Institut für Informatik
der Technischen Universität München

Metrics for Application Landscapes

Status Quo, Development, and a Case Study

Josef K. Lankes

Vollständiger Abdruck der von der Fakultät für Informatik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitzender:	Univ.-Prof. Alfons Kemper, Ph.D.
Prüfer der Dissertation:	
	1. Univ.-Prof. Dr. Florian Matthes
	2. Univ.-Prof. Dr. Martin Bichler

Die Dissertation wurde am 24.06.2008 bei der Technischen Universität München eingereicht und durch die Fakultät für Informatik am 18.10.2008 angenommen.

Zusammenfassung

Die Bedeutung und Komplexität der IT in Unternehmen sind in der Vergangenheit gestiegen und nehmen weiter zu. Daher stellt das Management der Anwendungslandschaft aktuell eine wichtige Herausforderung dar. Die vorliegende Arbeit untersucht die Anwendbarkeit von Metriken im Management von Anwendungslandschaften. Durch die Metriken sollen Ziele und deren Erreichung auch für die Fachseite transparenter werden.

Die Arbeit liefert zunächst eine Umfeldanalyse im Bereich Anwendungslandschaftsmetriken, und konzentriert sich dann auf Metriken zur Analyse von Fehlerfortpflanzung, deren Anwendbarkeit in einer Fallstudie nachgewiesen wird.

Der erste Teil der Arbeit untersucht in einer empirischen Analyse, wie und in welchem Umfeld im Management von Anwendungslandschaften Metriken zum Einsatz kommen. Die Analyse basiert auf Experteninterviews und einer Onlineumfrage. Sie identifiziert Metriken für Anwendungslandschaften als ein relativ junges Gebiet, dem Praktiker allerdings ein deutliches Potential zuschreiben. Dabei richtet sich das Interesse der Praktiker verstärkt auf Anwendungsfälle, die sich mit den Qualitätsmerkmalen *Funktionalität*, *Flexibilität* und *Risiken im Betrieb* befassen. Aus Sicht der Praktiker erlauben Metriken vor allem, Ziele zu definieren, den Status Quo und Verbesserungspotential aufzuzeigen, und Kommunikation über die Anwendungslandschaft zu unterstützen. In diesem Zusammenhang helfen Metriken vor allem Beteiligten der Fachseite, über bestimmte Strukturen oder Änderungen der Anwendungslandschaft zu kommunizieren, ohne sich dabei zu intensiv mit technischen Details zu befassen. Die Umfeldanalyse liefert die Grundlage für Leitlinien zur Entwicklung von Anwendungslandschaftsmetriken.

Der zweite Teil der Arbeit entwickelt Metriken und metrikbasierte Methoden, die sich mit Betriebsrisiken befassen. Die erste Methode unterstützt die Analyse von Fehlerfortpflanzung in einer Anwendungslandschaft, die zweite hilft bei der Bewertung von Projektvorschlägen, die Fehlerfortpflanzung eindämmen sollen. Beide Methoden visualisieren Metriken auf Softwarekarten.

Der dritte Teil der Arbeit beschreibt den Einsatz der Metriken in einer großen Bank. Diese Fallstudie zeigt die Anwendbarkeit der metrikbasierten Methoden in der Praxis. Dabei wurden zwei Vorschläge verglichen, die das Ziel hatten, in der Anwendungslandschaft stärker entkoppelte Substrukturen einzuführen. Die Fallstudie stellt dar, wie Metriken, Methoden und Darstellungen an eine bestimmte Einsatzsituation angepasst werden müssen. Darüber hinaus beschreibt sie den Aufbau von prototypischen Werkzeugen zur Berechnung und Darstellung der Metriken und liefert damit Erfahrungen zur Werkzeugunterstützung für Anwendungslandschaftsmetriken. Insgesamt führte der Einsatz der metrikbasierten Methoden zu einem verbesserten Verständnis der Projektvorschläge, womit die Fallstudie den Nutzen von Anwendungslandschaftsmetriken zeigen konnte.

Abstract

Managing an application landscape has emerged as a new challenge, since the importance, size and complexity of the landscapes is constantly growing in a major share of organizations. This work proposes metrics for application landscapes as a quantitative technique to address the challenge. Its aim is to make application landscape management more systematic and transparent, and render goals and their achievement more accessible to business.

This work starts with a broad, general outlook on the subject, followed by a focused treatment of metrics analyzing failure propagation, and concludes with their application in a case study.

The first part presents an empirical analysis of the environment, in which metrics for application landscapes are used. In this analysis, expert interviews and an online questionnaire identified application landscape metrics as a relatively young field, however perceived by practitioners as showing considerable potential. Use cases practitioners are especially interested in, include the quality attributes *functionality*, *flexibility*, and *operational risk*. Practitioners intend to use metrics in particular for setting goals, showing the status quo and potential for improvement, as well as communicating facts about an application landscape. In this context, business stakeholders value the ability to communicate the implications of specific structures or changes in an application landscape without delving too deeply into technical details. The analysis results form the basis of guidelines for application landscape metrics.

To give an example for application landscape metrics, the second part of the work introduces two methodologies concerned with operational risk: The first one supports analyses of failure propagation in an application landscape, while the second one addresses comparing proposals for limiting failure propagation. Both especially focus on visualizing the metrics they use.

In the third part, a case study applies the metrics based methodologies at a large bank, demonstrating their applicability to real world concerns. The case study compares two proposals created by stakeholders for limiting failure propagation. It demonstrates the adaptations necessary to employ the metrics based methodologies in a specific use case. As the case study discusses the prototypical software supporting the necessary calculations, it also contributes to the future development of EA management tools. Altogether, applying the metrics based methodologies provided the stakeholders with an improved insight into the effects connected to their proposals, thus demonstrating the benefit of application landscape metrics.

Acknowledgments

This work emerged from my activities as a research assistant at the Chair for Informatics 19 (sebis) at the Department of Informatics of the Technische Universität München. Here, I would like to thank all those who supported me in my research.

First of all, I would like to thank my doctoral advisor, Prof. Dr. Florian Matthes, for providing the opportunity to work on an interesting research topic, and for his suggestions and advice that greatly contributed to the success of this work. I would also like to thank Prof. Dr. Martin Bichler, for our conversations about the subject and for being the second referee of my thesis.

This work is part of and based on sebis' research into application landscapes and software cartography. Thus, I would like to thank my colleagues in the software cartography team for their patience in discussing ideas, developing approaches, writing publications, and reviewing texts. This cooperation greatly influenced my research on metrics for application landscapes. My thanks go to Sabine Buckl, Alexander M. Ernst, Christian M. Schweda, and Dr. André Wittenburg.

I would also like to thank the students which contributed to my research on application landscape metrics. Claudius Hauptmann provided me with insights into IT balanced scorecard topics in his project work. Stephen Lauschke's Master's thesis on layouting software maps enabled me to create the software maps used in the case study on application landscape metrics.

Discussing my research with practitioners and applying the metrics in a real world case study greatly contributed to my research, allowing me to keep my work focused, and evaluate its results. Thus, my thanks also go to the practitioners who gave me the opportunity to discuss my work with them, and apply it in practice.

Last but not least, I am most grateful to my family, especially to my parents, for their support and constant encouragement during my doctoral work, and to my wife, for her patience and understanding.

Garching b. München, June 2008

Josef K. Lankes

Contents

List of Figures

CHAPTER 1

Motivation and Outline

Contents

The application landscape, the entirety of the business applications used in an organization, has emerged as demanding high-level management attention from widely differing groups of stakeholders. It has entered the focus of interest ranging from strategy management, to which it may constitute an enabler or a bottleneck, to different IT functions, to which the application landscape is an entity to be adequately maintained and evolved.

These stakeholders involved in managing the application landscape need to be aware of, grasp, and communicate the relevant concepts. Thus, visualizations are one fundamental instrument to achieve this. However, visualizations alone reach their limits, when, e.g., concepts not easily visible but emerging from the application landscape as a whole, or its operation, are concerned. Metrics are an instrument to quantify such concepts, making them solid enough to be useful in sound management processes.

This work discusses the use of metrics in managing an application landscape. In addition to introducing the subject, Chapter 1 details the research questions to be addressed, points to the research methodologies used, and provides an outline of the text in Section 1.5.

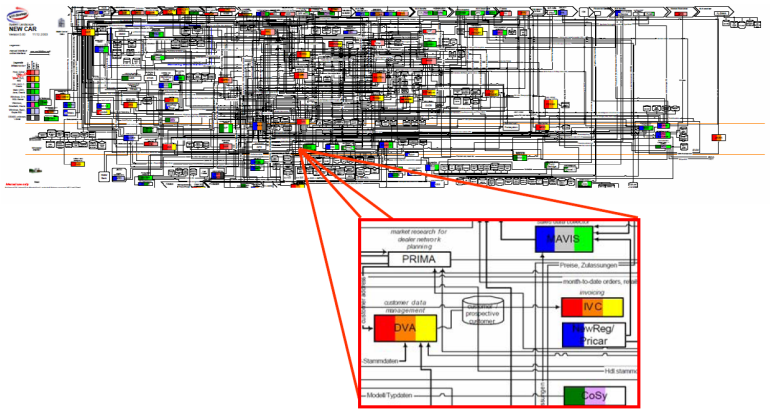


Figure 1.1.: Example of a software map from a real-life organization [Se05b]

1.1. Managing Application Landscapes as a Recent Challenge

Nowadays, organizations operate hundreds or even thousands of business applications in order to support their processes [Wi07]. These business applications, when taken together, form the application landscape, which is in turn considered part of the Enterprise Architecture (EA) [ME02] of the respective organization. Figure 1.1 shows a real-life application landscape, hinting at its complexity. The application landscape provides critical support to processes of the respective organization and is in turn evolved and operated by this organization, creating a complex web of interdependencies: Business applications are installed on hardware, are operated by organizational units, modified by projects, which should be in accordance to certain business objectives, just to name a few relevant concepts. Actually, information models proposing concepts to be documented about an application landscape or EA easily reach sizes of more than 100 entity types, not counting associations [Bu07a].

