifically target the EA management concerns of the enterprise. A predefined information model, as proposed by EA management tools (see [MBLS08] for an overview) or frameworks like TOGAF [The09], might contain concepts to address concerns not relevant to the enterprise or might miss concepts needed for specific concerns.

2.1.3 Problem

How do you create an information model suitable to fit your enterprise-specific needs, like various kinds of analysis or planning your EA?

The following *forces* influence the solution:

- **Company-wide versus team-wide consultation** The creation of an enterprise-specific information model is a collaborative task, strongly influenced by the participating stakeholder group. Hence, the question on the size of the stakeholder group applies, i.e. the question whether only the EA team should participate in the construction or stakeholders from the remainder of the company.
- **All-embracing versus concern-oriented** Which concerns should be addressed by the information model? Should the information model cover the EA in a holistic manner or should it be developed based on the selected concerns to be addressed?
- **Maximal versus minimal** What is a good size for an information model?

2.1.4 General Form

Currently, there is no commonly accepted standard information model for EA management, which satisfies the requirements of all enterprises. As a result most enterprises start to develop their own information model based on their requirements.
In particular, many different stakeholders within the enterprise are asked about their information demands and requirements in respect to EA management. This typically leads to a long list of requirements, resulting in an even larger list of concepts in the respective information model. Due to the length of the list and the high number of stakeholders, which are interested in EA management, this usually results in an Oversized Information Model. An exemplary one is shown in Figure 1.
The same is true for tools for EA management. Their information models have to fulfill the demands of various companies and interest groups. Table 2 shows a short overview about the size of information models implemented in typical EA management tools [MBLS08]. How can you maintain an information model including over 200 classes? In order to alter the information model you should at least know, why the classes and attributes have been introduced into the model. It gets even worse, because such models typically include at least twice as many associations and numerous attributes per class.

Vendor	Number of Classes
A	54
B	220
C	470

Table 2: Overview about information model size based on [MBLS08]

Oversized Information Model can be observed in various companies. A typical symptom is that a second information model is created, which is way smaller then the initial one.

2.1.5 Consequences

Oversized Information Models cannot be maintained in the future, as e.g. the maintaining group neither might be aware of the causes due to which certain concepts were introduced to the model nor might know what the concepts are used for. Another result is that the information stored according to a giant information model cannot be updated in a manner that is required to achieve an adequate information quality. At least if you have to consider the cost benefit ratio for collecting and using the information. The evolution of the information model during the maturation of the EA management endeavor is further hampered by an Oversized Information Model, as the sheer number of concepts and associations in between makes it very difficult to understand the model; thereby, adaptations and changes to the information model are effectively prevented.
When developing an information model, you should therefore execute a prioritization on the collected requirements to concentrate on the most important ones. This initial information model can then be extended in a stepwise manner to keep up with the maturity in EA management of the enterprise. In this case you should document, why the information model has been extended by which concepts or by which I-Patterns [CfIs09] in order to keep the knowledge on the reasons for the previous extensions for future extensions.

2.1.6 Revised Solution

The most important aspect of the revised solution is: Try to avoid Oversized Information Models.
When developing an information model you should first identify the stakeholders, which are relevant for the development. Relevant stakeholders might e.g. be business units as they could serve as future sponsors if the first endeavor is successful or people, who have a current pain, which is addressed in the endeavor.
In a second step try to identify their concerns, which should be addressed by the information model. This can be supported by the EAM pattern approach, which offers I-Patterns documenting information model fragments based on proven-practices [CfIs09], complemented by a description of the used concepts and relationships. After the required I-Patterns to address your concerns were selected, you can integrate and adapt them, e.g. by introducing additional attributes, to achieve your company-specific information model.
The benefit of this approach is that only those concerns are addressed by the information model, which really have to be addressed, leading to smaller and easier to handle models. This corresponds to the *relevance* criteria, proposed by Becker et al. [BRS95] as part of the *Guidelines of Modeling* (GoM). A similar approach is documented in Detailed Enterprise Model in [Amb08].
Another benefit of this approach is that I-Patterns document the problems, which they address including the associated forces. As a result the information model can easier be maintained than an information model, which does not have any documentation why entities, attributes, and associations have been included.

2.1.7 See Also

Oversized Information Model should be considered every time when developing a new or changing an existing information model. [Amb08] describes two anti patterns pointing to

similar problems. The first one is called Detailed Enterprise Model and described as follows:

> The enterprise model(s) are overly detailed, often in an attempt to comprehensive define what the enterprise does (or should do).

Modeling for Modeling's Sake is the second one.

