Enterprise Architecture Management Patterns

Alexander M. Ernst
Chair for Informatics 19
Technische Universität München
eMail: ernst@in.tum.de

October 3, 2008
CHAPTER 1

Hint to writer’s workshop participants

1.1 Remarks on this article

As a workshop participant you should concentrate and prepare the following EAM patterns:

- Standard Conformity Management (see page 11)
- Standard Conformity Exceptions (see page 32)
- Architectural Solution Conformance (see page 43)

The introductory part (see section 2), the additional EAM patterns and the closing section (see section 6) are given to introduce the overall context of EAM patterns and the EAM Pattern Catalog.

1.2 Next steps in EAM Pattern Catalog development

The EAM Pattern Catalog is currently available at http://www.systemcartography.info/eampc-wiki as a first version, based on the results of an extensive online survey. In order improve the current version and to further exploit the advantages of patterns an excerpt of the EAM Pattern Catalog is included in this document to be discussed in the pattern community.

The following changes are scheduled for this article:

- Problem statements will be adapted for a better focus on the underlying problem. In this context, context and forces will also be adapted.
- M-Pattern Standard Conformity Management will be split up in one M-Pattern and four sub-M-Patterns.
- The EAM patterns should continually be revised for readability and understandability, as well as they should be extended to give more detailed guidance in addressing the problems.
1. Hint to writer’s workshop participants

The following changes have already been done in the shepherding phase:

- Currently the EAM patterns follow a uniform structure, but is different than the ones usually used for documenting patterns. Therefore, a format for pattern description similar to Buschmann et al. [BMR+96] is used: Name, Also Known As, Short Description, Example, Context, Problem, Solution, Implementation, Variants, Known Uses, Consequences, See Also, and Credits. Versioning information and an identifier have been added to this template.

- The first version of EAM patterns were highly dependable on each other, especially the references between the different EAM pattern (M-, V-, and I-Patterns) types. In addition a problem section was only included in the M-Patterns. This has been changed to make EAM patterns more self contained.

- The first version of EAM patterns addressed more than one concern (problem) with one M-Pattern. Analyzing the concerns they have been split up in context, problem and forces.

- Example, context and problem sections have been revised to be more focused on the content of the EAM patterns.
This section gives an introduction to enterprise architecture management and an overview about the EAM Pattern Catalog.

2.1 Enterprise Architecture Management

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. Enterprise Architectures include everything that is needed to run a business, ranging from strategies (business, as well as IT strategies are of interest), over business processes, representing the value chain of the company, over business applications needed to support the business processes to infrastructure elements, like e.g. application servers or hardware. Documenting and managing the EA is an advanced topic, as the application landscape, which is part of the EA, includes a few hundreds up to a few thousand business applications and their interconnections in a mid-sized or large company. Thereby, managing the EA is a task, that has to be coped as the need for a flexible IT is an integral part of most companies. Another reason for the importance of EA management are regulations like e.g. Sarbanes Oxley Act (SOX), which dictate the management of the EA. Therefore, typical problems are likely to arise across companies. Whatever preliminary work exists in a company, there commonly is a demand for a more structured way to manage the evolution of the EA. A variety of approaches to introduce EA management has been proposed by academia and practice (see e.g. [BW05, Fra02, MBLS08]), but they all have to cope with at least one of the following problems:

- EA management is introduced from scratch, not considering related initiatives already present inside or outside the organization.

- EA management frameworks, like Zachman [Zac92], TOGAF [Gro03], etc., are usually either too abstract and therefore not implementable, or too extensive to be used in practice, as they have to be utilized as a whole.
• Lacking an actual starting point for an EA management initiative, companies tend to collect requirements from potential EA stakeholders in the organization. Consolidating their demands and integrating their information needs, an all-embracing EA management approach is likely to emerge, which would demand a vast amount of data to be gathered, although only a part of it would be needed to address the real pain points of the company.

• If an approach has been implemented, it is often not documented, why certain decisions have been taken, e.g. why a certain concept has been introduced in the information model. This leads to information models, which cannot be adapted or extended, due to the fact that no one knows what analyses rely on which parts of the information model.

• Approaches proposed e.g. by organizations or standardization groups are usually all or nothing approaches, meaning that they are supposed to be introduced as one single piece instead of incrementally. This results in an EA management approach that is not tailored to the company’s EA maturity.

In order to address the problems stated above, we propose to apply patterns, well known from other disciplines like architecture or software engineering. Using the pattern concept offers the possibility to profit from additional advantages. For example, according to [HA05], an advantage of the pattern concept is that it enables architects to understand the impact of the architectural decisions at design time, because patterns contain information about consequences and context of the pattern usage.

Different definitions for pattern exist, see e.g. [AIS77], [BMR+96], or [GHRV94], but they all include a common ground. Patterns

- are a general, reusable solution to a common problem and
- are dependent on their context.

The above described properties are the basis for the EAM pattern approach, which initially has been introduced in [BEL+07b]. EAM patterns describe solutions, based on observed practices for recurring problems in EA management that can and may have to be adapted to a specific enterprise context.

### 2.2 EAM Pattern Catalog

The EAM Pattern Catalog is a collection of related EAM Patterns and has been published as a technical report, which can be downloaded at http://www.systemcartography.info/eampc-wiki. In [BEL+07b] we have introduced the following three kinds of EAM patterns. A **Methodology Pattern** (M-Pattern) defines steps to be taken in order to address a given problem. Furthermore, as a guidance for applying the method, statements about the intended usage context and the problem which is addressed are provided, together with possible consequences and known usages. The procedures defined by the M-Pattern can be very different, ranging from e.g. visualizations and group discussions to more formal techniques as e.g. metrics calculations [LS08]. M-Patterns have been introduced, because missing methodologies constitute a common issue in current EA management approaches. Frameworks as e.g. TOGAF [Gro03] provide a process model (TOGAF ADM), but leave the details of
the methodologies supporting the specific activities in the EA management process relatively open. 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.

A Viewpoint Pattern (V-Pattern) provides a language used by one or more M-Patterns and thus proposes a way to present data stored according to one or more I-Patterns. In our research project Software Cartography (see e.g. [BEL+07a, BEL+07b, LMW05, Wit07]), we found that industrial users often specify viewpoints by example. This means that an exemplary view is provided for the viewpoint, possibly together with some textual explanations. While we do not contend that this may be sufficient in certain use cases, e.g. sketching concepts in presentations, we see problems arising, when the goal is providing official information to a wider audience for an extended period. In order to ensure the understandability of a diagram, we regard e.g. a legend to be mandatory.

An Information Model Pattern (I-Pattern) supplies an underlying model for the data visualized in one or more V-Patterns. An I-Pattern contains an information model fragment including the definitions and descriptions of the used information objects. As described in [BEL+07b], different languages are possible for describing an I-Pattern, varying in their degree of formality, including among others textual descriptions in natural language, the Meta Object Facility (MOF), Unified Modeling Language (UML) class diagrams, ontology languages, and mathematical formalizations, or combinations of these approaches. Choosing a specific approach basically has to consider the needs of the use cases to be supported. While an object-oriented description might be sufficient for creating a visualization or a tabular report, e.g. process simulation may only be reasonably possible on a more formal basis.

Therefore, we propose using a language adequate to the problem to be addressed, thereby strongly considering UML as the default language, as it is widely understood and has been found by us to be problem-adequate in many practical settings in the context of EA management information models [BEL+07a]. In case a language different from UML is chosen, complementing its specification with an UML-based description can yield advantages, especially as integrating information model patterns is simplified by them being available in a common language.