The above described growing complexity is met by the growing importance of the application landscape as an asset. Taking this standpoint, one may hypothesize that it is approaching a similar importance as, e.g., human resources or finance: No larger organization is likely to be able to exist without adequate IT [LLS06].

Most business processes need to be adequately supported by business applications, and several business concepts impossible without IT have emerged, e.g., e-business or the use of multiple sales channels in banking [St99].

In strategic management, the application landscape may take the role of an enabler or of a costly bottleneck. [Ro03] gives the example of UPS, where the package tracking capability present in the mid-1990s enabled the creation of new services, e.g., guaranteed delivery. However, such ”happy surprises” are, as also stated by [Ro03], rather rare. Many business strategies depend on the application landscape allowing the necessary adaptations in a reasonable way. [EGW07] names algorithmic trading as a recent trend in securities

trading. "Algorithmic Trading" identifies an approach, where a program automatically queries securities prices, and makes trades according to a predefined algorithm. However, it depends on the application landscape of the respective stock exchange, whether it can allow algorithmic trading and manage the resulting heavy load through constant queries by the trading algorithm.

Both the growing complexity and the growing importance lead to the conclusion that the application landscape has to be managed as an important long-term asset. Recent trends such as Enterprise Architecture (EA) management [LW04] or enterprise modeling [Fr02] substantiate this conclusion.

The structure of the application landscape with its interdependencies exerts a strong influence on critical success factors [AD05, BT00, Ro03], for example, the ability to incorporate new requirements into the application landscape depends on a structure that is friendly to modifications.

1.2. Enabling Management with Metrics

"You Can't Manage It If You Can't Measure It" is a proverbial allusion to the importance of metrics in management. Consequently, approaches based on measuring certain properties abound in business administration. They range from measures in finance and accounting ratios [RWJ02] to the multitude of metrics supporting strategy management in a balanced scorecard, which might very well extend into areas such as organizational learning and human resources [KN91].

However, metrics cannot yet be considered as widely used in managing application landscapes, especially in terms of metrics focusing on the structure of the application landscape. Metrics are, e.g., used in software engineering [KN91] and also for IT processes [IT05]. However, usage of and research into metrics for application landscapes might be considered less developed. In some cases, only the number of business applications is known, with the counting procedure and the underlying definitions only vaguely defined. Thus, management processes concerned with the application landscape might very well forfeit a much needed instrument. Metrics might be helpful in

Exploring the application landscape, in order to gain understanding of structures not easily visible (consider, e.g., transitive dependencies). Often, the focus thereby lies in finding potential for improvement, which might be discovered by looking for metrics values appearing unusual at first glance. Also, metrics can guide the eye to parts of the application landscape important in some respect, depending on the kind of metric used. These aspects grow in importance, if an analyst is not able to have an in-depth and broad overview of the object under consideration. While such an overview might be possible in smaller systems, making metrics a *nice to have*, it becomes more and more difficult in application landscapes, making metrics more and more a true enabler in handling complexity.

Deciding on activities affecting the application landscape and especially its structure. Such decisions should not be totally based on ad-hoc arbitrations made in a state of insufficient information about potential effects and side effects. This sets the stage for explicit evaluation techniques, e.g., metrics, that are able to evaluate the implications of changes. One might make assessments and discuss them ad-hoc in smaller systems, but as the system under consideration approaches the size and complexity of an application landscape, this might very well lead to seemingly endless discussions re-creating again and again un-documented models of how certain effects emerge. Separating the creation of such models, with the respective metrics, from using them is likely to pay off with increasing complexity of the subject under consideration. The multitude of stakeholders commonly involved in managing an application landscape is likely to add to this effect.

Communicating aspects of the application landscape or proposals concerned with the application landscape to other stakeholders, especially from non-technical domains. A stakeholder from the business side might be concerned with the effects specific structures in the application landscape have on characteristics directly relevant to his business cases. It might be relevant to him that a project improves changeability in certain business applications, enabling him to get future changes done more swiftly. However, he might not be interested in how this is achieved technically, e.g., by introducing specific interfaces. In fact, he may not even possess the technical knowledge necessary to understand such considerations. Therefore, he might be more interested in metrics tied to the qualities relevant to his business case than in technical details.

Thus, metrics, as an approach to quantify properties of objects [Kr71], can be viewed as a powerful aid for management activities in coping with an application landscape's complexity. Together with the above described lack of methodologies using application landscape metrics, this presents such metrics as a promising research area. The subsequent section introduces the research questions to be addressed in this work.

1.3. Research Questions

According to [GL04], research questions are questions of which the answers add knowledge to an existing knowledge base. As the research questions guide the research process and form the background against which the research results are to be evaluated, they are now explicitly developed and described:

1. **Which management situations can metrics support?**
    - **To what extent do practitioners see metrics for application landscapes as realistic and useful?**
    - **What are the constraints the management situations impose on metrics-based methodologies?**
    - **What quality attributes are relevant?**

The question is thus concerned with exploring the *environment* in which the metrics are intended to be used. **[Section 3.1 presents results from an environment analysis.]**

2. **How can metrics and metrics-based methodologies for addressing concerns in the management of application landscapes be constructed?** This question points towards prerequisites, guidelines, and caveats to be considered in the design process of such artifacts. **[Section 3.2 proposes guidelines for constructing and using application landscape metrics.]**
3. **How specifically can information about an application landscape be supplied to management processes by metrics?** This aims at the construction of metrics providing information about specific concerns found to be relevant to managing application landscapes. This includes proposing models explaining how the aspects of the application landscape measured by the metrics affect the quality attributes directly relevant to the stakeholders. **[Chapter 4 constructs metrics for addressing availability and failure coupling aspects in an application landscape structure.]**
4. **How can metrics improve the management of application landscapes?** Specifically, this question extends the previous one, by aiming at the design of actual methodologies, which describe how the metrics can be applied in addressing the respective concerns. This includes designing adequate visualizations of the metrics. **[Chapter 4 also proposes methodologies using the metrics for addressing availability and failure coupling aspects.]**
5. **How are metrics used in practice? Can metrics-based methodologies be a successful instrument in managing application landscapes?** In the context of this work, the question is primarily focused on the metrics and methodologies developed according to research question 3 and 4. However, a goal is to extend the findings to application landscape metrics in general. Specific aspects of these questions are as follows:

   - **What are the preconditions for application landscape metrics being applicable in an organization?**
   - **How can metrics for application landscapes be introduced?**
   - **Do the metrics (introduced according to the previous research questions) perform adequately?**

   [Chapter 5 discusses the use of the availability and failure coupling metrics in a case study, thus evaluating their applicability in practice.]

To guide the research process addressing these questions, the work selects a research methodology in the following section.

1.4. Methodological Issues

The research questions posed in Section 1.3 directly point to the design science approach. The questions outline a problem area in practice, and are addressed by research aimed at constructing and evaluating a solution. This is to a certain extent different from behavioral science, where research is aimed at constructing and examining theories, primarily concerned not about solving a problem but about finding *truth*[1].

While research concerned with *truth*, or *principled explanation of phenomena*, is served by paradigms as positivism, critical rationalism [St83], or interpretative research [Ei89, KM99], research focusing on utility, building and evaluating artifacts designed to meet certain (business) needs, is addressed by the *design science* paradigm.

According to [He04], design science approaches research as a problem-solving process, based on the principle "that knowledge and understanding of a design problem and its solution are acquired in the building and application of an artifact." In detail, these artifacts encompass the following [MS95, He04]:

- *Constructs*, providing a language to describe problems and solutions,
- *models*, describing reality according to specific constructs,
- *methods*, defining processes, which provide guidance on how to solve problems, and
- *instantiations*, showing that the constructs, models, or methods can be implemented in a working system.

Answering its research questions, this work creates constructs for modeling the application landscapes to be evaluated. It designs methods for evaluating the application landscape, which contain the metrics (described formally), and guidelines for their application (described textually). Lastly, it develops an *instantiation*, namely a prototypical tool support for calculating and visualizing the metrics.

Figure 1.2 presents a conceptual framework, which shows fields to be considered in information systems research, thereby specifically considering aspects relevant to the design science paradigm.

The artifacts to be designed in this work are, as mentioned above, metrics-based methodologies, including the respective modeling constructs, metrics definitions, visualizations and application guidelines, and the prototypical tool environment. Thus, research questions 3 and 4 are supposed to be addressed.

However, a dichotomy regarding *design* has to be taken into consideration: design describes both a product and the process leading to the product. Consequently, design science is also concerned with improving the design process. While the focus of this work is mainly on the artifacts themselves, it also considers the design process by providing guidelines for constructing application landscape metrics (see Section 3.2). This is meant to address research question 2.

