A Pattern based Approach for constructing Enterprise Architecture Management Information Models

Sabine Buckl, Alexander M. Ernst, Josef Lankes, Kathrin Schneider, Christian M. Schweda

Software Engineering betrieblicher Informationssysteme
Institut für Informatik
Technische Universität München
Boltzmannstraße 3, 85748 Garching
{buckls, ernst, lankes, schweda}@in.tum.de
{kathrin.schneider}@bmw.de

Abstract:
This paper sketches an approach to designing organization-specific information models for Enterprise Architecture (EA) Management based on patterns. Thus we intend to support the construction of *EA Management* information models in research and practice, a field we view in need of approaches to manage the complexity of such models.
Contributing to this complexity are the wide spread domains (e.g. processes, technical architecture, strategic issues) that are involved in *EA Management*. Moreover, this complexity burdens in practice both using existing information models and creating a new information model.
This necessitates approaches to manage this complexity. Our contribution in this respect lies in the introduction of patterns into the field of *EA Management* information models. Thereby, other approaches, as e.g. structuring the information models into layers, are complemented by the possibility to reuse pre-existing solutions to address *EA Management* issues.

1 Motivation

The following article identifies problems in approaches currently pursued in designing information models supporting *EA Management* and presents a solution based on composing such information models from patterns. Subsequently, we rely on the following definition of *EA Management*, which is in accordance to [ELSW06]:
"*EA Management* is a continuous and iterative process controlling and improving the existing and planned information technology (IT) support for an organization. The process not only considers the IT of the enterprise, but also business processes, business goals, strategies, etc.

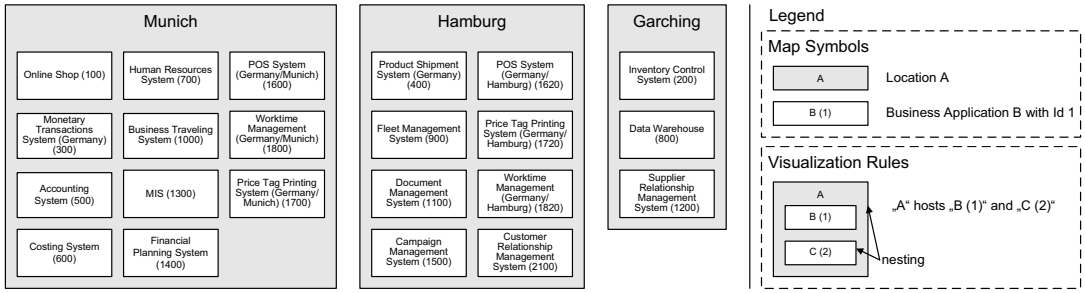

Figure 1: Exemplary software map showing which business application is hosted at which location

are considered in order to build a holistic and integrated view on the enterprise.

The goal is a common vision regarding the status quo of business and IT as well as of opportunities and problems arising from these fields, used as a basis for a continually aligned steering of IT and business."

In order to effectively manage the EA, which includes planning and designing the EA, a systematic documentation of the elements contained therein is often regarded as an important prerequisite. Such documentations are the focus of our research project software cartography (see e.g. [LaMW05a]), in which we develop, in co-operation with our project partners (among others BMW, HVB Systems, T-Com), methods and models for documenting, planning, and designing application landscapes[1], which make up an essential part of an EA. During our research, we discovered a large number of different visualizations for application landscapes, which we call software maps. In practice, besides different kinds of software maps, as described in [LaMW05a], also other visualizations, e.g. portfolio matrices used in the context of project portfolio management, or textual descriptions, as e.g. tabular reports, can be used to support *EA Management*. The exemplary software map in Figure 1 shows via nesting rectangles, which business application is hosted at which location. Thus, it shows information conforming to the information model fragment shown in Figure 2.

Hereby, we define an *EA Management* information model as a model which specifies, which information about the enterprise architecture, its elements and their relationships should be documented, and how the respective information should be structured. This is usually achieved via a model expressed in a language suitable for conceptual modeling, as e.g. UML, enriched with descriptions detailing the exact meaning of the concepts.

Considering the different aspects of *EA Management* in an information model in practice leads to information models, that are according to [Sebis05] likely to contain easily over 100 different classes. Due to the complexity and amount of relevant information, the employment of tools for documenting the EA is according to e.g. [Fran02] required. A central part of such tools is the

[1] In our research project we define application landscape as "the entirety of all business applications and their relationships in an organization".

Figure 2: Information model corresponding to the software map shown in Figure 1

information model. The information models provided by tools for supporting *EA Management* were examined as part of the *EA Management* Tool Survey [Sebis05] conducted in the research project software cartography, helping us to identify several problems typically connected to creating and using information models.