As already mentioned before pattern should constitute a reusable solution to a common problem observed in practices. In order to fulfill this requirement an extensive survey has been conducted. In a first phase (October 2006 until July 2007), the EAM Pattern Catalog was initialized by our group based on input from the following sources:

- Research project Software Cartography, Technische Universität München, Chair for Informatics 19 (e.g. [BEL+07a, BEL+07b, LMW05, Wit07])
- Partners of the research project Software Cartography
- EAM Tool Survey 2005 [seb05] and EAM Tool Survey 2008 [MBLS08]
- Enterprise Architecture at Work (ArchiMate) [JGBvB05]
- Management von IT-Architekturen (Edition CIO) [Der06]
- IT-Unternehmensarchitektur, 2007 [Kel07]
In a second phase (July 2007 until February 2008), the initial *EAM Pattern Catalog* was evaluated by 30 companies using an extensive online questionnaire to identify methodologies and viewpoints that are considered relevant and useful by practitioners\(^1\). Based on the evaluation of the questionnaire results, the *EAM Pattern Catalog* in its present form covers\(^2\):

- 43 concerns (48 have been excluded due to the questionnaire evaluation),
- 20 methodologies (10 have been excluded due to the questionnaire evaluation),
- 53 viewpoints (21 have been excluded due to the questionnaire evaluation), and
- 47 information model fragments (19 have been excluded due to the questionnaire evaluation).

The conceptual UML class diagram in Figure 2.1 shows the dependencies between the three EAM pattern types, their relationship to problems and the inner structure of the utilized concepts. The class *Problem* in the conceptual diagram reflects the fact, that the EAM pattern approach is problem driven. Problems, also known as pain points, are usually the entry point for management activities in EA management.

![UML class diagram](image)

Figure 2.1: UML class diagram describing the relationships between Problems, M-Patterns, V-Patterns, and I-Patterns

The *EAM Pattern Catalog* version 1.0 [BELM08] also includes a graphical overview of the described EAM patterns and their relationships, the so called EAM pattern graph. This can be used to navigate between the different EAM patterns and to find related ones.

---

\(^1\) See [BELM08] for details of the selection process as well as relevance and usage statistics for each element.

\(^2\) During the creation of this article the EAM pattern approach has been revised, e.g. the concerns of version 1.0 of the *EAM Pattern Catalog* have been split up into problems and forces, etc. Therefore, concerns are not explicitly covered in the rest of this article.
2.3 Access to EAM Patterns

The following matrix shows a mapping of EAM patterns to typical EA management topics and EAM pattern types and may be used to find EAM patterns of interest. Different groupings are thinkable, e.g. based on existing Frameworks, like e.g. ITIL, or TOGAF [Gro03]. For easier access to the EAM patterns, their page numbers are included in brackets. As this document is an excerpt of the complete *EAM Pattern Catalog* only a few cells of the table are addressed.
<table>
<thead>
<tr>
<th>Technology Homogeneity</th>
<th>M-Pattern</th>
<th>V-Pattern</th>
<th>I-Pattern</th>
<th>Anti-Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard Conformity Management (11)</td>
<td></td>
<td>Architectural Solution and Technology Mapping (20), Organizational Unit Business Application Cluster Map (23), Business Application planning (26), Architectural Solution Definition (29), Standard Conformity Exceptions (32)</td>
<td>Technology Usage (36), Business Application and Organizational Unit Relationship (39), Architectural Solution Conformance (43)</td>
<td></td>
</tr>
<tr>
<td>Business Process</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Application Landscape Planning</td>
<td></td>
<td>Organizational Unit Business Application Cluster Map (23), Business Application planning (26)</td>
<td>Business Application and Organizational Unit Relationship (39)</td>
<td></td>
</tr>
<tr>
<td>Support of Business Processes</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Project Portfolio Management</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Infrastructure Management</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Interfaces, Business Object, and Services Management</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2.1: Mapping of EAM Pattern to EA management topics
Methodology Patterns (M-Patterns)

M-Patterns are grouped according to their membership to typical EA management topics, like the ones introduced in chapter 2.3. This excerpt of the EAM Pattern Catalog [BELM08] includes a selection of one M-Pattern, which is part of the question complex Technology Homogeneity.
3. Methodology Patterns (M-Patterns)

3.1 Standard Conformity Management

M-Pattern Overview

<table>
<thead>
<tr>
<th>Name</th>
<th>Standard Conformity Management</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>M-4</td>
</tr>
<tr>
<td>Alias</td>
<td>Management of Architectural Standards, Blueprint Conformity Management</td>
</tr>
<tr>
<td>Summary</td>
<td>The M-Pattern Standard Conformity Management pattern defines and manages architectural blueprints and corresponding architectural solutions. Analysis on this information may lead to new guidelines concerning architectural standards, as well as roadmaps to increase or decrease standard conformity.</td>
</tr>
<tr>
<td>Version</td>
<td>2.0</td>
</tr>
</tbody>
</table>

3.1.1 Example

The application landscape of SoCaStore has become more and more heterogeneous during its evolution, meaning that many different architectural standards ave been defined and are in use. Moreover, there are many business applications, which do not conform to the defined standards. The high number of different standards and technologies used, calls for a high number of experts able to operate and maintain the business applications conforming to them. Additionally, licensing costs and costs for integrating different technologies have to be considered.

3.1.2 Context

An enterprise with a large number of business applications (more than 50), which are part of the application landscape and infrastructure software needed to run the business applications.

3.1.3 Problem

You feel the risk of an unmanaged application landscape, with a multitude of technologies, will increase cost of development of new business applications, evolution and retirement of existing business applications. You do not know if the business applications follow a common blueprint or architecture and what the impact of a change would be. Typically such a situation appears in large organizations with decentralized IT departments, after mergers and acquisitions, or just because of system entropy increasing over time. You believe architectural standards will help to reduce risks and costs through more homogeneity. How do you establish and manage conformity to such architectural standards?

The following forces influence the solution:

- Do currently used business applications correspond to architectural blueprints and solutions (architectural standards)? Are deviation reasons documented, e.g. strategic decisions?
3. Methodology Patterns (M-Patterns)

- Which activities or projects have to be started in order to improve conformance to architectural standards? Which modifications to the currently used business applications are necessary to achieve conformity?

- Where are architectural blueprints or architectural standards used, and are there areas where those standards are breached?

- How is an architectural blueprint / architectural solution made up?

- How can licensing costs for business applications and infrastructure be reduced?

- How can risks concerning the utilization of incompatible technologies for business applications be reduced?

3.1.4 Solution

The M-Pattern Standard Conformity Management addresses the problems described above by setting architectural standards, i.e. developing a set of architectural blueprints and architectural solutions, and assigning them to new and existing business applications, in order to increase efficiency in IT operation and development.

Architectural standards can thereby be divided in architectural blueprints, which define, which abstract technologies, like e.g. a relational database system, may be used for new business application and in architectural solutions, which are like an instantiation of an architectural blueprint with concrete technologies, like e.g. an Oracle 9i database.

Architectural solutions and architectural blueprints consider homogeneity not only on the level of a specific kind of technology e.g. programming languages or middleware, but include architectural solutions and consider technologies at the level of standardized technology bundles.

After architectural standards have been set, activities and projects for improving conformance to the standards can be derived, which may then enter project portfolio management as proposals.

![Management process for Standard Conformity Management](image)

Figure 3.1: Management process for Standard Conformity Management

Subsequently, four aspects of the methodology are described: Firstly setting architectural standards is considered, which afterwards have to be analyzed concerning standard conformity for specific business applications or subsets of the application landscape. This is followed by
an enforcement of the defined standards, which at last have to be evaluated if they are still feasible in the company under consideration. To complete the process loop of Standard Conformity Management (see figure 3.1), architectural standards which are no longer feasible have to be adapted or new standards have to be created. Subsection implementation will additionally cover the aspect of involving the right people and establishing the right governance structures.

Setting Standards: Creating Architectural Blueprints and Architectural Solutions

Before setting specific architectural standards, it is necessary to decide, what these standards should encompass. Possibilities here are e.g.:

- Which components (deployed and running sub-systems) a business application may consist of, and how these may communicate (connectors).
- The infrastructure software, on which the components rely on.
- The hardware running the components.
- Development environments used for developing the respective software.

The EAMVS online survey [BELM08] found that the first two items are most important to practitioners.\(^1\) Thereby, the first and the second item can be addressed by architectural blueprints and solutions. Understood this way, an architectural blueprint is an exemplary description of a software architecture in the component-and-connector viewtype according to [CBB+02]. This leads to different possible notations for defining architectural blueprints:

- We propose V-Pattern Architectural Solution Definition (see page 29), which is based on the respective UML-notation in [CBB+02].
- V-Pattern Architectural Blueprint (see page 315 in [BELM08]) is a possible alternative to V-Pattern Architectural Solution Definition, but this pattern was evaluated by the online survey of the Enterprise Architecture Management Viewpoint Survey to be of minor importance.
- The architectural description language ACME [GMW97] is another possibility.