[1] "The goal of behavioural-science research is truth. Theories posed in behavioural science are principled explanations of phenomena." [He04]

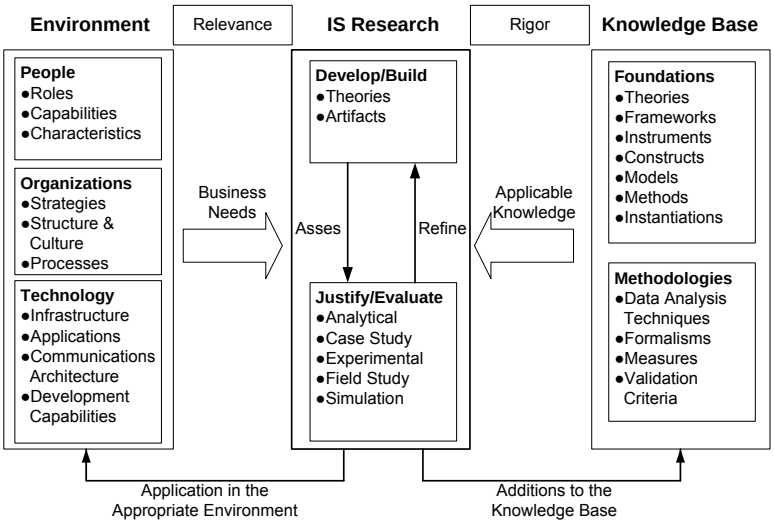


Figure 1.2.: Information systems research framework according to [He04]

Foundations are presented in Figure 1.2 as the knowledge base, which has to be adequately utilized for providing rigor to the research. Foundations of this research encompass the software cartography as developed by the Chair for Informatics 19 (sebis) of the TU München [LMW05a, Bu07a, Wi07] for developing the necessary modeling constructs, which clearly points to metamodeling using UML [OM05a, OM05b] and MOF [OM06a] as a foundation. The design of the metrics and metrics-based methodologies draws on measurement theory [Kr71] and the use of metrics in software engineering and business administration as reference disciplines. Furthermore, theoretic foundations in failure modeling (see e.g. [Ge89]) are used as a basis for specific metrics.

Chapter 3 additionally utilizes guided interviews and a questionnaire, together with the respective analysis methods and statistical techniques for giving a general overview of the environment, in which the metrics are designed to be applied. This addresses research question 1.

This overview can in turn be placed on the left side of Figure 1.2. Gathering information about this environment is firstly supposed to guide the development of the metrics designed in this work. It is supposed to enable it in achieving *relevance*, by providing the necessary information about "the goals, tasks, problems, and opportunities that define the business needs" [He04]. However, it is also meant to provide this information for use in further research about metrics for application landscapes.

Evaluation, shown as complementing development in the middle of Figure 1.2, is done mainly by a case study. Experimental validation is not used due to the difficulties in re-creating the complex environment in which managing an application landscape takes place in an experiment. Testing is to some extent done in the construction of the prototypical tool in the case study. Evaluation aspects are mainly discussed in Chapter 5.

Summarizing the above elaborations, the research presented here can be shown to conform with the guidelines for design science introduced by [He04]:

1. Design as an Artifact: This guideline is fulfilled, with the metrics-based methodologies being designed to address concerns in managing application landscapes. Also the prototypical tools constitute artifacts. However, the goal in their construction is not only showing that it is possible to calculate and visualize the metrics but also setting a foundation for evaluation (see guideline 3).

2. Problem Relevance: In order to ensure problem relevance, the metrics-based methodologies take into account an explicit exploration of the environment, i.e. the problem domain. Management of the application landscapes has been described as a recent challenge for organizations by, e.g., [LW04, Wi07].

3. Design Evaluation: For formally defined metrics, specific properties can be examined analytically. However, the main part of the evaluation here lies in the case study described in Chapter 5, which brings the metrics to use in a practical context, the application landscape of a large bank. The goal of the case study is to get feedback from the participating stakeholders, mainly with respect to possible usage scenarios and potential for improvement. This can be considered in future research and design efforts.

4. Research Contributions: The main contribution consists of the artifacts, i.e. the specific metrics and metrics-based methodologies presented in the work. In addition, the exploration of the problem domain and the guidelines for constructing metrics for application landscapes may be seen as a relevant contribution. In addition to that, especially when the metrics are statistically evaluated, they may get useful instruments in exploring dependencies between properties of an application landscape and certain quality attributes, thus making a contribution to the knowledge base.

5. Research Rigor: Building on the knowledge base as described above, the research tries to preserve rigor, especially via the use of metamodeling and the formal description of the metrics.

6. Design as a Search Process: In order to guide the development of metrics for application landscapes, the work provides a set of guidelines and draws conclusions from the case study, which can aid further research. The exploration of the problem domain can also be seen as providing information guiding the search process.

7. Communication of Research: To communicate the research results to technology-oriented audiences, this work describes the research in detail. To communicate the results to a management-oriented audience, presentations used in the context of the case study exist, which can be seen as a communication effort.

The research conducted according to this outline is subsequently presented as described in Section 1.5.

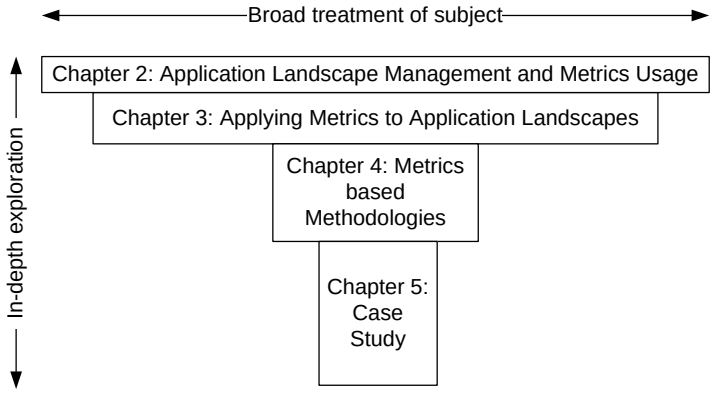


Figure 1.3.: Document structure

1.5. Document Structure

In addressing the research questions presented in Section 1.3 according to the methodologies outlined in Section 1.4, the work proceeds from a broad exploration of the subject to a focused, in-depth one, as shown in Figure 1.3.

Chapter 2 sets the foundations by defining the problem area, application landscapes and their management, and the instrument to be applied to this problem area, metrics.

Chapter 3 explores the status quo regarding the usage of metrics on application landscapes, including a related work analysis in Section 3.1.2.1, and derives guidelines for constructing metrics and metrics-based methodologies. The chapter explores how and to what extent practitioners wish to apply metrics in managing an application landscape, addressing research questions 1 and 2.

Chapter 4 takes a subset of the concerns practitioners want to address using metrics from Chapter 3, and develops metrics-based methodologies for those concerns. Thus, it designs methodologies for evaluating the influence of dependencies in an application landscape on service availabilities. In this context, Section 4.2 presents the knowledge base in failure modeling the approach is built on, and presents related work about failure propagation in software architectures. This addresses research questions 3 and 4.

Chapter 5, concerned with research question 5, evaluates the designed metrics-based methodologies by applying them in a case study in practice. This is more focused than Chapter 4, due to the specific concerns and data situation of the organization using the metrics. However, these aspects are explored quite in-depth, in a real-world practical setting.

Readers solely interested in the proposed metrics may directly jump to Chapter 4, with Chapter 5 providing additional insights into how to use the metrics in practice. Readers interested in data and information about the *environment* of and *market* for *metrics for application landscapes* may focus on Chapter 3.

CHAPTER 2

Application Landscape Management and Metrics Usage

Contents

In order to prepare the ground for describing research about applying metrics to managing application landscapes, this chapter describes both the problem area and the instrument to be applied to address respective problems. Thus, definitions used in this work are supplied, especially in the field of application landscapes and EA management, where universally accepted definitions of certain concepts are lacking. Additionally, the chapter provides pointers to literature basic to the subsequent work.

2.1. EA Management and Management of Application Landscapes

The problem area, application landscapes and its management, is here introduced by giving a basic definition of management, going on to sketch the concept of an application landscape, to finish with elaborating on managing an application landscape.

2.1.1. Management

Section 1.1 motivated the necessity to *manage the application landscape as an asset.* Management, as a process, is defined by [BP00] as "the process of assembling and using resources - human, financial, material, and information - in a goal directed manner to accomplish tasks in an organization."

The principal management functions are usually described as follows [BP00]:

Planning is defined as making decisions about the actions to be carried out in the future, based on expectations of future states. Plans are thus made for the staff. The extent of the staff, the granularity of the plans, and the timeframe covered by the plans varies with the kind of planning, ranging from *strategic planning* to *operative planning.*

Leading is concerned with influencing, motivating, and thus enabling others towards achieving specific goals.

Organizing involves adequate combination and utilization of resources. This ranges from the high-level organizational structure to specific teams. This function is meant to "bring order out of chaos".

Controlling, better called *Monitoring and Evaluating*, is about governing the work of the employees under the responsibility of a manager. Different approaches to monitoring and evaluating exist, which include *setting standards in advance*, *monitoring ongoing performance*, and *evaluating completed work.*

Consequently, management is both concerned with the present and the expected and desired future [Dr54].

Mintzberg [BP00] distinguishes three types of roles a manager has to fulfil. The *Interpersonal Roles* encompass ceremonial activities (e.g., welcoming high-ranking visitors), showing leadership, and fostering contacts outside the formal responsibility area. The *Information Roles* range from monitoring developments to disseminating information, also acting as a spokesperson, representing the respective business unit. Lastly, the *Decisional Roles* concern activities in decision making. These decisions can range from long-term, entrepreneurial decisions about new opportunities and business areas, via specific resource allocations, to disturbance handling and negotiations.

Management activities as described above appear at different hierarchy levels, with differing scopes. In the context of this work, the focus is specifically on the management of an *application landscape.*

2.1.2. The Application Landscape and Its Environment

[Bu07a, Wi07] define the application landscape as "the entirety of all business applications and their relationships in an organization". Business applications, and the kinds of entities they are highly interrelated with, are subsequently defined.

2.1.2.1. Business Applications

[Bu07a, Wi07] define a *business application* as software supporting a business process: "A business application is a software system which is part of an information system of an organization. An information system is according to [Kr05] 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, the infrastructure and the 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."

2.1.2.2. Processes and Users

Process is here defined according to [Kr05], as a sequence of logical individual functions with connections between them. It is assumed to have input and output factors, and a defined process objective [DFH03]. The process 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 organizational units responsible for executing the process use the supporting business applications, which is not limited to actual use by employees, but may include offering external business partners access to the applications.

2.1.2.3. Connections

In addition to the business applications, the connections between them are a main part of the application landscape. Connections enable business applications to mutually call functions, access, or exchange data.