Based on our experiences from the *EA Management* Tool Survey and insights gained from existing information models used by our project partners, we propose, as mentioned above, a concept based on information model fragments called *patterns*, which are developed particularly for specific concerns and recurring problems. These patterns can then be used as building blocks to compose an organization-specific information model.

The remaining article is structured as follows: Section 2 gives an overview of literature and approaches in practice concerning *EA Management*. Section 3 outlines problems of common information models and the approaches pursued in their creation. Our pattern based approach for information modeling is described in section 4 and illustrated by an example in section 5. The last section 6 summarizes the paper and provides a short outlook on planned research activities, which further purse our pattern based approach.

2 Related Work

According to [LaWe04], who conducted an extensive literature review about *EA Management*, first publications in this field go back to [Zach87], but only recently the number of articles published about the subject has increased, indicating that the topic is entering mainstream interest. Thus, different organizations have created frameworks providing guidance for *EA Management*, as for example *The Open Group* (TOGAF [TOG05]), *Meta Group* (Enterprise Architecture Desk Reference [META02]), or the *US Department of Defense* (DoDAF [DoD04]). While those frameworks provide high-level guidance and some of them add a process model for *EA Management*, they do not detail how specific tasks should be carried out and do not provide detailed information models suitable for supporting these tasks.

Such details are of course implemented in *EA Management* tools, which may support one or more of the frameworks mentioned above. Vendors taking their *EA Management* approaches to standardization, as e.g. *Adaptive, Ltd.*, which is a major contributor to OMG's *IT Portfolio*

Management Facility (ITPMF) [OMG06], add to published approaches created by academia. The *ArchiMate Project*, for example, proposes a notation and viewpoints for *EA Management* in [Lank05]. [BrWi05] offers an EA metamodel with over 50 classes, managing the arising complexity by structuring the model in layers. The metamodel for *EA Management* supplied by [Fran02], which supports *planning, designing, and maintaining corporate information systems*, is structured via a language architecture integrating partial models for special purposes as e.g. *strategic modeling* or *modeling organizational structures*. While these languages might be seen as similar to patterns in the sense of our approach, [Fran02] is more focused on providing an integrated metamodel by putting together these languages. Subsequently, we complement such approaches by describing a way for documenting information model patterns, which can then be integrated to a comprehensive, organization-specific information model.

3 Problems of *EA Management* Information Models and their Approaches

As described above, we are trying to complement existing approaches for structuring the complexity of *EA Management* information models by utilizing patterns, as the benefit of existing models seems to be reduced by the following tendencies:

- Research projects seem to prefer developing new models instead of improving existing ones. The same is true for visualization techniques for *EA Management* specific information and methodologies prescribing procedures for working with this information and its visualizations. There are hardly any well-known results in the field that are further developed and verified by research or other practitioners, at least not by a larger community. Each project seems to start developing its own approach.

  This tendency does not universally exist in other areas. In software engineering for example, there are publications that build on UML as a visualization of software structures and try to improve the readability of such diagrams, e.g. by finding advantageous layout criteria [PMCC01]. Contrary, the state of the art regarding enterprise modeling is described as *hardly coherent* by [Fran02].

- Projects in practice executing *EA Management* activities seem to neglect *EA Management* information models made available by research, unless a joint project with academia is executed or an information model is incorporated in the *EA Management* tool used. While such tools contain information models, visualizations, and methodologies, practitioners seem to experience problems therewith, as e.g. shown by the fact, that a major share of the project partners we have talked to in our research project *software cartography* uses *Microsoft Visio* or *Microsoft PowerPoint*-

based visualizations [LaMW05a] developed and used after self-defined guidelines[2] in their *EA Management* activities. Another practitioner states his experiences with *EA Management* information models and tools as follows [Riih05]:

"[...] we have tried out several tools in this area, without much success. [...] Their metamodels are rather complex, but not integrated within themselves."

These tendencies describe a situation in which *reinventing the wheel* seems to be a common practice in the creation of *EA Management* information models, both in practice and in academia. This is surprising, as we experienced, detailed in [Sebis05], that some questions, e.g. "which business application system supports which process at which organizational unit" are common to *EA Management* and should therefore also be covered by information models supporting *EA Management*. Additionally, the tendencies hamper the development of best practices, standards, and significant research results regarding *EA Management* information models, leaving the state of art in this field unable to satisfy the *need to use proven models*, which is e.g. indicated by [Bern03].
Sections 3.1 to 3.3 discuss issues in common *EA Management* information models and approaches guiding their creation, which possibly contribute to the above tendencies.