However, the description of the exemplary architecture in an architectural blueprint is technology-neutral. The specific technologies are set when an architectural solution is created based on a specific architectural blueprint, which assigns a specific technology to each so called abstract technology in the architectural solution. Several aspects may influence, which and how many architectural standards are offered. The following arguments are in favor for architectural standards:

- Projects may choose an architecture and technologies they see most fit for the respective tasks, without having to reinvent the wheel.

\(^1\) Ranked by practitioners regarding importance on a 1-5 scale (5 is most important), they received an average rating of 4 or more.
• Architectural standards document proven practices in combining technologies to fulfill certain tasks.

• Architectural standards may be used to reduce the heterogeneity of the application landscape.

• Knowledge about an additional architecture has to be kept available if the business application does not conform to defined standards, at least as long as it is operated.

• Knowledge about technologies is only needed for allowed technologies.

In contrast the following arguments are against architectural standard:

• It may be easier and faster to develop a business application exactly fitting its duties and responsibilities without following the defined standard architecture.

The set of offered standards has to strike a balance between these effects.

Analyzing standards: Analyzing standard conformity of business applications

The following steps describe how to create an overview of which business application uses which architectural solution, and give hints for analyzing this overview.

First of all the information about business applications and their architectural solution has to be collected. For collecting this information, it has to be noted that the employees operating a business application might not always be totally aware of the respective application’s architecture. Thus, the respective developers might have to be included into the data collection process. Of course, up-to-date architectural blueprint and solution definitions are a prerequisite for this task. Additionally, an understanding about the blueprints should exist with the developers, which can be facilitated by using V-Patterns like Architectural Solution Definition (see page 29).

The collected information should then be verified. Here also different possibilities apply, ranging from automated plausibility checks to manual reviews, which could be tied to visualization creation. If necessary, missing or possibly erroneous information has to be delivered in addition or corrected.

An Architectural Solution and Technology Mapping (see page 20) - diagram can provide first background information about the existing architectural blueprints and solutions. It can give a first overview of the technologies included in a standard. This allows a first stage of the analysis: The set of standards might be too small (too restrictive) or too big (too permissive). As a next step the application landscape has to be analyzed in terms of business applications, which not belong to an architectural standard. This can e.g. be done by highlighting such business applications. V-Patterns, which may be used for this purpose are Standard Conformity Exceptions (see page 32) and Architectural Standard Clustering (see page 101 in [BELM08]). V-Pattern Standard Conformity Exceptions can indicate, where architectural standards are met, where this is not the case, and where breaching the standard is specifically allowed.
Utilizing these two V-Patterns, the focus is likely to be on the business applications not conforming to the respective architectural standard. On the one hand, such business applications might be looked at specifically, considering e.g.:

- Does it require not to conform with the standard?
- How much are costs thus induced? Who bears these costs?
- Has the wrong standard been prescribed for the application?

On the other hand, analyses can also focus on the totality of the non-conforming business applications, e.g. looking at:

- What do they have in common?
- Are the standards inadequate for important parts of the application landscape?
- Are there organizational units for which there are no means of enforcing the standards?

Especially an Architectural Standard Clustering-diagram (see page 101 in [BELM08]) might be helpful in getting an impression of the importance of the different architectural solutions. A standard only existing to serve a small proportion of the business applications might need a special justification.

Breaching standards can e.g. be allowed if significant business success is tied to the possibility to have projects outside the respective standards. However, this introduces the issue of who receives the benefits derived from breaching the standard, and who bears the costs induced thereby. This topic is further discussed in the Consequence Section.

**Enforcing Standards: Deriving Measures for increasing Homogeneity**

Once architectural standards are set, measures for improving conformance have to be developed and discussed. Certainly, such measures are described in a detailed, textual way. However, diagrams like V-Pattern Business Application planning (see page 26) can give an overview of the changes in the application landscape due to a (specific) proposed measure. Deriving measures involves finding the non-conforming business applications e.g. via analyzes as described above. Based on this, the reasons for the business applications non-conforming to the standards can be determined. This sets the ground for deciding, whether a specific business application currently not conforming to its standards has to be changed. Subsequent points might be important in such a discussion:

- Has the wrong standard been set for a business application? In this case, the standard should be changed.
- If there is excessive cost for standard conformance, an exception could be sensible.
- If the benefit of conforming to the standard cannot be realized in a specific situation, this might also be a reason for an exception.

If it is decided that one or more business applications have to be changed, the respective proposal has to be created, and can then be entered into project portfolio management, or an equivalent management process.
3. Methodology Patterns (M-Patterns)

Evaluate Standards: Find standards which have to be adapted

Setting standards, analyzing and enforcing them is not sufficient for a continuous management approach. As requirements and technologies change over time the standards, which are currently in use have to be evaluated on their applicability in the future.

There are different ways on how to achieve such an evaluation. A simple approach would be to count how often a certain standard is in use. If this value is below a certain threshold, an in-depth analysis should be started why the standard is only seldomly used. One reason for this could be that the standard has been created for specific requirements. In this case the standard need not be revised. Another reason could be that the standard uses a technology which is no longer considered to be state of the art. In this case the standard should be adapted in the next step of the management process.

More sophisticated approaches, like technology roadmaps defining the upgrade paths for technologies that may be used in standard definitions could also be used but require higher efforts to be realized.

3.1.5 Implementation

In order to implement this M-Pattern within an organization it is very important to create the required governance structures and to involve the right people, meaning that it is required to establish a group of people, which are able to define the required architectural standards. This group of people is called the Architectural Standard Group and usually recruits its members from the software architect and from the enterprise architect group of the company, as knowledge about technologies and their interrelations is required. See Architect Also Implements in [CH04] for detailed information about this topic.

Only defining the standards usually is not enough as it is required that these standards are controlled and if necessary are enforced. This should be done by a special group of enterprise architects, the Architectural Standard Control Group, which should be incorporated in every project exceeding a certain project cost limit. The limit is depending on the size of the company and the budget available for EA management.

A third group of people is required for, when escalation is required. This group, the Architectural Standard Board, should be on board level and should incorporate members of the business as well as of the IT part of the company. If no consensus between the project and the architectural standard group is possible, the architectural standard board has to decide if standards may be breached or not. The enforcement of this decision may also influence the budget of the project under consideration.

3.1.6 Known Uses

The approach documented in M-Pattern Standard Conformity Management is in use in the following companies:

- BMW Group
- HVB
- Enterprise Architecture Management Tool Survey 2008 / SoCaStore (sebis)
The approach documented in this M-Pattern can be used the following EA management tools

- ARIS (IDS Scheer AG)
- planningIT (alfabet AG)

### 3.1.7 Consequences

It is helpful, if not necessary for the M-Pattern, that architectural solutions are *boundary objects* between enterprise architects and software architects. These two domains need an aligned understanding of the architectural standards, enabling them to efficiently communicate in using them.

A boundary object is an object which allows members of different communities to build a shared understanding in respect to certain things. Boundary objects are interpreted differently by the different communities, and realizing and discussing these differences can lead to a shared understanding. [SG89, Str99]

If architectural standards are to be beneficial, there has to be an entity having both power and commitment to enforce the standards as described in the implementation section. This entity is then likely to be also in charge of allowing exceptions from the standards. Thereby, it has to address the problem that the benefit and the costs of conforming to blueprints and solutions occur in different places:

- It is likely that the costs for conforming to an architectural standard occur directly with the development team or operators responsible for the respective application (in the short term). Costs can also occur at users, if a conforming business application is less suitable, e.g. due to decreased performance, which is not improvable without a highly specialized architecture.

- The benefit of increased homogeneity are likely to be of a more long-term nature, and occur primarily with the IT departments responsible for operating and developing business applications. However, if more efficient development can lead to a more swift project execution, business might be able to benefit from a reduced time to market.