A plethora of technologies and approaches for connecting business applications has evolved, distinguished by several aspects. One characteristic concerns the points in time and conditions, under which communication is possible, where, e.g., online and batch communication are distinguished. Regarding the mode of communication, remote method invocation, message passing, and use of shared data can be differentiated. The topology by which the connections are realized is another aspect. While direct connections are possible, integration infrastructures have turned out to be helpful in reducing the number of necessary interfaces. [Ke02, SS03]

However, not only technical aspects contribute to the difficulties of connecting business applications. Functional[1] aspects also have to be considered. If two business applications have to be connected, it must be ensured that the respective employees have a shared understanding of the necessary terms, that the functions executed or business objects handled by the respective applications are defined in a way that they fit together.

[1] In German: fachlich

Nonfunctional requirements are another field where connections between business applications exert an influence. It might, e.g., be hypothesized that a high number of connections to other applications complicates customizing or developing the respective business application.

However, adequately connecting business applications is of paramount importance in deriving benefit from IT. Business applications, which separately address specific domains on their own, without any connections, are generally considered disadvantageous. They lead to high efforts for data exchange and may also run the risk of using inconsistent data, with respective influence on process quality.

2.1.2.4. Running and Developing Business Applications

Business applications are hosted by organizational units, which are responsible for running the respective business applications. These are the technical contact persons for the applications. They are concerned with the infrastructure, on which the application landscape is built, e.g., middleware, databases, operating systems, and of course hardware.

Possibly different organizational units, or even external contractors, may be responsible for developing and maintaining, or in the case of standard software, customizing business applications. Such organizational units are concerned with facts as, e.g., programming languages the business applications are developed in, allowed or prescribed software architectures, or projects modifying business applications and their goals. While project execution is often outsourced, defining project proposals and deciding on them is much more likely to remain in-house, as this determines the direction in which the application landscape evolves, thus laying a base for future capabilities and even strategies [RWR06].

2.1.2.5. Strategy Management and the Application Landscape

Strategies, as alluded to in Section 1.1, affect and are affected by IT in general, and the application landscape in particular. This is discussed by a host of literature, of which two examples are mentioned here briefly.

[HV99], for example, discuss, how business strategy and IT strategy have to be aligned. They describe four domains and how they should be aligned to each other. The domains are distinguished along the dimensions *business vs. IT* and *internal vs. external*. The domain *I/S infrastructure and processes* (internal aspects of IT strategy), encompasses, besides IT processes and IT skills, decisions areas related to the *portfolio of applications*. This can be seen as synonymic to the *application landscape*.

Ross, Weill, and Robertson put a *foundation for business execution* at the core of their approach to EA management. This *foundation for business execution* is defined

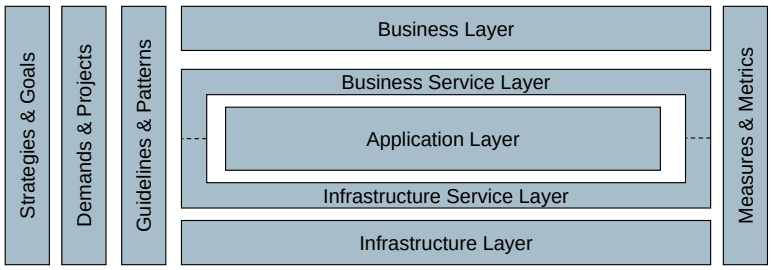


Figure 2.1.: Kinds of entities in an application landscape: layers and cross functions [Wi07]

as "the IT infrastructure and digitized processes automating a company's core capabilities" [RWR06]. It is described as the foundation on which strategic initiatives have to build, and is a key factor to their success[2].

2.1.2.6. Summary

Figure 2.1 summarizes the kinds of entities likely to be relevant in the context of application landscapes [Wi07].

The *business layer* reflects entities connected to the business itself, e.g., the abovementioned processes, products, or organizational units. The *application layer* is mainly concerned with business applications, their connections, and related elements. Services provided by the business applications to the business are of concern in the *business service layer*, which can include the respective service level agreements (SLAs). The *infrastructure layer* is about entities relevant in providing the technical infrastructure, on which the business applications rely. In addition, this layer is encapsulated by an *infrastructure service layer*, responsible for the infrastructure services on which the business applications rely, also possibly including SLAs.

The cross functions can concern entities from all layers. *Strategies and goals* derived from them should guide the needs for action on the application landscape. Thus, strategies and goals can generate *demands and projects*, which leads to the respective cross function in Figure 2.1. Standardization and homogeneity issues demand that projects consider certain *guidelines and patterns* when working on the application landscape, which might, e.g., include architectural guidelines or best practices in development. The *measures and metrics* are supposed to aid planning and controlling the application landscape, pointing to the subject of this work.

[2]Section 2.1.3.1.3 introduces the approach to EA management described by Ross, Weill, and Robertson more in detail.

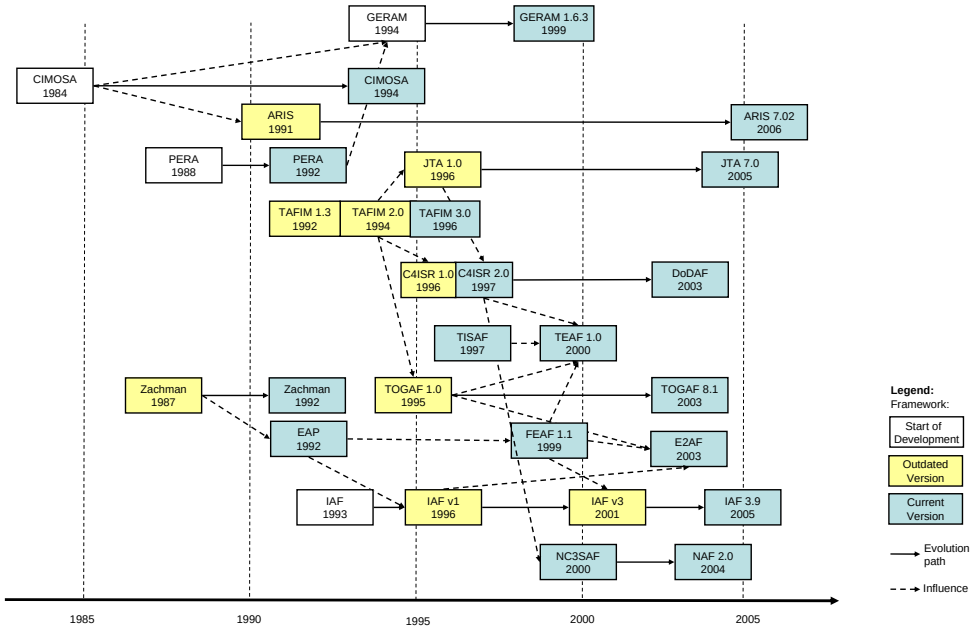


Figure 2.2.: Overview of EA management frameworks [Le07]

2.1.3. Managing an Application Landscape and EA Management

The importance and complexity of the application landscape make it a target for management. Management activities have to ensure that the organization makes optimal use of its IT, in this case, especially of the application landscape.

Different approaches have been developed to aid the respective management activities. To begin with, several standardization organizations have developed so-called EA management frameworks. [Le07] presents an overview of the evolution history of a major share of these frameworks, here shown in Figure 2.2.

Thus, the term *EA management framework* includes rather diverse concepts, as described by [Le07]. They vary regarding the areas they make suggestions about (IT management processes, deliverables, ...), or the domains they are focused at (private enterprises, government, defense, ...). Below, a short outline of TOGAF is presented as an example for an *EA management framework*.

Not less diverse are approaches developed by academia or single practitioners, e.g., by Dern [De06], Frank [Fr02], Keller [Ke06], Lankhorst [La05], Ross, Weill, and Robertson [RWR06], Winter [BW05], Wittenburg [Wi07], or the Abacus approach [Du05, Li06]. The approaches by Wittenburg and by Ross are subsequently sketched.

2.1.3.1. Selected Approaches

As mentioned above, three approaches to EA management are now presented, to highlight their diversity and their commonalities.

2.1.3.1.1. TOGAF

TOGAF [TOG02], *The Open Group Architecture Framework*, has been developed by *The Open Group* as a *set of supporting tools* for developing an EA. *Enterprise Architecture* is here defined as encompassing the following:

Business process architecture concerns business aspects, as processes, strategy, governance, and organizational structure.

Data architecture concerns organization of data, and the resources used for managing these data.

Application architecture concerns the structure of the business applications to be deployed, and how they relate to the core processes.

Technology architecture concerns the software and hardware infrastructure (hardware, networks, middleware), and the standards governing those elements and their operation.

Basically, this reflects the application landscape and its environment as described in Section 2.1.2.

At the core of TOGAF is the ADM, the *Architecture Development Method*, which proposes a high-level process cycle guiding the development of an EA.

Its processes, visualized in Figure 2.3 as circles, with single-headed arrows giving their logical succession, are described in a generic manner. This is because the ADM specifically describes that the processes have to be tailored to the specific use cases, resulting in an organization-specific ADM.

In this context, process objectives, the approach taken in the process, inputs, steps, and outputs form the main parts of each process description. The process cycle itself proceeds from business requirements to actual implementation. Summarized, the ADM processes shown in Figure 2.3 are as follows:

Prelim: Framework and Principles This is the first step in applying the ADM. The goals of this process are related to preparing the organization for executing the ADM, i.e. to perform architecture work. The outputs include an organization-specific adaptation of the TOGAF framework, basic architecture principles, and the business principles, goals, and drivers.