> Someone thought it would be a good idea to develop an enterprise model but did not have a concrete plan for how to use it in practice. Vague ideas that development teams will be able to use the model for guidance aren't sufficient.

The difference between the anti patterns in [Amb08] and Oversized Information Model presented in this paper is basically the extent of its descriptions and that Oversized Information Model subsumes the two anti patterns by [Amb08]. A benefit is that another set of patterns in the context of EA management is referenced and therefore incorporated in the EAM pattern language.

2.2 Missing Legend

Missing Legend shows why every visualization should provide a legend.

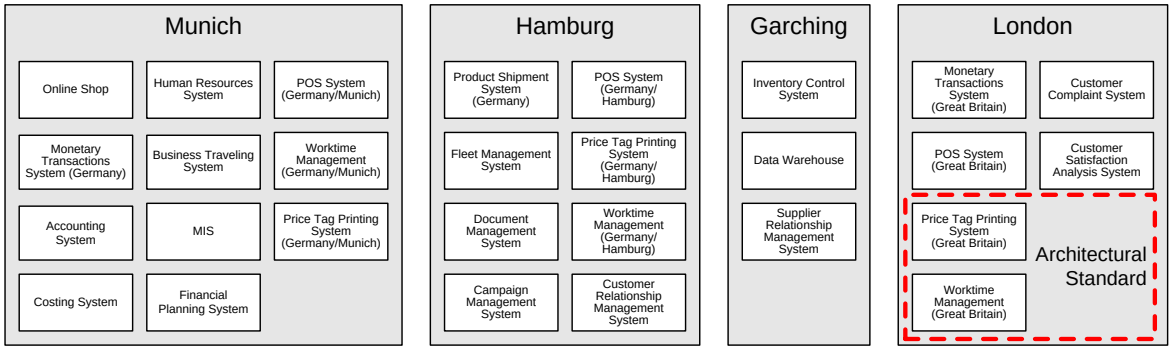

Figure 2: Visualization without legend and additional annotation

2.2.1 Example

The department store *SoCaStore* wants to get an overview about its application landscape, which has grown unplanned for ten years now. To get this task done the management selects an employee from the IT department to create a visualization of the application landscape, which is understandable by everyone within the company and can be used, even if its creator is not available.

2.2.2 Context

Situations where EA visualizations are created or updated to incorporate the new make-up of the enterprise; especially, if the visualizations are used by various people to foster communication about EA management problems.

2.2.3 Problem

How do you create a visualization, which is understandable by a wide variety of people, which were not involved the creation process?

The following *forces* influence the solution:

- **Ease of creation versus utility** Creating a visualization with a legend can be considered more complex to do, than simply *drawing* a graphical model. Nevertheless, the omission of a well-defined notation, made explicit in a legend, greatly reduces the visualization's utility, if reuse by people different from the creator is intended.
- **Standardized notation versus free style** Is it required to use a standardized notation to create comprehensible visualizations?
- **Manual versus automatic creation** Is there any influence of the degree of automation on the understandability of created visualizations?

2.2.4 General Form

The simplest solution for the employee in the above example is to collect the required information about the application landscape and then to just start to draw using simple symbols, like rectangles, circles, or lines.
This typically results in visualizations like the one shown in Figure 2. If you have not been part of the team, which created the visualization, you can just speculate about the meaning of the inner and outer boxes. Why are different colors used and why are some boxes placed within other boxes?
In this example it becomes even worse as someone included an annotation, the dotted box, in the visualization. The dotted box includes the text "Architectural Standard". But what does it mean? Do only the two included white boxes satisfy architectural standards? Do all other boxes represent software not corresponding to architectural standards?
Missing Legend can be observed everywhere and is not bound to any company or person.

2.2.5 Consequences

A negative consequence of Missing Legend is that it is time consuming and error prone starting to interpret a visualization every time you need it. Additionally, if you come to a wrong interpretation, this may lead you to wrong decisions.
If the semantics is not clear you are unable to communicate the information behind a visualization. [CBB+02] includes an illustrating example for this situation.

> "These pictures are meant to entertain you. There is no significant meaning to the arrows between the boxes." A speaker at a recent software architecture conference, coming to a complex but ultimately inadequate boxes-and-lines everywhere viewgraph of her system's architecture and deciding that trying to explain it in front of a crowd would not be a good idea.