If the decision process is not able to balance this on a cross-organizational level, it might happen that decisions are locally optimal for specific organizational units, but suboptimal for the organization as a whole. An example for an approach trying to balance the aspects is allowing deviations from the standard, but estimating the future effort of fixing issues created by this, and imposing a respective fee on the organizational unit that demands breaching the standard.

Another consequence is that defined architectural standards have to be maintained and evolved to keep up with new technologies, developments, etc. On the one hand this has a positive effect as there is a need to continually rethink defined solutions resulting in a potential improvement of the defined standards. On the other hand investments are needed to be able to maintain and evolve the standards, which have to be in balanced with the potential savings.
3. Methodology Patterns (M-Patterns)

3.1.8 See Also

In order to support the implementation of M-Pattern Standard Conformity Management the following V-Patterns should be considered:

- *Architectural Standard Clustering* (see page 101 in [BELM08])
- *Architectural Solution and Technology Mapping* (see page 20)
- *Business Application planning* (see page 26)
- *Architectural Solution Definition* (see page 29)
- *Standard Conformity Exceptions* (see page 32)
- The architectural description language ACME [GMW97]

3.1.9 Credits
Viewpoint Patterns (V-Patterns)

This chapter contains a selection of V-Patterns, which are part of the *EAM Pattern Catalog* [BELM08].
4.1 Architectural Solution and Technology Mapping

<table>
<thead>
<tr>
<th>V-Pattern Overview</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>Id</td>
</tr>
<tr>
<td>Alias</td>
</tr>
<tr>
<td>Version</td>
</tr>
</tbody>
</table>

4.1.1 Example

The application landscape of SoCaStore has evolved over the years to fulfill business demands as quickly as possible. To achieve the required speed to support these demands new business applications have been developed without caring about selecting technologies for the new business application or defining architectural solutions. This resulted in a lack of information and knowledge about the technologies used in the company. In order to change this situation visualizations are needed to give an overview about the dependencies between the operated business applications and the technologies they are made up of.

4.1.2 Context

Getting and maintaining an overview about the technologies, which build up architectural solutions is difficult in large companies. How can you visualize this relationship in a concise manner?

4.1.3 Problem

You want to reduce costs and security risks by limiting the number of used technologies for business applications. To reduce, you first have to know what technologies are in use and where. How do you visualize such technology usage to get a quick overview?

The following forces influence the solution:

- You want to get easy visual feedback of impact analysis.
- You want to be able to spot popular technologies, because these are candidates for future architectural blueprints.
- With detailed version information you are able to see if migration between versions or evolution is feasible.
- You want to be able to get an overview about technologies that may be used in combination to prevent compatibility problems.

4.1.4 Solution

This V-Pattern consists of a table containing the technologies used in an architectural solution. Thereby, an "X" in a table cell symbolizes the usage relationship. It may be used in different
4. Viewpoint Patterns (V-Patterns)

ways. At first you may get an overview about the different technologies used within a company, together with the information, in which architectural solution the technologies are used. At second the V-Pattern may be used to perform impact analysis. Therefore, you select a technology, which will e.g. be changed or replaced, and you can than see the impact on the architectural solutions, which are in use within your company. As the number of different technologies and architectural solutions in use within a company may be high it may be useful to filter the information visualized, e.g. to select a technology and fade out all architectural solutions, which do not use the selected technology, in order to support the user in performing its impact analysis.

4.1.5 Implementation

This V-Pattern can be implemented in a spreadsheet tool, or if a graph representation (see variants section) is chosen in a graph layout tool. If filtering should be used, than this functionality should be supported by the tool.

4.1.6 Variants

Different variants for this V-Pattern exist. The information shown in Figure 4.1 could also be shown as a simple textual report, listing the technologies for an architectural solution. Another possible visualization would be as a simple graph, where technologies and architectural solutions are represented by nodes and the usage of a technology in an architectural solution is visualized by an edge connecting the appropriate nodes. The same kind of viewpoint can be created for architectural blueprints and abstract technologies. In this case the same alternatives, textual listing, graph visualization, etc. apply.

4.1.7 Known Uses

The following companies use this V-Pattern:

- HVB
Views according to this V-Pattern can be created, e.g. using the following EA management tools

- alphabet (planningIT AG)
- ARIS (IDS Scheer AG)
- System Architect (IBM)
- SoCaTool (sebis)

4.1.8 Consequences

The benefit of this V-Pattern is its simplicity. A simple table or graph is sufficient to address the problem described in the problem section. On the one hand these kinds of visualizations can easily be created on the other hand they can very intuitively be used to reduce the number of technologies, if variants or version differences can be eliminated. Additionally you can get a gist of the impact of such a technology elimination and it is easier to spot the impact of required technology changes, i.e. because of security issues.

Drawbacks of this V-Pattern are costs and time involved to collect the needed data in order to be able to create the visualization. Validity of the visualized information is another critical aspect of this V-Pattern. Technologies or business applications may already have changed, e.g. new versions have been introduced, before the information about them can beneficially be used.

4.1.9 See Also

This V-Pattern may be useful when using M-Pattern Standard Conformity Management (see page 11). The visualized information is based on I-Pattern Technology Usage (see page 36).

4.1.10 Credits
4.2 Organizational Unit Business Application Cluster Map

<table>
<thead>
<tr>
<th><strong>V-Pattern Overview</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td><strong>Id</strong></td>
</tr>
<tr>
<td><strong>Alias</strong></td>
</tr>
<tr>
<td><strong>Summary</strong></td>
</tr>
<tr>
<td><strong>Version</strong></td>
</tr>
</tbody>
</table>

4.2.1 Example

The application landscape of SoCaStore has continually grown since the foundation of SoCaStore. In the next few months a new subsidiary should be founded in Hong Kong, which demands for an appropriate IT support. Using already existing business applications is a solution, which is time-saving and cost-saving. In order to prepare them for their new tasks it is important to know where they are hosted, who uses them, who is responsible for them, etc. But this overview about the application landscape is not available through the continually growth of SoCaStore and now has to be regained.

4.2.2 Context

In an enterprise with a large number of business application it is hard to judge who is responsible for running them or who benefits or suffers from changes to them. How can you visualize this, so the relationship between organizational units and business applications is obvious.

4.2.3 Problem

Which relationships exist between business applications and organizational units? This kind of analysis is of importance, e.g. when trying to analyze and define responsibilities, utilizations, etc. for business applications.

The following forces influence the solution:

- Responsibilities for business applications have to be visualized to explicate them.
- Documenting usage of business applications is important.

4.2.4 Solution

This V-Pattern belongs to the software map type Cluster Map, which uses the concept of grouping (clustering) of elements in a visualization to express a relationship between them. The positioning of the different clusters is of minor importance as it doesn’t transport any information, but may be used to improve recognition like organizational unit headquarter is always positioned in the top left corner.
4. Viewpoint Patterns (V-Patterns)

Figure 4.2: Exemplary view for V-Pattern Organizational Unit Business Application Cluster Map

In this V-Pattern the cluster map concept is used to cluster business applications in organizational units. Figure 4.2 exemplarily visualizes a hosting relationship. This is only one possible semantic for this relationship, additional variants are described in the variants section of this V-Pattern.

A business application may appear multiple times within a view corresponding to this V-Pattern, e.g. if it is used by multiple organizational units.

Different kinds of usages are supported by this V-Pattern. First of all it is possible to give an overview about the as-is situation, or about planned and target scenarios of the application landscape, when incorporating the aspect of time. At second it is possible to do extended analyzes, like e.g. impact analysis, etc.

4.2.5 Implementation

Views according to this viewpoint can automatically be created by any drawing tool. As manual creation is time consuming and error prone it is advised to use a tool to generate the visualization.

4.2.6 Variants

Additional variants for this V-Pattern exist, as different semantics are possible for the relationship between business applications and organizational units. Three examples are listed below:

- Organizational unit *hosts* business application
- Organizational unit *uses* business application
- Organizational unit *is responsible for* business application
Each of these possibilities results in a variant of this V-Pattern. Thereby, the clustering of elements is used to represent the different relationships.