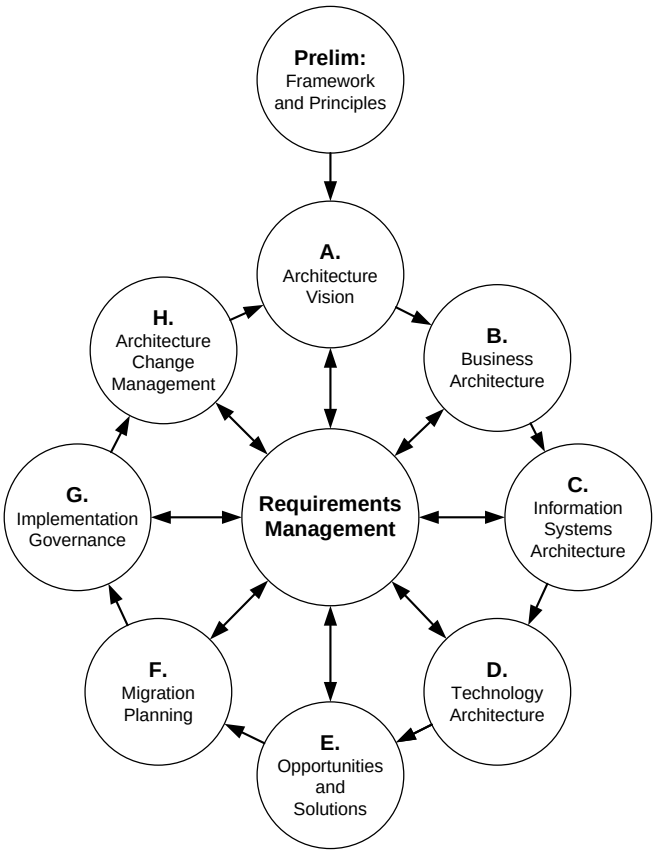


Figure 2.3.: Architecture development method in TOGAF [TOG02]

A: Architecture Vision Denotes here several kinds of requirements for starting a successful architecture project. This includes ensuring the necessary management support, identifying the business goals to be achieved by the architecture project, and reviewing the relevant architecture and business principles. Defining stakeholders, concerns, scope, and constraints of the project, and summarizing all these issues in a statement of work are also of concern here.

B: Business Architecture This process aims at describing the actual and target business architecture, and the gaps between them. Thus, views adequate for demonstrating how the stakeholders' concerns are met have to be selected and created. Examples are business process models or class diagrams focusing on business concepts.

C: Information Systems Architecture This process is concerned with developing target architectures for data and business applications. Thus, the focus is on the business processes as described in step B that are supported by IT. In addition to the respective architectural descriptions, the outputs of the activity can also include suggestions for changes to the business architecture, which are made necessary by findings from step C.

D: Technology Architecture This process builds on the architectures developed in the previous processes (business architecture, data architecture, applications architecture). Also building on the technology principles of the organization, a target technology architecture, including a gap analysis to the current technology architecture, is created. *Technology architecture* models are thus organized into architecture building blocks, for which the following is modeled: Details of the technical functionality, a list of all necessary standards, and a description of the building block detailed enough for implementation.

E: Opportunities and Solutions In the development of the target architectures described above, different implementation options have been identified, with basic options being, e.g., build vs. buy, re-use of existing business applications, or development of a new application. The goal of this process is to select among these options, and define work packages, leading to an implementation and migration strategy, a high-level implementation plan, and a list of impacted projects.

F: Migration Planning This process is concerned with prioritizing the implementation projects, considering costs, benefits, and logical dependencies of the projects. Among its outputs are a detailed implementation and migration plan, and, if necessary, the respective contracts.

G: Implementation Governance The *implementation governance* process is concerned with the actual software development projects implementing the designs created in the above processes. Its goal is to give architecture-specific support to the implementation projects, e.g., via implementation recommendations, and to ensure that the projects result in architecture-compliant systems. The process is thus conducted in parallel with the software development processes, but is not identical with them.

H: Architecture Change Management This process starts with the completion of process G. Its objective is to manage the future evolution of the architecture. Basically, it has to evaluate proposed changes to the architecture, and decide

- under which conditions the EA, or a subset of it, is allowed to change, and how this has to happen, as well as
- under which conditions a new ADM cycle is started.

The circle labeled *Requirements Management* in the middle of Figure 2.3 signifies a central process driving the other ADM processes. The requirements management process is responsible for identifying and documenting requirements, and subsequently communicating them to the other ADM processes.

Besides the processes, TOGAF makes some suggestions on how to document the deliverables created in the processes of the ADM. The *Enterprise Continuum* is the (virtual) repository, where all kinds of architecture assets are stored. These are the deliverables created in the execution of the ADM, but also industry-wide, or generally applicable assets (e.g., TOGAF itself) belong here. While TOGAF does not offer an integrated metamodel for storing these information, it proposes selected viewpoints for some deliverables suggested by the ADM.

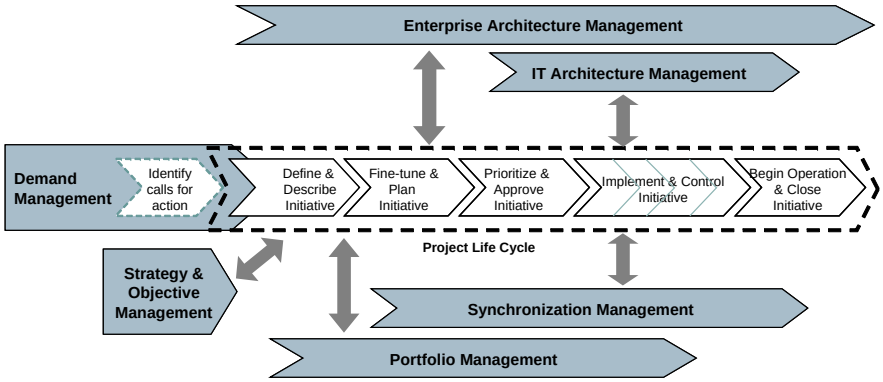


Figure 2.4.: Processes involved in managing an application landscape [Wi07]

2.1.3.1.2. Wittenburg

[Wi07, WM07] introduce a set of processes for managing an application landscape, as shown in Figure 2.4. The specific goals are

- documenting the application landscape,
- planning the evolution of the application landscape, and
- identifying potential for improvement, including improving the alignment of business and IT.

Demand Management collects and documents emerging requirements on the application landscape. Suggestions for changing the IT are inputted into this process, and then coherently documented. A committee may then decide, whether to accept or reject a demand.

This process should not be mixed up with requirements engineering in software engineering, which is usually concerned with a similar subject at a finer granularity. More important here is linking the demands to the goals and strategies they are meant to support.

Project Portfolio Management is then responsible for creating an optimal portfolio of projects to be executed, based on the demands documented according to the *Demand Management* process. The process starts with bundling the demands into project proposals; thus, a demand may result in one or more proposals. The description of the project proposals is not limited to the actual work packages and milestones, and includes risk and cost assessments. This provides a basis for prioritizing the projects and assigning corresponding budgets, leading to some projects being approved, and others rejected.

Strategy & Objective Management is responsible for enabling demands and projects to be linked to strategies. It operationalizes strategies into specific goals. This is supposed to make transparent why certain changes have been or have to be performed. The transparency is in turn supposed to increase the alignment of the project portfolio with

the business strategy. A possible implementation of this process can, e.g., be guided by the Balanced Scorecard approach [KN91, SK04].

IT Architecture Management introduces homogenization aspects, by prescribing standardized architectures and technologies to be used by specific business applications. Standardized architectures prescribe a certain architectural pattern for each business application (e.g., a client-server or a web architecture), while technology standards impose constraints on the specific products used in implementing them. This also works towards a focus on tried and tested best practices in large application landscapes.

Synchronization Management is concerned with delayed projects, deriving the effects of a project delay on other projects, of which, e.g., the start might have to be postponed, as they rely on certain results from the delayed project. Thus, synchronization management should identify and handle the delayed projects as early as possible.

EA management documents and manages the interaction of the elements of concern in the above described management processes, which are basically elements according to the *kinds of entities in an application landscape* introduced in Figure 2.1. By providing adequate documentation, the process makes interactions and relationships not directly apparent in the other processes themselves visible. For creating these documentations, [Wi07] proposes different kinds of software maps, which are introduced in Section 2.1.3.2.

2.1.3.1.3. Ross, Weill, and Robertson

Ross, Weill, and Robertson approach the issue of EA management more from a business-centered perspective [RWR06]. Their approach is based on the finding that successful enterprises, in terms of profitability, time to market, and IT costs, are distinguished from the less successful ones by a solid *foundation for execution*.

The *foundation for execution*, as shown in Figure 2.5, encompasses the IT and the IT-supported business processes that automate a company's core capabilities. Processes belonging to the core capabilities can thus range from everyday administrative processes to key processes realizing strategic competitive advantages. The advantage of automated core capabilities is that they no longer divert management attention to low-level problem tackling, allowing it to focus on issues important for creating value.

In order to build such a foundation for execution, a company has to make basic decisions on how it conducts business, here called its *operating model*. The operating model is mainly concerned with issues regarding

- process integration, i.e. to which extent processes in different business units operate on shared data, and
- process standardization, i.e. to which extent processes in different business units are standardized.

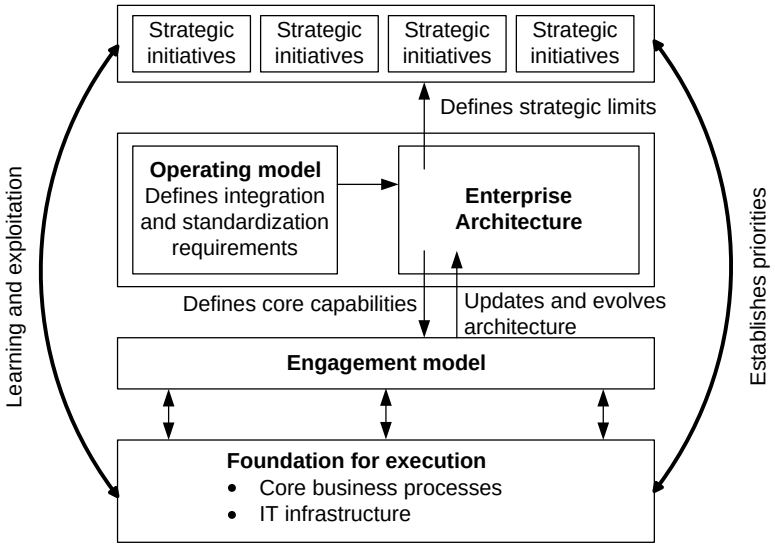


Figure 2.5.: Creating and exploiting the foundation for execution [RWR06]

As the foundation for execution is built according to this operating model, the operating model can not be easily changed, but allows agility within its limits. As shown in Figure 2.5, strategic initiatives are thus limited by the *EA* designed according to the operating model. In turn, the strategic initiatives can exploit the advantages of a solid foundation for execution.