3.1 Model Size: Giant vs. Midget Models

According to [ELSW06] *EA Management* information models seem to be in danger of falling into one of two traps. On the one hand, small information models with brief documentation hardly deliver benefit to an *EA Management* project. On the other hand, all-embracing information models that cover almost all kinds of information that can be relevant to *EA Management* are also not without their problems. Table 1 gives an overview of the number of classes in the shipped metamodels of the tools covered by [Sebis05].

Number of classes	Number of tools
not known	2
up to 50	2
50 – 350	3
350 or more	2

Table 1: Overview of the information model sizes encountered in [Sebis05]

In most organizations, not all features of an all-embracing information model might be of relevance, due to the focus of a given *EA Management* project on specific issues or the sheer fact that an organization is too small to profit from explicitly managing certain facets of its EA. In

[2] In these cases, the information model can be seen as not being made explicit, giving the visualizations a drawing-like quality, as e.g. described in [ELSW06].

such cases, the disadvantages connected to the size of the information model are likely to outweigh its benefit. Such disadvantages center around understanding and using the model, as well as making organization-specific modifications to it. [Bern03], for example, states this complexity issue of *EA Management* information models. These disadvantages prevail, even if some parts of the information model can be hidden by the *EA Management* tool used (cf. [Sebis05]), as someone has to understand a big model at first in order to determine what has to be hidden and what has to be modified to fit the needs of the organization under consideration.

Additionally, giant and midget models seem to be no popular objects of research, contributing to the tendency of *EA Management*-specific research to build new information models instead of improving existing ones. While midget models often barely contain enough substance, which can be targeted by research and be used as a basis for proposing research questions or hypotheses, the situation also easily gets troublesome with the giant models. Subjecting certain aspects of an *EA Management* information model to research can be difficult in the complex web of an all-embracing information model.

3.2 Giving explicit Account for Model Utility: Stockpiling useless Information

Regarding architectural descriptions, conventions as e.g. the IEEE Std. 1471-2000 (Recommended practice for the architectural description of software intensive system) [IEEE00] state that the creation of architectural views has to be justified by the existence of stakeholders, whose concerns can be addressed by the respective views. This paradigm makes even more sense in the context of EAs, as the collection of the respective information, which often has to be organized in a decentralized fashion, is more difficult in this field than in the documentation of single application systems [LaMW05b]. Introducing certain data structures into an *EA Management* information model, thus creating the demand to collect the respective information, should therefore be justifiable by adequate concerns being addressed using this information.

Thus, failing to provide the information due to which concerns certain data structures are used in an information model, which is not an unrealistic danger in practice according to [Sebis05], leaves difficulties to decide which concepts are really relevant for addressing specific problems in a specific organization. This could contribute to the tendency that the respective information model is only hesitantly used in practice, especially, when this intransparency occurrs in giant-like information models.

Failing to provide the concerns also constitutes an obstacle to research regarding *EA Management* information models. When evaluating the suitability of a certain model in order to use this as a basis for reasonable proposals for improvement, it seems beneficial to know what exact problems are to be addressed by this model.

3.3 Missing Methodologies

Even if it is known or somehow guessed from insufficiently explicated concerns, which data structures are intended to be taken into consideration when addressing a specific concern, it may not be clear how these structures should be used. While many information models seem to intend the information to serve for the creation of visualizations or reports that are meant to make the EA more transparent, also other use cases are possible, as e.g. metrics calculation or automated checking for violation of certain constraints.
The practical usage of an information model is made more difficult, if the methodologies for addressing concerns based on this information are left out. [Bern03] reinforces the importance and benefit of knowing the usage context of *EA Management* models. In case they are missing, methodologies have to be added by the users in the project introducing the information model, leading to additional effort. Also, descriptions of the intended data usage could often help to clarify what is meant by the abstract concepts[3] forming the classes and relationships in an *EA Management* information model.
The advantage explicit methodologies could give to research regarding EA information models directly connects to the point made in section 3.2, as the evaluation whether an information model is useful for addressing certain concerns also has to take into consideration, how these concerns are meant to be addressed using the respective information model. Hereby, it becomes clear that an *EA Management* information model should not be seen in isolation. It is only complete together with *viewpoint definitions*, offering a problem-adequate way of visualizing data from the information model, and *methodologies*, which describe procedures how information model and viewpoint can be used to address specific concerns.
Especially the importance of methodologies can be substantiated via the methodology definition according to [KrSH93], which states that
"Within an engineering discipline, a method describes a way to conduct a process. In the context of systems engineering, a method is defined as consisting of:

- An underlying model (which refers to the classes of objects represented, manipulated and analysed by the method)
- A language (referring the concrete means of describing the products of the method)
- Defined steps and ordering of these steps (which refers to activities performed by the user of the method)
- Guidance for applying the method"

[3]The frequently and incoherently used term "service" can serve as an example here. While there is hardly a common definition of this term, mutual understanding about it can increase significantly, if it is explicitly stated what is about to be done with services. Knowing that e.g. application development is the subject can add to the understanding regarding the term.