2.2.6 Revised Solution

In cases where a visualization is used by multiple people for discussing or communicating problems it is required to add a legend to the visualization.
Figure 3 shows an example for a visualization based on Standard Conformity Exceptions [CfIs09], which includes a complete legend. The legend details about the symbols and colors used within the visualization. Additionally, there is also information available what is meant by the positioning of the used symbols.
For sure it is not possible to answer all questions about a visualization concerning semantics based on the legend, but this is not its goal. It is just meant to clarify the semantics of the visualization.
Even though, the author of the view is not available any more it can still be used as the meaning is clearly described by the legend. For this reason the visualization can be used as a reliable source of information.
Using tools to create visualizations, may help to include an appropriate legend to any visualization, especially if they are generated automatically. In these cases the tool already has all required information to generate a legend, because the information is needed for the generation of the visualization itself. Nevertheless, not all EA management tools currently support the creation of a legend. [MBLS08]

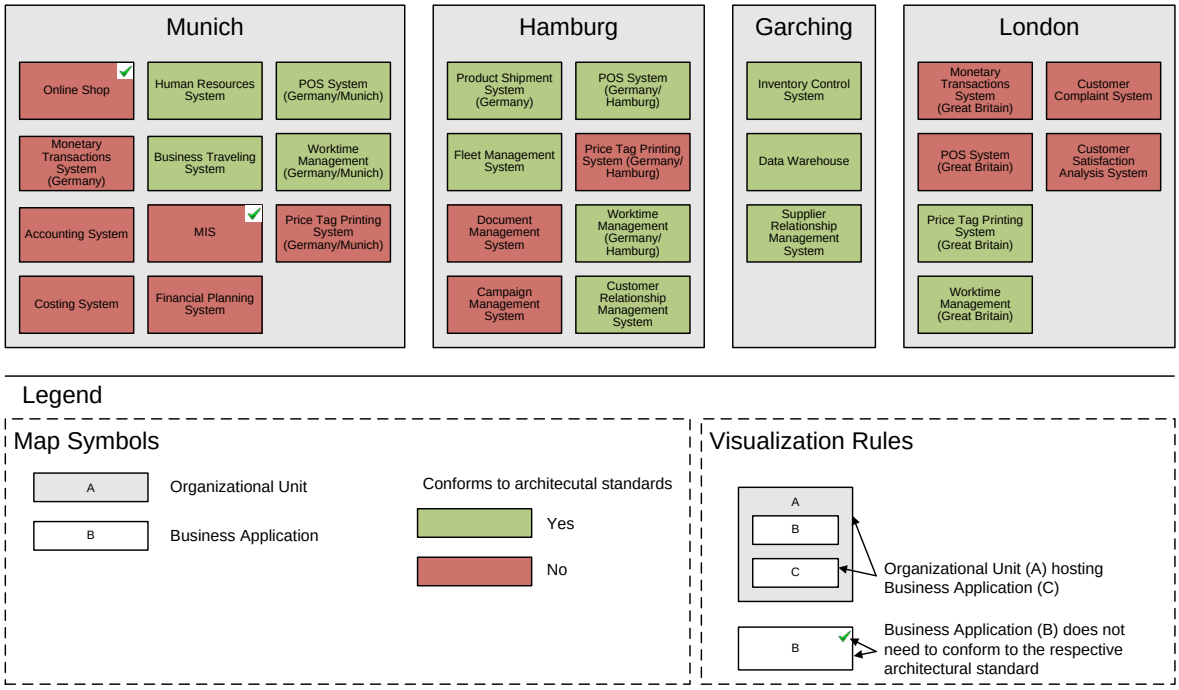

Figure 3: Visualization with included legend

Another way to resolve the problem of having to include a legend is to use a standardized notation, like e.g. the Unified Modeling Language (UML) [OMG05], etc. because the notation should be known or is it least documented.

2.2.7 See Also

Missing Legend should be considered whenever a visualization is created, which should be used for communication means between various different people. Especially if the user of the visualization is someone else then the author.

3 Acknowledgment and Outlook

This section includes acknowledgments to the people who supported the creation of this article and gives an outlook to the next steps in the development of the EAM pattern approach.

3.1 Acknowledgments

We want to thank all participants of the writer's workshop of PloP09 and especially our shepherd Hironori Washizaki for the time they spent for reading, commenting, and discussing this article.

3.2 Next Steps in EAM Pattern Approach Development

The *EAM Pattern Catalog* is available at http://eampc-wiki.systemcartography.info/, based on the results of an extensive online survey and some articles on EAM patterns, see e.g. [Ern08, BEK+09, LFB+09, MJBS09, BEMS09]. In order to improve the current version and to further exploit the advantages of patterns in EA management, an excerpt of the *EAM Pattern Catalog* had been included in this document to be discussed in the pattern community.

References

[Amb08] Scott W. Ambler. Enterprise modeling anti-patterns. `http://www.agilemodeling.com/essays/enterpriseModelingAntiPatterns.htm` (accessed at 2009-04-07), 2008.