In this context, the EA is the organizing logic guiding how business processes and IT are built, which are thus designed in accordance with the chosen operating model. This constitutes a rather high-level definition of *EA*, which is detailed at the more fine-grained levels *business process architecture*, *information architecture*, *applications architecture*, and *technology architecture*.

To build a foundation for execution, an *IT engagement model* has to provide governance mechanisms aligning IT to both local and companywide objectives, by linking senior-level IT decisions (project portfolio decisions, companywide process design) and project-level decisions. This happens in accordance with the EA, but also further evolves the EA. While the EA limits strategic initiatives, these initiatives influence of course the priorities of the projects building and evolving the foundation for execution.

2.1.3.1.4. Summary

The above outlines of three approaches for EA management show that the field is characterized by very diverse approaches with differing definitions of key terms. However, some key characteristics of the field can be summarized as follows:

- EA management involves both IT and business.

- The application landscape, or application portfolio, plays a role in EA management, although at different levels of granularity in the different approaches.
- EA management is a long-term activity involving multiple, diverse stakeholders.

2.1.3.2. The Role of Visualizations: Software Cartography

Section 2.1.3.1 highlighted the central role taken by the application landscape in EA management. As this work specifically focuses its metrics-based approach to supporting EA management on the application landscape, software cartography as a means for visualizing an application landscape is now introduced, also for visualizing metrics.

Software cartography deals with the design and creation of visualizations of application landscapes, called *software maps*, making use of concepts from (conventional) cartography. Basically, software maps are graphical models of application landscapes visualizing different aspects, which correspond to the concerns of various stakeholders.

In cartography [Ro95, Sl05], a map consists of a base map, and possibly some layers. The base map is made up by a two- or three-dimensional space, typically a topographical map showing the earth or parts of it. A thematic map uses a topographical base map to visualize certain aspects like population density or political election results on layers. In software cartography [LMW05a, Wi07], the same layering principle is applied to visualize different aspects of elements in the application landscape. Unfortunately, no two prominent topographical characteristics such as longitude and latitude exist in software cartography. Therefore, different types of software maps are distinguished, which are categorized by the rules for building the base map.

To understand the elements of a software map, the following three terms are important [Er06]:

Map Symbol Map symbols are the graphical elements of a software map. Examples are *rectangles*, *circles*, *chevrons*[3], and *traffic lights*.

Visual Variable A map symbol owns a set of visual variables, influencing the graphical representation of an instance. Different map symbols own different visual variables, e. g., a rectangle owns the visual variables *centerPoint*, *width*, *height*, *borderColor*, *fillColor*, *borderStyle*, *textAttribute*, etc.

Visualization Rule A visualization rule defines visualization constraints or targets. A *constraint visualization rule* demands a specific relationship between map symbols, e. g., the nesting of a map symbol instance inside another map symbol instance. A *target visualization rule* may, e.g., demand area minimization of a map symbol instance. Constraint visualization rules must be fulfilled to visualize the information in a correct way. Target visualization rules should be fulfilled to the highest possible degree to result in aesthetically appealing visualizations.

Subsequently, the different software map types are introduced.

[3] A Chevron is a fish symbol, which is often used to represent a business process.

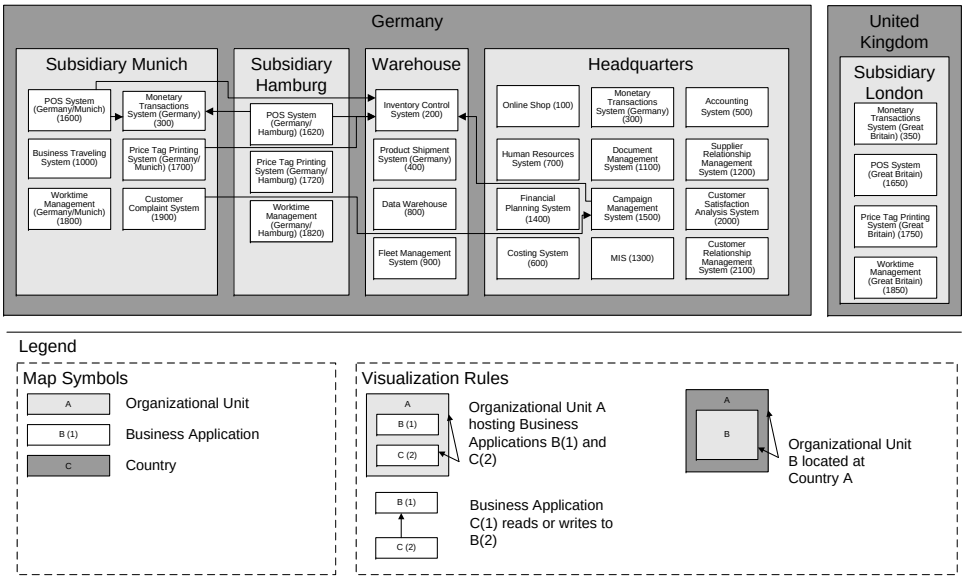


Figure 2.6.: Example of a cluster map

2.1.3.2.1. Software Maps with a Base Map for Positioning

For software maps *with* a base map for positioning, the position of an element on the base map conveys information. If the position of an element is changed on the base map, its meaning may change, too. [Wi07] introduces two software map types in this category: *cluster map* and *Cartesian map*.

Cluster Map A cluster map is a software map, which uses logical units (named *clusters*) to build the base map and to group business applications in these units. This grouping is visualized by *nesting* the graphical representations of business applications into the symbols representing the corresponding logical unit, expressing a specific relationship between the logical unit and the *nested* business applications.

Logical units are, e. g., organizational units, functional areas, or (geographic) locations. According to that, the nesting may be used to visualize a *hosted-at*-relationship between a location (e. g., a computing center) and a business application.

Figure 2.6 shows an example of a cluster map[4], visualizing which location *hosts* which business applications. The nesting of the rectangle *Online Shop (100)* inside the rectangle *Munich* has the meaning that this location hosts the business application.

[4]While [Wi07] prescribes a header containing creation date and a contact organization for each software map, this work does not include such an element in its software maps. In Chapters 2-4, it would contain only exemplary information, in Chapter 5, the information would have to be removed for reasons of anonymization.

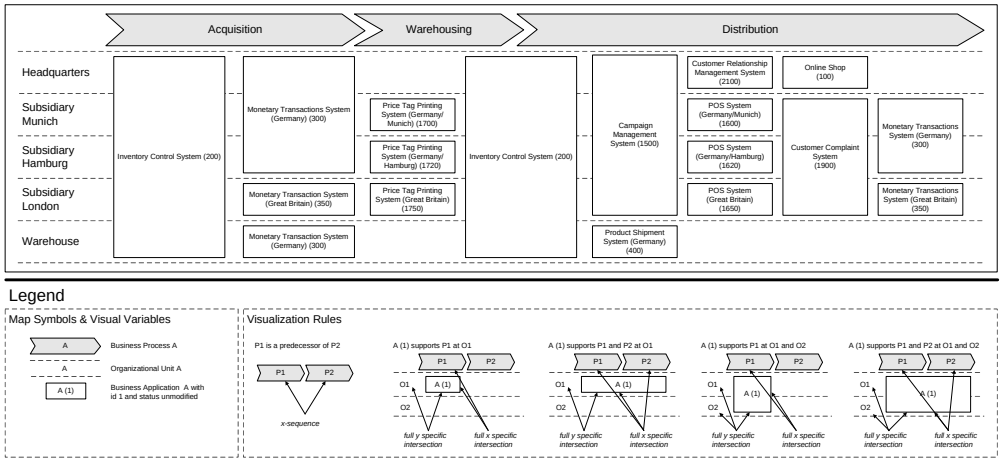


Figure 2.7.: Example of a process support map

This map type does neither specify how the clusters should be placed on the base map nor how the different elements, nested in a logical unit, should be placed in relation to each other. Thus, area minimization can, e.g., be demanded of a good layout.

Cartesian Map A Cartesian map is a software map using two axes (dimensions) for creating the base map. Therefore, the base map of a Cartesian map relies on two aspects of the EA for determining the x-axis and y-axis. [Wi07] identifies the *process support map* and the *time interval map* as the most important kinds of Cartesian maps.

The base map of a *process support map* positions the business processes on the x-axis and, e.g., the organizational units or system types on the y-axis, building a grid-like positioning schema. Figure 2.7 shows an example of a process support map. The position of the rectangle labeled *Inventory Control System (200)* (for the business application) below the chevron *Acquisition* (for the business process) and beside the labels *Headquarters*, *Subsidiary Munich*, *Subsidiary Hamburg*, *Subsidiary London*, and *Warehouse* (for the organizational units) means that this business application supports this business process at the different organizational units.

The base map of a time interval map resembles the basic buildup of a Gantt diagram, with the time used on the x-axis. The y-axis is made up of the elements, of which development in time should be visualized. The exemplary time interval map in Figure 2.8 shows the life-cycle of business applications and their versions. The rectangles positioned below the time line and beside the business application/business application version visualize the status of the respective element on the y-axis. The color coding of the rectangles is used to distinguish different states, like *planned*, *in development*, *in production*, and *in retirement*.