Of course the tendencies discussed above may be impacted by the fact that *EA Management* is a relatively young discipline, with mature standards and best practices therefore yet to come. But we view that the issues outlined in sections 3.1 to 3.3 might well obstruct the development of exactly these standards and best practices.

4 Pattern based Approach

Addressing the issues encountered in *EA Management* information models, which we regard to partially be arising from the approaches taken during information modeling, we propose an approach based on *patterns*. This concept is already used in various domains, as e.g. in architecture or in software design [GHJV94], where a pattern is defined as follows:
"A (design) pattern names, abstracts, and identifies the key aspects of a common design structure that make it useful for creating a [...] design. The design pattern identifies the participating classes and instances, their roles and collaborations and the distribution of responsibilities. Each design pattern focuses on a particular [...] design problem or issue. It describes, when it applies [...], and the consequences and trade-offs of its use."
Defined like this, a pattern describes a basic idea, that has turned out to be useful for addressing specific concerns in a practical context and in a generalized way. Due to this generalization, a pattern can be adapted and reused in related contexts. Besides this, patterns are defined implementation-independent, constituting a blueprint for a solution actually to be implemented.
We regard *EA Management* patterns as building blocks for the concept of an organization-specific support for *EA Management*, consisting of a conceptual information model, viewpoints and methodologies for using the respective information. Thereby, the *EA Management* patterns themselves are organization independent, based on best practices developed by academia and practice and hence are possibly of benfit to a wide range of users, offering the possibility to further improve existing patterns in an environment without organization-specific restrictions. *EA Management* patterns can furthermore be combined to one or more catalogues, which can be maintained by a community in order to control and foster the further development, like e.g improvements of existing or the introduction of new patterns. This community can additionally make sure that the advantages of this approach can better be communicated than in the case of solutions created by single organizations or research groups.
Figure 3 sketches the ideal type of project[4] concerned with *EA Management* support utilizing *EA Management* patterns. Such a project uses one or more catalogues of *EA Management* patterns, supplied by *pattern designers*, as a basis. From these catalogues, the *developers* for *EA Management* support choose patterns, that are perceived as adequate for addressing specific concerns

[4]Figure 3 sketches a waterfall-like approach, but of course also iterative approaches can be taken.

of the respective organization, preferably under participation of the prospective *users*. After integrating these *EA Management* patterns, thereby creating a coherent organization-specific conceptual model, the respective concepts can be implemented, e.g. in an *EA Management* tool or a suite of tools, that fit the requirements of the organization under consideration.

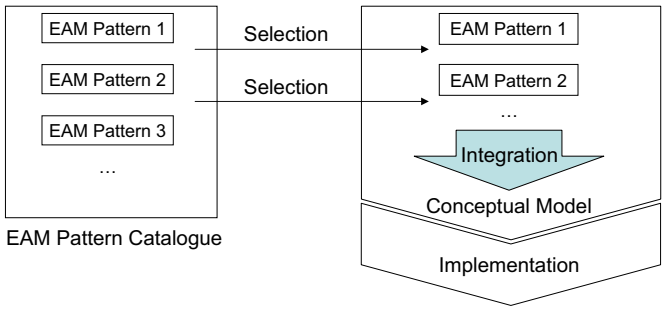

Figure 3: Implementing an *EA Management* approach based on *EA Management* patterns

Thus, the pattern based approach tries to avoid the pitfalls described in sections 3.1 to 3.3, as the *EA Management* patterns form small, reusable units preferably based on established practices that are used, when suitable for addressing specific concerns.

4.1 Basic Structure: A three-tiered Approach

In order to not solely describe the information under consideration in a specific pattern, but also to make the methodologies for using the respective data explicit and therefore avoiding pitfalls as described in section 3.3, we view it important to have descriptions with the information model patterns that delineate, how the data stored should be presented and used to address specific concerns, similar to the concepts of *language* and *steps for applying the method* in the methodology definition according to [KrSH93] in section 3.
Hence, we decided to build an approach on three different kinds of *EA Management* patterns:

Methodologies *defining steps* to be taken in order to address given concerns. Furthermore, as a guidance for applying the method, statements about its intended usage context are provided, which include the concerns to which the methodology can be applied. These concerns are addressed by procedures defined by the methodology, which can be very different, ranging from e.g. visualizations and group discussions to more formal techniques as e.g. metrics calculation.

Viewpoints providing the *languages* used by methodologies. A viewpoint proposes a way to present data stored according to one or more information model patterns.

Information model patterns supplying *underlying models* for the data visualized in one or

more viewpoints. An information model pattern conveys an information model fragment including the definitions and descriptions of the used information objects.