### 4.2.7 Known Uses

The following uses are known:

- *Enterprise Architecture Management Tool Survey 2008 / SoCaStore (sebis)*
- Klinikum der Universität München

Views according to this V-Pattern can automatically be created, e.g. using the following EA management tools:

- alphabet (planningIT AG)
- ARIS (IDS Scheer AG)
- Iteraplan (Iteratec)
- SoCaTool (sebis)

### 4.2.8 Consequences

A benefit of this V-Pattern is that it is a good starting point for EA management activities and offers many different analyzes. Views according to this V-Pattern can easily be explained and used, but contain a lot of valuable information about the current situation of the application landscape and may also be used for planning aspects. Additionally, the creation of views corresponding to this V-Pattern is simple and can even be done manually in some kind of drawing tool in the last resort. Furthermore, the amount of information that has to be collected is limited.

Another benefit is that this kind of visualization can easily be enriched by additional layers giving additional information, like costs for maintaining a business application, connections between business applications, etc. See the next section for further details.

### 4.2.9 See Also

The V-Pattern is based on information according to I-Pattern *Business Application and Organizational Unit Relationship* (see page 39) and its variants. Additionally, V-Pattern *Organizational Unit Business Application Cluster Map* is the basis for all V-Patterns, which rely on visualizing the relationship between business applications and organizational units together with other information. The following list gives an overview about these V-Patterns.

- *Business Application planning* (see page 26)
- *Standard Conformity Exceptions* (see page 32)

### 4.2.10 Credits

© TU München, sebis, 2008. All rights reserved.
4. Viewpoint Patterns (V-Patterns)

4.3 Business Application planning

<table>
<thead>
<tr>
<th>V-Pattern Overview</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>Id</td>
</tr>
<tr>
<td>Alias</td>
</tr>
<tr>
<td>Summary</td>
</tr>
<tr>
<td>Version</td>
</tr>
</tbody>
</table>

4.3.1 Example

SoCaStore wants to start to consolidate its application landscape. Therefore, existing business applications have to be modified or shut down and new ones have to be introduced. This is only possible if an overview about the application landscape and the planned changes is available.

4.3.2 Context

In a large application landscape the future development, e.g. the introduction of new business application has to be planned. How can the planned changes to the application landscape be visualized to get an overview?

4.3.3 Problem

You want to plan the evolution of the business applications, which make up the application landscape. To do this, you need an overview about which business applications have to be introduced, changed, shut down, or are not changed at all. Additionally the relationships between the business applications and the organizational units is of importance, e.g. to find the responsible person for an organizational unit with a lot of upcoming changes to discuss the consequences of these changes. How do you visualize the status of the business applications in order to get a quick overview?

The following forces influence the solution:

- You want to identify organizational units where a lot of changes take place and which are not at all affected.
- You want to find conflicts in the development plan of the application landscape.
- The effects of the future changes to the business application has to be explicated.

4.3.4 Solution

This V-Pattern uses the concept of a cluster map showing a relationship between business applications and organizational units, based on the V-Pattern Organizational Unit Business Application Cluster Map and its variants.
4.3.5 Implementation

The information about the type of change that has to be done on the business application should be visualized on a different layer than the relationship between organizational units and business application to be able to profit from the layering principle. When there is a demand to utilize the layering concept it is advised to use a tool supporting this functionality.

4.3.6 Variants

As already mentioned in the solution section different semantics for the relationship between business applications and organizational units exist. Each of them constitutes a different variant of this V-Pattern.

Additionally the information, which business applications are affected by changes can be visualized on a different software map type, like a Cartesian map, in particular a process support map. V-Pattern Process Support Map (see page 105 in [BELM08]) gives more information about this kind of software map type. The additional relationship to business processes offers the possibility for extended analyzes, like an analysis which business processes are primarily effected by the planned changes and which ones do not have to be considered.
4. Viewpoint Patterns (V-Patterns)

The variants mentioned above may also consider a time aspect, meaning that the visualization of the application landscape will look different if it e.g. shows the status for today and the status in a year from now.

4.3.7 Known Uses

The following uses are known:

- *Enterprise Architecture Management Tool Survey 2008 / SoCaStore (sebis)*

Views according to this V-Pattern can automatically be created, e.g. using the following EA management tools

- alphabet (planningIT AG)
- ARIS (IDS Scheer AG)
- SoCaTool (sebis)

4.3.8 Consequences

Normally, a business application can only be changed by a project, resulting in a need to also collect information about the project itself and not only about the scheduled changes for the business applications.

If a track back of the changes on a business application to a project is needed, it is advisable to have additional information about this project available, e.g. in a textual form, for further analyzes.

Considering time aspects demands for additional information, e.g. about the start time and duration of a project changing a business application. Therefore, another I-Pattern is needed to fulfill this additional demand.

4.3.9 See Also

Creating views based on this V-Pattern requires to collect information according to I-Pattern *Planned Project Effects* (see page 206 in [BELM08]) to visualize, which business applications have to be changed due to which projects. Additionally, information about the relationships between the business applications and the organizational units can be gained by I-Pattern *Business Application and Organizational Unit Relationship* (see page 39) or its alternatives.

4.3.10 Credits

© TU München, sebis, 2008. All rights reserved.
4.4 Architectural Solution Definition

<table>
<thead>
<tr>
<th>V-Pattern Overview</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>Id</td>
</tr>
<tr>
<td>Alias</td>
</tr>
<tr>
<td>Summary</td>
</tr>
<tr>
<td>Version</td>
</tr>
</tbody>
</table>

4.4.1 Example

Due to the uncontrolled evolution of SoCaStore’s business applications a multitude of different architectures are in use and are planned for future developments. This should be prevented in the future by providing defined and obligatory architectural standards. In order to define this architectural standards in a uniform way a uniform notation has to be used.

4.4.2 Context

Defining architectural standards and maintaining these is difficult if many different architectural standards and different notations are in use. What kind of visualization should be used to define and maintaining architectural standards?

4.4.3 Problem

You want to increase homogeneity of business applications’ architectures by using defined architectural standards. In order to achieve this goal you have to decide for an obligatory notation for defining and maintaining architectural standards. What notation should be used and how do visualizations created according to this notation look like?

The following forces influence the solution:

- You want to get an easy overview about defined architectural standards.
- You want to get an overview about the design of an architectural standard.
- The notation should be easy to understand and should offer the possibility to create visualizations, which are not misleading.
- The notation should be easy to understand.

4.4.4 Solution

V-Pattern Architectural Solution Definition uses the notation of an UML 2.0 object diagram to visualize the structure of an architectural solution. An architectural solution includes the allowed technologies, like e.g. Apache 2.0, Internet Explorer 6.0, etc. and the allowed connectors between these technologies, e.g. that an http connection may be used between the Internet Explorer 6.0 and the Apache 2.0. An example of a view defining an architectural solution is given in Figure reffigV-66.

© TU München, sebis, 2008. All rights reserved.
4. Viewpoint Patterns (V-Patterns)

4.4.5 Implementation

Views, which are based on this V-Pattern may be created using an UML modeling tool, in order to use the semantic checking incorporated in the tools. When using a variant not relying on the UML notation any kind of drawing tool may be used.

4.4.6 Variants

A variant of this V-Pattern is concerned about Architectural Blueprints. Thereby, the architectural solution is an concretion of an architectural blueprint, which defines which abstract technologies, e.g. a web client, a web server, etc. may be used and in which combination. Both variants can also be combined, meaning that information about technologies and about abstract technologies is shown within one visualization. See consequence section for more information.

A second variant would be to abdicate on the notation of UML 2.0 object diagram. On the one hand this has the advantage, that the views can be drawn with any visualization tool, on the other hand this leads to the problem that drawing without defined semantics may result in misleading views.

4.4.7 Known Uses

The following uses are known:

- BMW
Views according to this V-Pattern can automatically be created, e.g. using the following EA management tools

- ARIS (IDS Scheer AG)
- Rational Software Architect (IBM)
- System Architect (IBM)

### 4.4.8 Consequences

Visualizing information about an architectural solution and the associated blueprint in one visualization may lead to large and hard to understand views. Therefore, it may be reasonable to abstain from information about the architectural blueprint or the architectural solution. A benefit of this V-Pattern is that it is easily understandable by different groups, like software architects, enterprise architects, etc. within the company this improves communication between them. Besides the easy understandability of visualizations according to this V-Pattern they are extensive enough to avoid misleading interpretation and utilization.