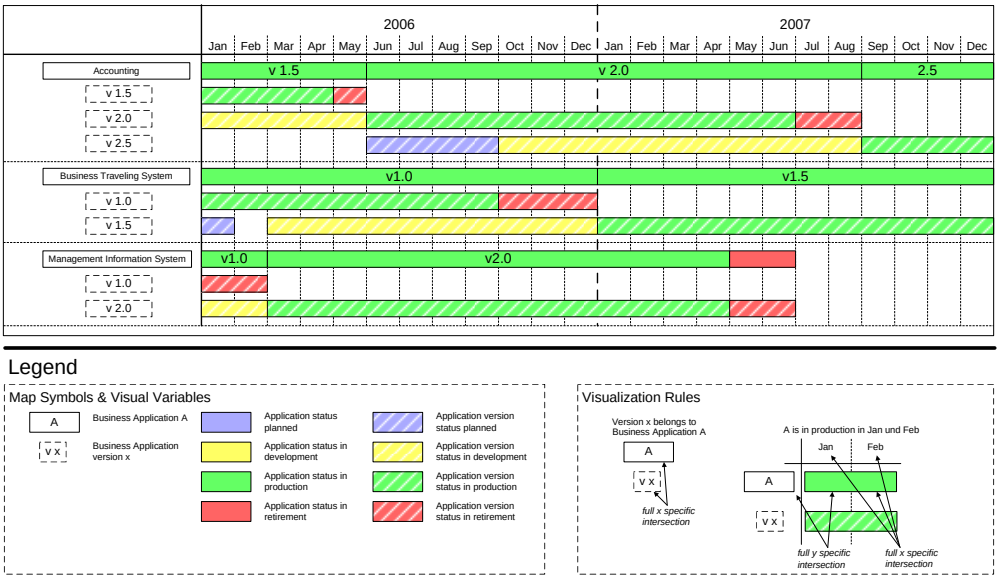


Figure 2.8.: Example of a time interval map

2.1.3.2.2. Software Maps without a Base Map for Positioning

In addition to the map types already introduced, there are also maps where the position of the symbols on the base map has no specified meaning. The decisions concerning the positioning of symbols on the base map can thus be made arbitrarily by the map creator. Positioning can, e.g., be used for aesthetic purposes as in a multitude of other graphical models, as, e. g., UML class diagrams [OM05b], Entity Relationship diagrams [KNS92], or representations of graphs via nodes and edges. The last type of graphical models is the reason why software maps without a base map for positioning are also called graph layout maps. Usually, this type of map is employed if the user has to create a visualization optimally suited for a very specific problem, e. g., an impact analysis.

Figure 2.9 shows an example for a graph layout map. To create this map, a layout algorithm has been used, which centers a symbol representing a user-chosen business application and surrounds it by symbols representing business applications, which are connected to the central business application.

2.1.3.2.3. Layers in Software Maps

Software maps are graphical models of the application landscape, and visualize business applications, their attributes, and their relationships to other elements of the EA. As it is clearly not advisable to display this multitude of different aspects at the same time, software maps support a principle for reducing the visualization complexity, the layering principle. According to this principle, different aspects, as, e. g., operating costs of business applications or availability measures, are displayed in different layers. For any of these layers, except the base map, a *reference layer* exists, where the referenced symbols

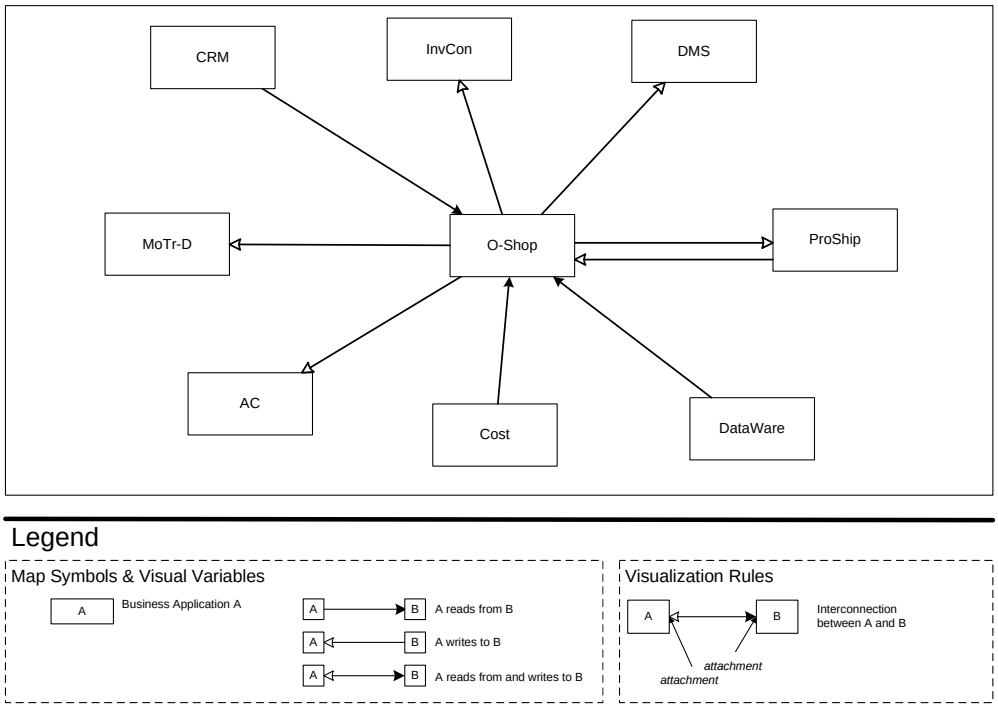


Figure 2.9.: Example of a graph layout map

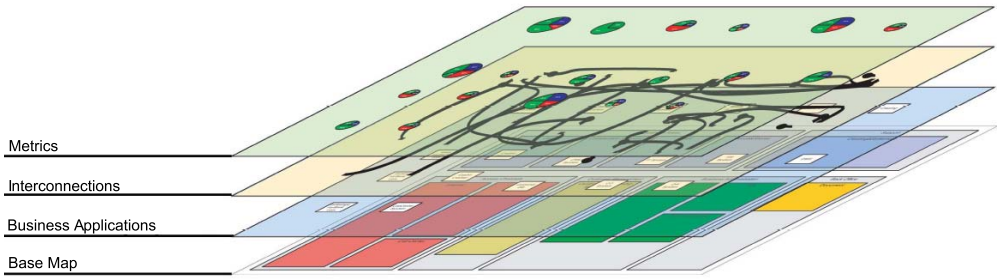


Figure 2.10.: Layering principle of software maps

are seated on; this layer can be the base map. Symbols on a layer can then be placed with respect to symbols on the respective reference layer. Figure 2.10 illustrates the layering principle. It visualizes, among others, metrics for specific business applications as pie charts placed over the respective application rectangles.

A main advantage of the layering principle is the reduction of visualization complexity, as the layers can be shown or hidden as desired by the user, thus varying the information density of the map. Additionally, the layering principle is advantageous, as, e. g., certain business applications are easily recognized throughout the different visualizations due to their unchanged positions on the map. Therefore, the stakeholders can more easily read the information in a visualization.

2.2. Measurement Theory and Metrics Utilization

Section 2.1 introduced application landscape management and EA management as problem areas, and pointed to approaches used in this domain. In order to facilitate applying metrics in this area, basics of measurement theory and metrics usage are now introduced.

2.2.1. Theoretical Foundations: Measurement Theory

Many things are measured, e.g., sizes of everyday objects, possibly via a tape measure, sizes of galaxies, or of atoms, intelligence of human beings, or satisfaction of an anonymous customer base with products.

The basic definition, underlying such diverse use cases, is that "the process of measurement can be described as assigning numbers to objects of classes in a way to faithfully represent certain properties" [Kr71]. Measurement is thus based on a *homomorphism*, a structure preserving mapping from an *empirical relation system* into a *numerical relation system*.

An *empirical relation system* can be defined as $< M, R_1, ...R_n >$. M is a set of elements, and the R_i are different kinds of relations of possibly varying rank. *Empirical* means that M and the R_i are, by certain means, observable.

Examples are as follows:

- A set of rods A, and a relation $\succeq$ indicating whether a rod is of equal or greater length than another, which is here assumed to be observable by holding the rods together.
- A set of balls A, and two relations, a binary relation $\succeq$ and a ternary relation $\oplus$. $\succeq$ indicates whether a ball is of equal or higher weight than another one. Each triple $(a_1, a_2, a_3) \in \oplus$ contains balls so that a_1 and a_2 together have the same weight as a_3. Comparing the weights is here supposed to be possible using a balance.

A homomorphism, e.g., from $< A, \succeq, \oplus >$ to $< \mathbb{R}, \geq, + >$ as described above can be defined as a function $v : A \rightarrow \mathbb{R}$, for which the following holds:

- $a_1 \succeq a_2 \iff v(a_1) \geq v(a_2)$
- $v(a_1 \oplus a_2) = v(a_1) + v(a_2)$

The numerical relation system used determines the *level of measurement*. If it, e.g., supports the relations $+$ and $\geq$, one measures at an interval scale. If only $\geq$ is supported, measurement is merely possible at an ordinal scale. Several levels of measurement can be distinguished [FHT96, Fa99]:

1. Nominal scale: For the nominal scale, the different attribute values have to be distinguishable. Measurements may be transformed by injective functions without losing information. Possible operations are counting the frequencies of the different attribute values.

   Examples: nationality, color, Social Security number

2. Ordinal scale: The different values of an ordinally scaled attribute can be ordered, however; the size of the distances is undefined. Thus, measurements may be transformed by strictly monotonic increasing functions without losing information. In addition to operations possible on the nominal scale, comparisons are supported.

   Examples: school grades, wind intensity measured using the Beaufort scale, the rods example from above

3. Interval scale: In addition to the properties of an ordinal scale, the differences between the attribute values can be interpreted. Measurements may be transformed by positive-linear functions of kind $v' = \alpha v + \beta \quad (\alpha, \beta \in \mathbb{R}, \alpha > 0)$. Besides the operations possible on the ordinal scale level, addition and subtraction are sensible operations on an interval scale.