The three kinds of *EA Management* patterns are closely tied to each other, via relationships as shown in Figure 4. Templates, which serve to provide exemplary outlines for documenting and describing the patterns in an in-depth and structured way, can guide the compilation of actual patterns. An detailed description of the template regarding information model patterns can be found in the next section, while this article, focusing on information models, does not further elaborate on methodologies and viewpoints. Each *EA Management* pattern has to be adapted to the organization-specific context and combined with other *EA Management* patterns in order to make contributions to a specific concept for *EA Management* support.

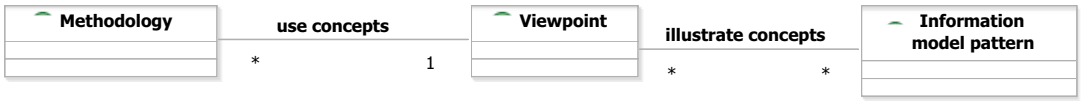

Figure 4: Methodology, viewpoint, and information model pattern

4.2 Documentation and Usage of Information Model Patterns

According to the three-tiered structure of our *EA Management* pattern approach, described in the previous section, the *information model pattern*, its structure, and its usage is detailed below. An exemplary outline for documenting information model patterns can be seen in the template in Table 2. It is organized in three main parts, an *overview section*, a *solution section*, and a *consequence section*, similar to the basic elements of a pattern description stated by [GHJV94].

Overview section	
Id	An unique alphanumerical identifier
Name	A short and expressive name for the pattern
Alias	Names this pattern is also known as (optional)
Summary	A short summary of the pattern of about 100 words
Solution Section	
Information Model	An *information model fragment* in a certain language (see section 4.2.1), together with additional documentation
Consequence Section	
Appliance	Guidance on how to use the information model pattern
Consequence	Consequences resulting from the usage of the pattern

Table 2: Template structure of the Information Model Pattern

4.2.1 Languages for Information Model Fragments

Documenting an information model fragment always relies on a certain language, in which the elements of the fragment are expressed. Basically a lot of different languages suitable for that kind of conceptual modeling are known. The following list is intended to show some of the more prominent examples, outlining both advantages and disadvantages of the languages:

Textual Description in Natural Language: The model elements and their relationships are described in natural language. This seems to be a good choice as it produces easily understandable and adaptable descriptions. The main disadvantage is that this kind of documentation easily leads to mistakable constructs, insufficient for exactly defining information model fragments.

Meta Object Facility (MOF) and Unified Modeling Language (UML) Class diagrams: The model elements and their relationships are described via object-oriented concepts, captured e.g. in the UML or MOF 2.0 metamodel. This description can rely on UML class diagrams, which should be usable by most developers. A disadvantage of these languages is that they lack a formal basis and hence possibly limit the domain in which these kind of information model fragments can be used.

Ontology Languages: The model elements and their relationships are described in terms of an ontology language. Such a language might provide more expressiveness than the approach of MOF or UML and is based on a formal foundation including the possibility to use automated reasoning in the model. Disadvantageous is, that ontology languages are not so widespread in the area of conceptual modeling.

Mathematical Formalization: The model elements and their relationships are described in mathematical or logical terms, providing a strong formal background. The main disadvantage is, that mathematical or logical models are usually difficult to use.

Of course, the possibility to combine two or more of the variants described above could unleash the advantages of the languages combined, possibly without having to consider the individual disadvantages. Nevertheless, one disadvantage seems inevitably related to such a combined approach, namely the effort for describing an information model fragment in both languages and keeping these descriptions consistent.

4.2.2 Language Choice and Documentation Guidelines

With different possible languages for describing information model fragments and considering their individual advantages and disadvantages, some languages may be more or less adequate for a specific pattern. This is especially obvious, if the information model fragment under

consideration is only used for creating a visualization of the application landscape or a tabular report, where an object-oriented description should be sufficient. For other usecases, e.g. the calculation of metrics or the simulation of processes, the situation looks different, as this may only be possible in a reasonable way if the information model has a more formal basis.

Therefore, we propose using a language adequate to the problem addressed, strongly suggesting to use UML as the language of choice[5], as this language is widely understood and has been found by us as problem-adequate in many situations. For example, this language is commonly used in the information models we found in our research project *software cartography*.

There is also a second reason for suggesting *one* language. Utilizing the pattern based approach for information modeling, a crucial point is the *integration* of information model patterns to form a complete information model. This integration can be regarded more simple, if the different patterns are specified using one language, which cuts the effort of translating the models between languages.

Nevertheless, there might be situations in which it seems advantageous to use a language different from the default one, as e.g. UML would lead to a documentation far to extensive or not expressive enough for the problem at hand and therefore would not be problem adequate. In such cases it can be advisable to support the information model fragment modeled in a problem adequate language with a corresponding information model fragment documented in the default language, as this simplifies the integration of the patterns.