### 4.4.9 See Also

Another V-Pattern, called *Architectural Blueprint* (see page 315 in [BELM08]) is also focused on defining architectural blueprints utilizing but also incorporates the concept of tiers to separate different layers of the architecture.

Creating views based on this V-Pattern requires to collect information according to I-Pattern *Technology and Connector Usage* (see page 223 in [BELM08]) to visualize, the relationships between technologies, connectors, and abstract technologies.

### 4.4.10 Credits

© TU München, sebis, 2008. All rights reserved.
4. Viewpoint Patterns (V-Patterns)

4.5 Standard Conformity Exceptions

<table>
<thead>
<tr>
<th>V-Pattern Overview</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>Id</td>
</tr>
<tr>
<td>Alias</td>
</tr>
<tr>
<td>Summary</td>
</tr>
<tr>
<td>Version</td>
</tr>
</tbody>
</table>

4.5.1 Example

SoCaStore is using the concept of architectural blueprints and architectural solutions for a few months now, but the effects of this concept, like standardization of the application landscape, etc., have not yet been analyzed. To conduct such analyzes visualizations are needed, which not only show the standard conformity of the application landscape, but also the allowed exception.

4.5.2 Context

Analyzing the standard conformity of business application is a difficult task if the application landscape excesses a certain size, usually this happens if more than 100 business applications have to be considered. It gets even worse, if exceptions to defined standards have to be considered. How can you visualize this in a summarily way?

4.5.3 Problem

You want to reduce costs by increasing the degree of standardization of the application landscape. To achieve this you first have to get an overview about the application landscape and its current status concerning the standardization. Before you can begin to adopt the business application not conforming to standards, you also have to consider that there also exist allowed exception. How do you visualize an overview about the standardization of the application landscape and also include information about allowed exceptions?

The following forces influence the solution:

- You want to get an overview about allowed exceptions to architectural standards.
- You want to identify organizational units where there is no information available about the standardization of business applications.
- You want to find organizational units with an exceptionally high amount of (not) standardize business applications.
4. Viewpoint Patterns (V-Patterns)

4.5.4 Solution

This V-Pattern uses the same concept, a cluster map as its base, as V-Pattern Organizational Unit Business Application Cluster Map (see page 23), resulting in the same variety of semantics that can be used in this V-Pattern. In this case a layer is added to the cluster map showing, which business applications conform to architectural standards, and where exceptions from these standards are tolerated. Figure 4.5 shows this on an exemplary cluster map, based on the hosting relationship between business applications and organizational units. Conformance to architectural standards is visualized by colors, exceptions to these standards are marked by a checkmark.

4.5.5 Implementation

The information about the type of change that has to be done on the business application should be visualized on a different layer than the relationship between organizational units and business application to be able to profit from the layering principle.

4.5.6 Variants

As already mentioned in the solution section different semantics for the relationship between business applications and organizational units exist. Each of them constitutes a different variant of this V-Pattern. See V-Pattern Organizational Unit Business Application Cluster Map (see page 23) for more information.

Additionally the information, which business applications are affected by changes can be visualized on a different software map type, like a Cartesian map, in particular a process map.
support map. V-Pattern Process Support Map (see page 105 in [BELM08]) additionally offers the possibility to analyze the standardization of business applications in respect to business processes.

In contrast to Figure 4.5 it would also be possible to visualize the information where exceptions to architectural standards on an addition layer. This offers the possibility to hide this information as long as it is not needed, leading to an easier to interpret view.

If the information about exceptions is not important for analyzes within a company this information can and should be omitted.

4.5.7 Known Uses

The following uses are known:

- Enterprise Architecture Management Tool Survey 2008 / SoCaStore (sebis)

Views according to this V-Pattern can automatically be created, e.g. using the following EA management tools

- planningIT (alfabet AG)
- SoCaTool (sebis)

4.5.8 Consequences

When documenting and visualizing the information that an exception to an architectural standard is tolerated it should also be documented why the exception is tolerated, e.g. in a separate document, in order to support additional analysis and next steps. This can be beneficiary for further analysis and next steps, but also includes the disadvantage that the required information has to be collected and has to be maintained.

If the information about allowed exception to architectural standards is not of importance it should not be visualized, resulting in a reduced amount of information that has to be collected to be able to create the visualization.

A benefit of this V-Pattern is that organizational units, or business processes in case a process support map is used, with a high number of business applications not conforming to architectural standards can easily be found and the additionally included information about the allowed exceptions makes it easy to find the business applications where you should start to increase the standardization.

4.5.9 See Also

Creating views based on this V-Pattern requires to collect information according to I-Pattern Architectural Solution Conformance (see page 43) to visualize, which business applications do, or do not conform to architectural standards, together with the information where exceptions are tolerated. Additionally, information about the relationships between the business applications and the organizational units can be gained by I-Pattern Business Application and Organizational Unit Relationship (see page 39) or its alternatives.

4.5.10 Credits

© TU München, sebis, 2008. All rights reserved.
CHAPTER 5

Information Model Patterns (I-Patterns)

This chapter contains a selection of I-Patterns, which are part of the *EAM Pattern Catalog* [BELMO8].
5. Information Model Patterns (I-Patterns)

5.1 Technology Usage

<table>
<thead>
<tr>
<th>I-Pattern Overview</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>Id</td>
</tr>
<tr>
<td>Alias</td>
</tr>
<tr>
<td>Summary</td>
</tr>
<tr>
<td>Version</td>
</tr>
</tbody>
</table>

5.1.1 Example

SoCaStore wants to start an initiative to reduce the number of technologies used within the company. As a first step the technologies, which are currently in use have to be collected. In order to store this information for future usage an information model has to be created as a basis for an implementation in a repository.

5.1.2 Context

Getting an overview about which technologies are used by which business application in a company is a good starting point for the homogenization of the application landscape. Additionally, this information may be used for documenting the current structure of business applications but also to plan future ones or to document proven practice. How can this information be kept available?

5.1.3 Problem

You want to reduce costs (licensing, maintenance, etc.), increase homogenization for the technologies used in a company or document proven practice, which combination of technologies work together well. What is a good way to store and maintain such kind of information? The following forces influence the solution:

- Information collected should be usable for impact analysis, e.g. if a technology has to be changed.
- You want to get an overview about the technologies, which are in use in a company.

5.1.4 Solution

The solution for the problem described above is based on two entities and one relationship, which are defined as follows:

- ArchitecturalSolution: A concrete stack of corresponding technologies, which are intended to be used together in realizing business applications, together with additional information on how to integrate these technologies into an complex architecture. Combining technologies together to an architectural solution among others indicates, that components created from the technologies are technically suited for interaction and integration.
5. Information Model Patterns (I-Patterns)

Technology Usage

- Technology: A Technology represents a technical constituent of a business application, ranging from an implementation framework or platform to a database management system or user interface toolkit. Exemplarily spoken, technologies may be "Apache 2.0.53" or "Oracle 9.2i".

- ArchitecturalSolutionUsesTechnology: The association uses indicates, which architectural solution uses which technologies.

5.1.5 Implementation

This I-Pattern may be implemented in a spreadsheet tool or in some kind of database system.

5.1.6 Variants

The information model fragment shown in Figure 5.1 may be extended e.g. by additional attributes, like licensing costs for technologies, or more advanced concepts like lifecycles for technologies, but as well as for architectural solutions.

5.1.7 Known Uses

The following uses of this I-Pattern are known:

- Enterprise Architecture Management Tool Survey 2008 / SoCaStore (sebis)

An equivalent information model fragment is included in the following EA management tools

- ARIS (IDS Scheer AG)
- planningIT (alfabet AG)
- SoCaTool (sebis)

5.1.8 Consequences

A liability of this I-Pattern is the amount of information that has to be collected to be able to perform reasonable analysis, as an architectural solution is typically built up by four or more technologies. If standard conformity analysis is not part of the selected EA management approach this I-Pattern should be omitted.