[BEK+09] S. Buckl, A. Ernst, H. Kopper, R. Marliani, F. Matthes, P. Petschownik, and C. M. Schweda. Eam pattern for consolidations after mergers. In *SE 2009 - Workshopband*, Kaiserslautern, 2009.

[BEL+07a] Sabine Buckl, Alexander M. Ernst, J. Lankes, Florian Matthes, Christian Schweda, and André Wittenburg. Generating visualizations of enterprise architectures using model transformation (extended version). *Enterprise Modelling and Information Systems Architectures – An International Journal*, 2(2):3 – 13, 2007.

[BEL+07b] Sabine Buckl, Alexander M. Ernst, Josef Lankes, Kathrin Schneider, and Christian M. Schweda. A pattern based approach for constructing enterprise architecture management information models. In *Wirtschaftsinformatik 2007*, pages 145 – 162, Karlsruhe, Germany, 2007. Universitätsverlag Karlsruhe.

[BEL+08] Sabine Buckl, Alexander Ernst, Josef Lankes, Florian Matthes, and Christian M. Schweda. Enterprise architecture management patterns – exemplifying the approach. In *The 12th IEEE International EDOC Conference (EDOC 2008)*, Munich, 2008. IEEE Computer Society.

[BELM08] Sabine Buckl, Alexander M. Ernst, Josef Lankes, and Florian Matthes. Enterprise Architecture Management Pattern Catalog (Version 1.0, February 2008). Technical report, Chair for Informatics 19 (sebis), Technische Universität München, Munich, Germany, 2008.

[BEMS09] Sabine Buckl, Alexander Ernst, Florian Matthes, and Christian Schweda. Enterprise architecture management pattern for ea visioning. In *Proceedings of EuroPLoP 2009*, 2009.

[BMM98] William J. Brown, Raphael C. Malveau, and Thomas J. Mowbray. *AntiPatterns: Refactoring Software, Architectures, and Projects in Crisis*. Wiley, New York, USA, 1998.

[BMR+96] Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, and Michael Stal. *Pattern-oriented software architecture: a system of patterns*. John Wiley & Sons, Inc., New York, NY, USA, 1996.

[BRS95] Jörg Becker, Michael Rosemann, and Reinhard Schütte. Grundsätze ordnungsmäßiger modellierung. *Wirtschaftsinformatik*, 37(5):435–445, 1995.

[CBB+02] Paul Clements, Felix Bachmann, Len Bass, David Garlan, James Ivers, Reed Little, Robert Nord, and Judith Stafford. *Documenting Software Architectures: Views and Beyond*. Addison Wesley, Boston, 2002.

[CfIs09] Technische Universität München Chair for Informatics 19 (sebis). Eam pattern catalog wiki. http://eampc-wiki.systemcartography.info (cited 2009-04-01), 2009.

[Ern08] Alexander Ernst. Enterprise architecture management patterns. In *PLoP 08: Proceedings of the Pattern Languages of Programs Conference 2008*, Nashville, USA, 2008.

[Int07] International Organization for Standardization. Iso/iec 42010:2007 systems and software engineering – recommended practice for architectural description of software-intensive systems, 2007.

[LFB+09] Armin Lau, Thomas Fischer, Sabine Buckl, Alexander M. Ernst, Florian Matthes, and Christian M. Schweda. Ea management patterns for smart networks. In *SE 2009 - Workshopband*, 2009.

[LS08] Josef Lankes and Chrisitian M. Schweda. Using metrics to evaluate failure propagation and failure impacts in application landscapes. In *Multikonferenz Wirtschaftsinformatik*, Berlin, Germany, 2008. GITO-Verlag.

[MBLS08] Florian Matthes, Sabine Buckl, Jana Leitel, and Christian M. Schweda. *Enterprise Architecture Management Tool Survey 2008.* Chair for Informatics 19 (sebis), Technische Universität München, Munich, 2008.

[MJBS09] Christoph Moser, Stefan Junginger, Matthias Brückmann, and Klaus-Manfred Schöne. Some process patterns for enterprise architecture management. In *SE 2009 - Workshopband*, 2009.

[OMG05] OMG. Unified modeling language: Superstructure, version 2.0, formal/05-07-04, 2005.

[tC02] 107th Congress. Sarbanes-oxley act of 2002 (public law 107–204). http://www.sec.gov/about/laws/soa2002.pdf, 2002.

[The09] The Open Group. TOGAF "Enterprise Edition" Version 9. `http://www.togaf.org` (cited 2009-07-10), 2009.