As a last documentation guideline we would like to advise the *pattern designer* to complement each information model fragment with a textual description promoting understandability.

4.2.3 Integrating Information Model Patterns

Aspects of integrating information model patterns during the creation of an organization-specific information model are not detailed in this article. Nevertheless, two integration approaches are subsequently sketched - relying on experience in conceptual modeling.

The steps for achieving an integration of information model patterns are similar to the ones applied in software engineering. If UML is used for documenting the information model pattern, a simple way to integrate two information model patterns is to identify one or more identical classes within both patterns. These classes can than be used as a point of integration. Another approach involves introducing a new relationship between two classes from different patterns. Potentially this could employ inheritance, e.g. the class *business application* from one pattern could inherit life cycle attributes from a class *LifecycledElement* introduced in another pattern.

[5] In the following called *default language.*

5 Example of the *EA Management* Pattern Approach

This section exemplifies the pattern-based approach by describing patterns to address concerns related to redundant applications in an application landscape. Thereby, the viewpoint and the methodology are sketched briefly, with the focus lying on the information model pattern, as this constitues the core subject of the article. Due to readability reasons we introduce the viewpoint first and then complement it with its underlying information model pattern.

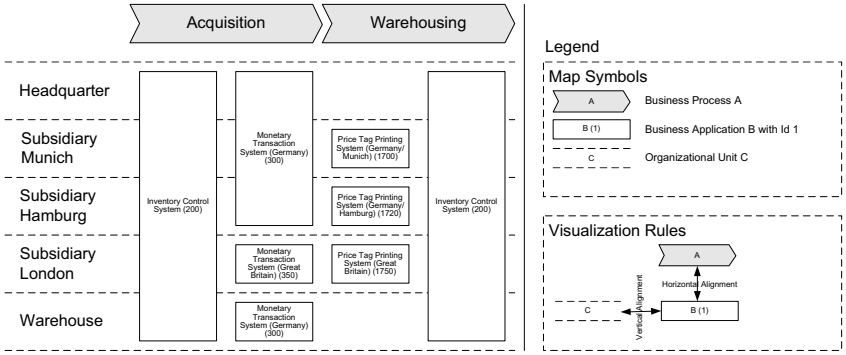

Figure 5: Exemplary process support map

5.1 Viewpoint: Process Support Map (Id V-PSM-1)

Figure 5 shows a *process support map*, a kind of software map we discovered as a visualization used for addressing concerns as mentioned above in practice [LaMW05a].
Basically, the process support map is a software map utilizing positioning of symbols to show, which business processes are supported by which business applications in which organizational units. Thereby, chevrons visualizing a process chain, seen as a sequence of processes, make up the x-axis. The y-axis is made up of labels representing organizational units. The rectangles in the main area of the map symbolize business applications, and their positioning expresses which process is supported by which business application in which organizational unit. Information model pattern *I-BPS-1*(section 5.2) supplies the concepts on which the process support map is built. Methodology *M-ARI-1*(section 5.3) works on visualizations conforming to this viewpoint.

5.2 Information Model Pattern: Business Process Support (I-BPS-1)

According to section 4.2, the information model pattern is described in three parts.

Overview section:

Id:	I-BPS-1
Name:	Business Process Support

Alias:	IT Building Plan
Summary:	The pattern provides a structure suitable for organizing information about which business applications support which business processes in which organizational units.

Solution section

As this pattern is mainly constructed to serve methodologies based on visualizations that have no strict background in formal methods, it is described in UML, backed up by textual definitions and explanations. The concepts of figure 6 are defined as follows:

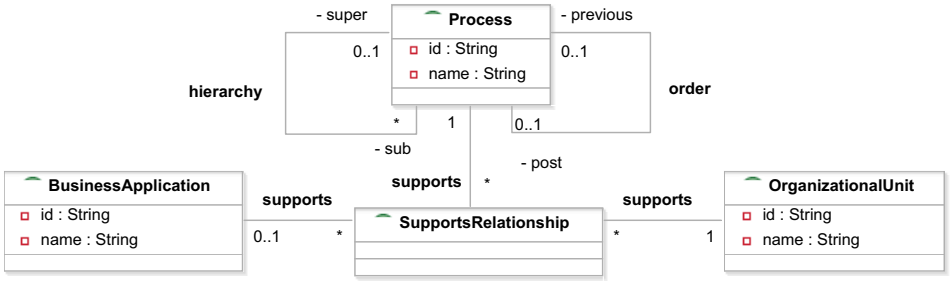

Figure 6: Information model fragment of the business process support pattern

Process: A process can, according to [Krcm05], be defined as a sequence of logical individual functions with connections between them. [DiFH03] states input and output factors and a defined process objective as important characteristics of a process. The process class here should not be identified with single process steps or individual functions, but with high-level processes at a level similar to the one used in value chains. The process class owns the following attributes[6]:

id unique alphanumeric identifier

name a descriptive name of the process

OrganizationalUnit: An organizational unit represents a subdivision of the organization according to its internal structure. E.g. the entities showing up in an organigram can be used as organizational units.