A benefit of this pattern is that it presents an easy way to document proven practice about which technologies can be used well together. This information may later be used when planning new business application or defining new architectural standards.

© TU München, sebis, 2008. All rights reserved.
5. Information Model Patterns (I-Patterns)

The data collection effort per year for information about the structure of architectural solutions and its components or technologies has been stated by practitioners using such an approach as:

![Bar chart showing data collection effort per year](chart.png)

5.1.9 See Also

V-Pattern Architectural Solution and Technology Mapping (see page 20) may be utilized to perform analysis on the information stored according to this I-Pattern.

5.1.10 Credits
5.2 Business Application and Organizational Unit Relationship

<table>
<thead>
<tr>
<th>I-Pattern Overview</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>Id</td>
</tr>
<tr>
<td>Alias</td>
</tr>
<tr>
<td>Summary</td>
</tr>
<tr>
<td>Version</td>
</tr>
</tbody>
</table>

5.2.1 Example

SoCaStore wants to start its EA management initiative with a minimal effort as the benefit of an EA management is unclear. As a first step a minimal information model should be created, which provides sufficient information to do an EA management show case, but for this purpose only limited information should be collected. An important aspect is, that the information collected should be reusable for the next steps in introducing an EA management.

5.2.2 Context

Getting an overview about which business applications are in use in a company, together with their relationships to organizational units is a good starting point for and a very important information in an EA management approach. How can this information be stored in a concise way?

5.2.3 Problem

You want to find a starting point for your EA management initiative, which covers a main topic in typical EA management approaches and which can be reused in the future development of the EA management approach. Starting with the relationships between business applications and organizational units has proven to be a good one in many companies. How can this information be collected and stored?

The following forces influence the solution:

- The selected information model should get by with a minimal effort for data collection to fill a repository where the information model has been implemented.
- Information collected according to the information model should be reusable with other I-Patterns.
- You want to be able to do the first show cases for an EA management approach.
- You want to find organizational units with(out) intensive relationships to business applications.
- You want to know which business applications are in use in a company.
5.2.4 Solution

Figure 5.2: Information mode fragment for I-Pattern Business Application and Organizational Unit Relationship

This I-Pattern consists of two entities BusinessApplication, OrganizationalUnit, and one relationship hosts and is exemplarily visualized in Figure 5.2. The hosts relationship shown in this figure is only one possible semantic for this relationship, additional variants are described in the variants section of this I-Pattern.

The entities and relationships can be defined as follows:

- OrganizationalUnit: An organizational unit represents a subdivision of the organization according to its internal structure. A possible example are the entities showing up in an organigram.

- BusinessApplication: A software system, which is part of an information system within an organization. An information system is therein according to [Krc05] understood as a socio-technological system composed of a software system (i.e. the business application), an infrastructure, and a social component, namely the employees working with the system. An information system is further described as contributing to the business process support demanded by the organization.

- OrganizationalUnitHostsBusinessApplication: The association hosts indicates, which organizational unit is hosting a business application.

5.2.5 Implementation

This I-Pattern may be implemented in a spreadsheet tool or in some kind of database system. When using different variants for this relationship within on information model for EA management, the semantics of the different relationships should be explicitly defined in order to avoid confusion.

Using only one relationship to implement different variants should be avoided as this leads to problems when analyzing information stored according to this information model.

5.2.6 Variants

Additional variants exist for this I-Pattern. The relationship between business applications and organizational units may thereby have different semantics, with three of them listed below:

- Organizational unit hosts business application
- Organizational unit uses business application

© TU München, sebis, 2008. All rights reserved.
• Organizational unit is responsible for business application

Each of these different semantics results in a variant of this I-Pattern. An example for the uses variant is shown in Figure 5.3

Figure 5.3: Information model fragment for variant of I-Pattern Business Application and Organizational Unit Relationship based on the uses variant

The following definitions specify the uses and the responsible for relationship.

• OrganizationalUnitUsesBusinessApplication: The association uses indicates, which organizational unit uses which business application.

• OrganizationalUnitResponsibleBusinessApplication: The association responsible for indicates, which organizational unit is responsible for which business application.

5.2.7 Known Uses

The following uses of this I-Pattern are known:

• Enterprise Architecture Management Tool Survey 2008 / SoCaStore (sebis)

• Klinikum der Universität München

An equivalent information model fragment is included in the following EA management tools

• ARIS (IDS Scheer AG)

• planningIT (alfabet AG)

• SoCaTool (sebis)

5.2.8 Consequences

This I-Pattern may sound trivial, but as the question where to start an EA management approach is not a trivial one, this pattern is of important.

A benefit of this I-Pattern is that it is a good starting point when building an EA management information model. The amount of information that has to be collected is limited, but it
5. Information Model Patterns (I-Patterns)

offers extensive possibilities to perform the first analyzes of the application landscape, which can then be used to show the benefits of the selected EA management approach. Additionally, information collected and stored according to this I-Pattern can easily be reused in an extended EA management approach.

If this I-Pattern should be integrated in an information model coping with information about BusinessApplicationVersions for a BusinessApplication, one should consider to change the BusinessApplication in this I-Pattern to the BusinessApplicationVersion.

5.2.9 See Also

This I-Pattern is the basis for V-Pattern Organizational Unit Business Application Cluster Map (see page 23) and its variants.

5.2.10 Credits

© TU München, sebis, 2008. All rights reserved.
5.3 Architectural Solution Conformance

<table>
<thead>
<tr>
<th>I-Pattern Overview</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>Id</td>
</tr>
<tr>
<td>Alias</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Version</td>
</tr>
</tbody>
</table>

5.3.1 Example

SoCaStore wants to start an initiative to analyze the status of the application landscape concerning the conformance of business applications to architectural solutions. Only to look for business applications not conforming to defined solutions seems not to be sufficient in this case, as there also exist allowed exceptions to this allegation. In this initiative an additional problems exists, because the required information could not be gather for all business applications.

5.3.2 Context

Managing information about which business application conforms to which architectural solution or why it does not conform to any, is difficult in a large application landscape. How do you collecting, storing and managing such information in a condensed and valid way?

5.3.3 Problem

You want to keep track about the status of the business applications concerning their solution conformity. How should an information model look like, to be able to collect and store the required information?

The following *forces* influence the solution:

- You want to be able to differentiate between business applications where no information about its conformance is available and business applications not conforming to architectural solutions.

- Allowed exceptions to architectural standards should be recognizable.

- Minimal effort should be needed to collect and manage information about solution conformance.

- You want to be able to identify business applications not conforming to architectural solutions.
The solution for the problem described above is based on three entities and three relationships, which are defined as follows:

- **ArchitecturalSolution**: A concrete stack of corresponding technologies, which are intended to be used together in realizing business applications, together with additional information on how to integrate these technologies into a complex architecture. Combining technologies together to an architectural solution among others indicates, that components created from the technologies are technically suited for interaction and integration.

- **BusinessApplication**: A software system, which is part of an information system within an organization. An information system is therein according to [Krc05] understood as a socio-technological system composed of a software system (i.e. the business application), an infrastructure, and a social component, namely the employees working with the system. An information system is further described as contributing to the business process support demanded by the organization.

- **NoArchitecturalSolution**: This entity represents the Non-Solution, i.e. it means, that an associated business application does not follow or does not need to follow any architectural solution.

- **BusinessApplicationConformsToArchitecturalSolution**: The association conformsTo indicates, in accordance to which architectural solution a business application is actually realized. Such a solution might be the singleton instance of the NoArchitecturalSolution, thereby indicating, that no standard solution has been used. Further, no such information might be present, described by the absence of an associated solution.

- **AllowedRelationship**: The association allowed explicates, which architectural solutions are per standard available for realizing the corresponding business application. Therein, the non-solution, as reflected by the singleton instance of the class NoArchitecturalSolution, is used to represent, that a business application does not need to conform to any architectural solution. This is especially necessary, to distinguish between the prescription of no solution vs. the absence of a prescription of that kind, i.e. missing data.
• BusinessApplicationType: The BusinessApplicationType is used to model, whether a business application has been developed as a piece of individual software or is a bought standard solution.

For determining information about the standard conformance of the business applications, such as displayed in Figure 4.5, the derived attributes \textit{standardConform} and \textit{exceptionAllowed} are used. The values of these attributes are derived by expressions similar to the following:\footnote{These expressions might be realized in the Object Constraint Language (OCL). For reasons of readability, we chose a mathematical notation instead.}

\begin{align*}
\text{standardConform} = \begin{cases} 
\text{null} & \text{for } (\text{realizedSolution} = \text{null}) \lor \\
& \quad (\text{allowedSolutions} = \text{null}) \\
\text{true} & \text{for } \text{realizedSolution} \in \text{allowedSolutions} \\
\text{false} & \text{for } \text{realizedSolution} \notin \text{allowedSolutions}
\end{cases}
\end{align*}

\begin{align*}
\text{exceptionAllowed} = \begin{cases} 
\text{null} & \text{for } \text{allowedSolutions} = \text{null} \\
\text{true} & \text{for } \text{NoArchitecturalSolution} \in \\
& \quad \text{allowedSolutions} \\
\text{false} & \text{for } \text{NoArchitecturalSolution} \notin \\
& \quad \text{allowedSolutions}
\end{cases}
\end{align*}

In deriving these values, the result \textit{null} is used to indicate, that based on the current information no valid statements on the respective property can be made.

\subsection*{5.3.5 Implementation}

This I-Pattern should be implemented in some kind of database system in order to guarantee consistency for the derived attributes \textit{standardConform} and \textit{exceptionAllowed}.

\subsection*{5.3.6 Variants}

A possible variant for this I-Pattern would be to simple add an attribute to every business application, indicating if the business application under consideration is conforming to defined architectural standards or not. It is not advised to use this simplified variant, as it restricts the possible analyzes. The advantage is that the amount of information, which needs to be collected is limited.
5. Information Model Patterns (I-Patterns)

5.3.7 Known Uses

The following uses of this I-Pattern are known:

- Enterprise Architecture Management Tool Survey 2008 / SoCaStore (sebis)

An equivalent information model fragment is included in the following EA management tools

- SoCaTool (sebis)

5.3.8 Consequences

A liability of this I-Pattern is the amount of data that has to be collected to be able to reasonable analyze the data. Especially the information of conformance to an architectural solution can only be answered by the business application owner. Therefore, every business application owner has to be interviewed, resulting in a certain investment.

A benefit of this V-Pattern is that an explicit distinction between ”there is no information about an architectural solution” and ”there is an exception from an architectural solution” is possible.

The data collection effort per year for information about the conformity of business applications to architectural solutions, reasons for non-conformity, etc. has been stated by practitioners using such an approach as:

![Bar Chart]

5.3.9 See Also

I-Pattern Architectural Solution Conformance is closely related to defining and documenting architectural solutions. This is addressed by I-Pattern Architectural Solution (I-66) (see page 223) in [BELM08] and M-Pattern Standard Conformity Management (see page 11). This I-Pattern can be used to manage information for V-Pattern Standard Conformity Exceptions (see page 32).

5.3.10 Credits

© TU München, sebis, 2008. All rights reserved.
CHAPTER 6

Using the *EAM Pattern Catalog*

The *EAM Pattern Catalog*, as a collection of observed practices in EA management, may support different activities with three of them detailed below. This chapter may be seen as an extension to the excerpt of the *EAM Pattern Catalog* contained in this document, in order to get a more detailed view on the complete *EAM Pattern Catalog* approach. For the complete *EAM Pattern Catalog* see [http://www.systemcartography.info/eampc-wiki](http://www.systemcartography.info/eampc-wiki).

6.1 Establishing an organization-specific EA Management through EAM Pattern Integration

The *EAM Pattern Catalog* supports introducing a light-weight, organization-specific approach to EA management based on best practices. In this use case it is assumed that EA management is introduced in a green field approach. In this case, first of all the *pain points* of the company, the so called problems have to be identified. This is supported by the list of problems included in the *EAM Pattern Catalog* [BELM08].

The selected problems include references to M-Pattern that can be used to address these problems. According to the approach sketched in Section 2.2, the methodology described in the M-Pattern uses certain V-Pattern for visualizing aspects of the EA, which are referenced by the M-Pattern. Based on the selected V-Pattern the associated I-Patterns can be selected. The last step is to integrate the EAM patterns to a organization-specific approach for EA management. Section 6.4 gives some hints on how to integrate the EAM pattern. This approach is the same as the generalized process on how to implement an EA management approach based on EAM patterns shown in Figure 6.1.

One or more catalogs of EAM patterns, supplied by *pattern authors*, serve as a basis. From these catalogs, the *developers* for EA management support choose EAM patterns, that are perceived as adequate for addressing specific problems of the respective organization, preferably under participation of the prospective *users*.

After integrating EAM patterns, thereby creating a coherent, light-weight, organization-specific conceptual model, the respective concepts can be implemented, e.g. in an EA management tool or a suite of tools.
This procedure offers the possibility to incrementally implement an EA management approach, starting with an initial set of M-Patterns, V-Patterns, and I-Patterns, which on the one hand includes rationale for the decisions made, e.g. why certain elements of the information model have been selected and on the other hand can later be extended, when a higher maturity level has been reached. In this case the EAM pattern graph (see [BELM08] page 23) can be used to e.g. identify EAM patterns, which easily fit into the already selected EAM patterns due to being closely related.

For example, it can be possible to create additional visualizations using the information already collected. In this case the I-Patterns, which are already in use have to be determined and then further V-Patterns have to be found, which use the same information model fragments.

The same is true for M-Patterns, as they use V-Patterns. Therefore, it may be possible that V-Patterns, which are already in use, can be utilized to address additional problems with M-Patterns.

6.2 Inspiring and Assessing an already implemented EA Management approach

The second usage scenario for the EAM Pattern Catalog is to take it as a reference book for suggestions concerning the approach currently selected in a company. This offers the possibility to compare the own EA management approach with best practices in use elsewhere. The EAM Pattern Catalog can e.g. be used to look for typical problems, which occur in other companies. This case may best be addressed by simply flipping through the EAM Pattern Catalog.

Additionally the EAM Pattern Catalog can suggest visualizations that can be found in academia and practice, which may be helpful in the currently selected EA management approach.

In this cases the EAM pattern graph (see [BELM08] page 23) can be used to find on the one hand M-Patterns to address the problems and on the other hand to find I-Patterns, showing the information needed to be able to create the required visualizations.
6.3 **EAM Pattern Catalog** as a basis for academic research

In addition to the application of the *EAM Pattern Catalog* in practice, it may also be seen as a basis for future academic research. Currently, there is no common ground for research on EA management, meaning that there is no approach for EA management, which may be iteratively enhanced and extended. There are punctiform approaches for specific EA management topics, but these lack the integration into a holistic EA management approach, and the acceptance in wider communities.

The pattern based approach addresses this deficiency as it offers the possibility to improve single EAM patterns without having to create a completely new approach. Furthermore, the existing *EAM Pattern Catalog* can easily be extended due to the openness of the pattern based approach.

Therefore, we are currently establishing a community, which will govern the future development, by e.g. performing reviews, improvement, extension, etc., of the *EAM Pattern Catalog*.

6.4 Integrating EAM Patterns

Integrating the selected EAM patterns is an important aspect of using the *EAM Pattern Catalog*. Special attention has to be paid to potential conflicts, inconsistencies, or discrepancies, due to contradictory assumptions made by different EAM patterns, especially when using EAM patterns of different origins\(^1\).

Such diverging assumptions may be completely valid for the EAM patterns themselves, e.g. due to them being based on different theories, being designed for different environments, or addressing different problems. However, when simultaneously contained in a specific approach to EA management, diverging assumptions may easily turn out to be damaging or depriving results coming from the approach of their validity. This motivates the need to carefully manage such discrepancies in integrating EAM patterns, preferably avoiding them altogether.

For more information on how to integrate EAM patterns see [BELM08].

---

\(^1\)Integrating the EAM patterns shipped with the *EAM Pattern Catalog* [BELM08] should be a minor problem, as the EAM patterns are developed relying on a common terminology in order to fit to each other.


© TU München, sebis, 2008. All rights reserved.