Business Application: A business application is a software system which is part of an information system of an organization. An information system is according to [Krcm05] understood as a sociotechnical system, which is, besides the software system, made up of the infrastructure the software system is based on, and a social component, namely the employees or stakeholders concerned with it. Thereby, infrastructure

[6]The corresponding attributes of the other classes are subsequently not further detailed.

and social component are not considered as belonging to the business application, while the characterization "business" restricts the term to applications that support at least one process of the respective organization.

SupportsRelationship: The class SupportsRelationship represents the support of a process by a business application at a specific organizational unit. Basically, it constitutes, together with its three associations, a ternary relationship between *BusinessProcess*, *OrganizationalUnit*, and *BusinessApplication*. This is necessary in order to be able to tell exactly which organizational unit uses which business application to support a given process. The class is utilized here instead of directly relying on the ternary relationship in UML in order to allow adaptations of the fragment like e.g. introducing attributes to the relationship. This might e.g. be useful if the relationship is time-dependent.

The associations of the information model fragment serve the following purposes:

hierarchy: A process can be part of a larger, encompassing process (super-process) and can include other processes (sub-processes).

order: A process, seen on the level as stated above, can be part of a value chain, which is an (at least partially) ordered sequence of processes. Thus, a process can have a predecessor process and a successor process, expressed by the order relationship.

supports: Instances of the class *SupportsRelationship* use links according to the three associations that are called *supports* to indicate that a business application supports a certain process at a specific organizational unit.

Consequence section:

A possible *appliance* is viewpoint *V-PSM-1*, with its attached methodologies, which is able to visualize information structured according to this information model pattern[7].

An important *consequence* of this pattern is the challenge of collecting the necessary data. Existing business applications and the execution of processes at different organizational units might be considered well-known in most organizations. Contrary, the support of an business application for a specific business process is more often only implicit knowledge of certain employees, which might considerably add to the burden of information gathering.

---

[7] Of course other visualizations are possible, but not detailed here.

5.3 Methodology: Business Application Redundancy Identification (Id M-ARI-1)

This methodology briefly describes a procedure to address the concern *business application redundancy identification*. Redundancy in this context means, that the same functionality supporting a process, may be realized by different business applications in different organizational units.

In order to contribute to addressing this concern, the user should employ the viewpoint *V-PSM-1* (see section 5.1) to identify processes, where the different organizational units use different application systems to support the same process. An example for this can be seen in Figure 5, where the process *Acquisition* is supported by two different business applications named *Monetary Transaction System (Germany)* and *Monetary Transaction System (Great Britain)* in the different organizational units.

The next step should be an analysis of these potential redundancies. First of all, they can turn out to be no redundancies at all, as it is possible e.g. in Figure 5, that *Monetary Transaction System (Germany)* supports other subprocesses of the process *Acquisition* than *Monetary Transaction System (Great Britain)*. A software map that does not show the processes on the level on which this is visible, but only a more aggregated view, cannot show this fact explicitly. Moreover, if there are redundancies, they might have been deliberately introduced, e.g. in order to achieve a higher flexibility. In such cases, it may be reasonable to retain the redundancy. In case that no such reasons can be found, the results from the analysis regarding redundancies can be used as input for activities defining visions or plans for the evolution of the application landscape. This can include definitions of project proposals that serve the elimination of the redundancies.

6 Resume and Outlook

In this article, we tried to address the challenges faced in the creation of an information model for *EA Management* and presented our approach to support the designing process with predefined information model patterns.

While *EA Management* frameworks like Zachmann [Zach87], TOGAF [TOG05], etc. offer guidelines for designing, planning, and implementing *EA Management*, they currently do not detail an *EA Management* information model. Enterprises at an initial state of *EA Management* have to implement these frameworks either using existing information models or creating their own information model beginning from scratch, which is a way not without pitfalls.

The approach presented in this paper tries to support enterprises introducing *EA Management* by providing a structured way for laying the basis for an organization-specific information model. Furthermore, the approach presented does not focus on the information itself, but on the concerns of stakeholders typically found in *EA Management*. These concerns are then addressed via

EA Management patterns. Having provided an initial example, how such a pattern could look like, we are currently describing more patterns, as we have encountered them in our project *software cartography* as part of the *state of the art*, e.g. at industry partners or in literature. Once having collected these patterns, we plan to conduct a series of interviews with people concerned with *EA Management* related tasks, in order to gain information about both relevance and usage context of the patterns identified. As a final step of this evaluation, we plan to consolidate these patterns in an *EA Management* pattern catalogue, which is intended to form a knowledge base for the construction of organization-specific information models supporting *EA Management*.

References

[Bern03] *Bernus, Peter:* Enterprise models for enterprise architecture and ISO 9000:2000. In: Annual Reviews in Control 27 (2003), pp. 211-220.

[BrWi05] *Braun, Christian; Winter, Robert:* A Comprehensive Enterprise Architecture Metamodel and Its Implementation Using a Metamodeling Platform. EMISA 2005, pp. 64 - 79.

[DiFH03] *Disterer, Georg; Fels, Friedrich; Hausotter, Andreas (Eds.):* Taschenbuch der Wirtschaftsinformatik. Carl Hanser Verlag, München, Wien 2003.

[DoD04] *Department of Defense:* DoD Architecture Framework Version 1.0, Volume I: Definitions and Guidelines. Department of Defense (DoD), USA 2004. http://www.defenselink.mil/nii/doc/DoDAF_v1_Volume_I.pdf.

[ELSW06] *Ernst, Alexander; Lankes, Josef; Schweda, Christian; Wittenburg, André:* Tool Support for Enterprise Architecture Management - Strengths and Weaknesses. In: The Tenth IEEE International EDOC Conference, Hong Kong 2006, pp. 13-22.

[Fran02] *Frank, Ulrich:* Multi-Perspective Enterprise Modeling (MEMO) - Conceptual Framework and Modeling Languages. In: Proceedings of the 35th Annual Hawaii International Conference on System Sciences 35 (2002), pp. 1258-1267.

[GHJV94] *Gamma, Erich; Helm, Richard; Johnson, Ralph; Vlissides, John:* Design Patterns - Elements of Reusable Object-Oriented Software. Addison-Wesley Longman, Reading 1994.

[IEEE00] *IEEE Computer Society*: IEEE Std 1471-2000: IEEE Recommended Practice for Architectural Description of Software-Intensive Systems. IEEE Computer Society 2000.

[Krcm05] *Krcmar, Helmut:* Informationsmanagement. 4th edition, Springer, Berlin, Heidelberg 2005.

[KrSH93] *Kronlöf, Klaus; Sheehan, Anne; Hallmann, Matthias:* The Concept of Method Integration. In: *Kronlöf, K. (Ed.):* Method Integration. John Wiley & Sons Ltd., West Sussex, England, 1993, pp. 1 - 18.

[LaWe04] *Langenberg, Kerstin; Wegmann, Alain:* Enterprise Architecture: What Aspects is Current Research Targeting? EPFL Technical Report IC/2004/77.

[LaMW05a] *Lankes, Josef; Matthes, Florian, Wittenburg André:* Softwarekartographie: Systematische Darstellung von Anwendungslandschaften. In: Wirtschaftsinformatik 2005, Bamberg 2005, pp. 1443-1462.

[LaMW05b] *Lankes, Josef; Matthes, Florian; Wittenburg, André:* Architekturbeschreibung von Anwendungslandschaften: Softwarekartographie und IEEE Std 1471-2000. In: Software-Engineering 2005, Essen 2005, pp. 43-54.

[Lank05] *Lankhorst, Marc:* Enterprise Architecture at Work: Modelling, Communication and Analysis. Springer, Berlin, Heidelberg, New York 2005.

[META02] *META Group:* Enterprise Architecture Desk Reference. META Group, Inc. 2002.

[OMG06] *OMG:* IT Portfolio Management Facility (ITPMF) Available Specification, dtc/06-05-01. Object Management Group 2006.

[PMCC01] *Purchase, Helen; McGill, Matthew; Colpoys, Linda; Carrington, David:* Graph drawing aesthetics and the comprehension of UML class diagrams: an empirical study. In: Australian symposium on information visualisation - Vol. 9., Australian Computer Society, Inc., Sydney 2001, pp. 129 - 137.

[Riih05] *Riihinen, Jaakko:* Enterprise Architecture Tool Survey TU Munich. Email correspondence between J. Riihinen (Director, Chief Enterprise Architect, Nokia) and R. Schlossar (BOC Information Systems), 2005-03-21.

[Sebis05] *sebis:* Enterprise Architecture Management Tool Survey 2005. Technische Universität München, Chair for Informatics 19 (sebis), Munich 2005.

[TOG05] *The Open Group:* TOGAF 'Enterprise Edition' Version 8.1. The Open Group 2005.

[Zach87] *Zachman, John*: A framework for information systems architecture. In: IBM Systems Journal 26 (1987) 3, pp. 276-292.