   Examples: temperature measured using the Celsius scale

4. Ratio scale: In addition to the properties of the interval scale, the zero of a ratio scale is non-arbitrary. Information-preserving transformations are thus limited to $v' = \alpha v \quad (\alpha > 0)$. This allows defining ratios between arbitrary attribute values, e.g., a is twice as heavy as b. Thus, division is added to the set of possible operations.

   Examples: prices, lengths, effort in person-months

Thus, the higher the level of measurement, the more information about the unit of observation is derived by measuring, indicated by more operations making sense on the measurement values, and transformations distorting the information more easily.

The theoretical background outlined above reveals a basic limitation of metrics. Measurement on a specific level demands that an empirical relation system sufficient for constructing the respective homomorphism exists.

Taking the first example from above (different rods), it is not possible to measure the rods' length on the level of an interval scale, as the empirical relation system given only contains information sufficient for ordering. Thus, measurement is possible only on an ordinal scale.

A similar example is given by Fenton, pointing to the limitations of measuring software complexity [FP97]. The example encompasses three structures, as shown in Figure 2.11. While it is obvious that x is less complex than y, inserting z into this order is basically not possible without an arbitrary decision.

Thus, an empirical relation system sufficient for ordinal scale is not present on the structures, and their complexity can thus not be measured. Of course, when focusing on

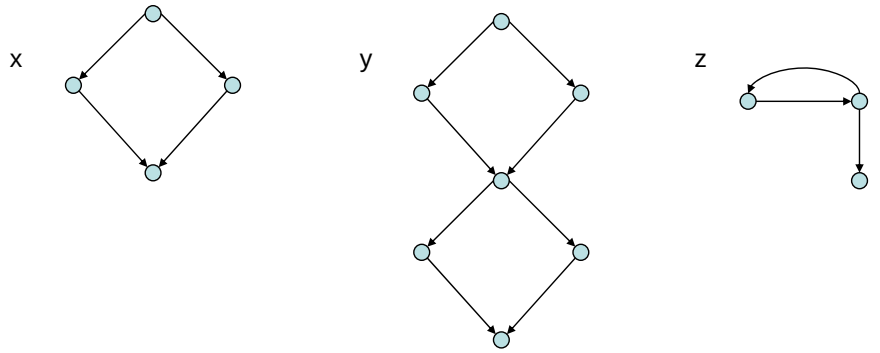


Figure 2.11.: Three graphs, which cannot be strictly ordered by complexity

a specific aspect, e.g., the number of nodes or edges, this aspect of complexity can be measured.

2.2.2. Applying Metrics

Use of metrics is common to a multitude of domains, e.g., natural sciences, where many of the foundations of measurement theory originated from. Further developments have been introduced in social sciences [Kr71] or economics [Ei87].

This section briefly outlines how metrics are used in software engineering and business administration. On the one hand, this is supposed to illustrate how metrics can be used. On the other hand, describing how metrics are used in these two areas bordering on the management of application landscapes is meant to provide a contrast for the description of metrics usage in managing application landscapes in Chapter 3.

2.2.2.1. Metrics in Software Engineering

Metrics in software engineering target a wide range of aspects, from software development processes to the different products created therein, as architectures, source code, or the developed software itself.

Examples for metrics measuring software development processes are productivity metrics, e.g., LOC/person-day or function points/person-day, or the CMMI measurement [CM06, CM07] for determining process maturity.

Regarding product metrics, research into metrics targeting the structure of software dates back to McCabe's cyclomatic complexity [Mc76] and the Halstead metrics [Ba98]. These metrics have been designed to capture aspects of size or complexity of a program written in a procedural programming language.

Different sets of object-oriented metrics, e.g., the MOOD [Ab95] and the MOOSE [CK94] catalog, have been developed to measure specific aspects of object-oriented software, as

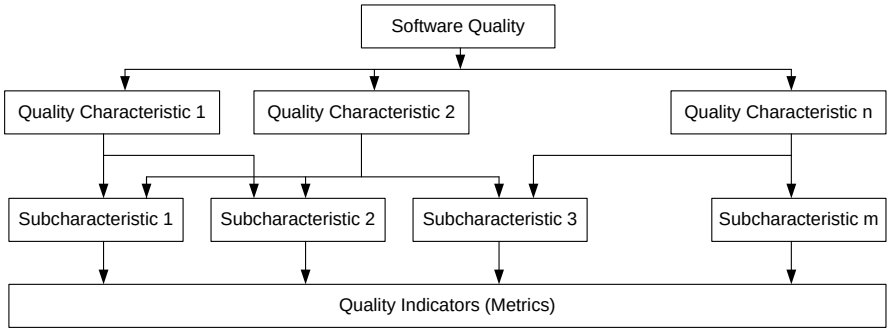


Figure 2.12.: Structure of an FCM-model

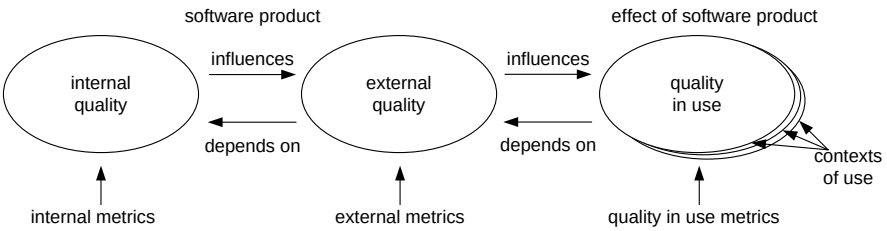


Figure 2.13.: Kinds of metrics proposed in ISO 9126 [IS03a]

coupling of classes, use of polymorphic behavior, inheritance, encapsulation, or information hiding.

In order to use product metrics before source code has been written, metrics specifically measuring software architecture have been developed. Metrics measuring properties of an object-oriented design, without needing the source code, have, e.g., been proposed in [Be98] or [BD02].

Product metrics can be employed in software engineering according to several usage scenarios. The most straightforward scenario uses metrics as quality indicators. This is, e.g., described in the standard ISO 9126 [IS01, IS03a, IS03b, IS04], which proposes a set of quality attributes for software, and different kinds of metrics as indicators for these quality attributes.

The quality attributes are structured into a factor criterion metrics model (FCM-model). A FCM-model refines *software quality* into subcharacteristics, leading to a tree- or net-like structure. Quality indicators, e.g., metrics, appear at the lowest level of this structure. Figure 2.12 shows an exemplary FCM-model. The FCM-model of ISO 9126 has two levels of quality attributes. The quality attributes at the first level of the model are *Functionality*, *Reliability*, *Usability*, *Efficiency*, *Maintainability*, and *Portability*.

Regarding the metrics used as quality indicators, the standard distinguishes three kinds of metrics, as shown in Figure 2.13. Thus, internal quality relates to intermediate deliverables in product construction, and can thus be already measured during development.

It allows making predictions about external quality, which characterizes the behavior of the system. Quality in use is concerned with whether the product, with its behavior, meets the needs of the users in a specified usage context.

Making use of the fact that metrics are tied to quality attributes is not limited to making predictions about a system. [Ve05], as an example, propose a process in which metrics guide the review of a system. This is achieved via the following steps:

1. A metrics specialist performs an initial metrics analysis, based on a model of the system under consideration.
2. The metrics specialist screens the metrics derived from the initial analysis. This results in a first set of questions.
3. The metrics specialist performs a comprehensive metrics analysis, to get answers to the questions derived in the previous step.
4. The metrics specialist discusses the results from the comprehensive metrics analysis together with the architects of the system.

This process is meant to focus the attention of the architects in the review workshop on the important issues, and thus supply improved information about the problems discussed. The process does not rely on statistically explicating the relationship between the metrics and certain quality characteristics, e.g., by estimating a regression model, but uses both metrics and knowledge from domain experts.

[LTC02] used metrics for tracking the evolution of a software. The goal was to examine whether the software has deviated from its initial design far enough to justify a project for restructuring.

2.2.2.2. Metrics in Business Administration

Business administration, ranging from strategy to operations research, from human resources to finance, encompasses a multitude of fields where measurement is essential. The subsequent list shows some applications of metrics in business administration:

- Different kinds of metrics for the efficiency or performance of employees exist. Such metrics can be rather straightforward in factory work, e.g., numbers of processed parts per hour. They may however also get more sophisticated, if, as in *Management by Objectives*, where managers are concerned, the work is less easily measurable [Dr77].
- For processes, e.g., in production, a plethora of metrics exists. Core subjects to be measured include efficiency, failure rates, or adherence to delivery dates (see e.g. [Ro02]).
- In marketing and distribution, aspects as customer satisfaction, or the success of marketing actions needs to be measured [Ba07].

- Measures in finance and accounting ratios target the financial profitability and stability of an organization.
- The balanced scorecard is a metrics-based approach for operationalizing and enforcing strategies.

Measures in finance and accounting ratios, and the balanced scorecard are subsequently presented more in detail.

2.2.2.2.1. Metrics in Finance and Accounting

Finance is to a considerable extent concerned with the profitability and the liquidity of an organization. This motivates a need for metrics, which allow making statements about the extent to which those goals are met by an organization.

There are several metrics operating on balance sheet data, which are commonly used to determine whether an organization is financially healthy, or is in danger of becoming illiquid, among others, the following [RWJ02, Dr99]:

- Debt ratio=$\frac{Total\ \ debt}{Total\ \ assets}$; This metric gives information about to what extent creditors are protected from insolvency, and the ability of firms to obtain additional financing.
- Quick ratio=$\frac{Quick\ \ assets}{Total\ \ current\ \ liabilities}$; Quick assets are assets quickly convertible to cash. Thus, the metric gives information about to what extent an organization is able to pay off current liabilities without selling inventories.

By relying on balance sheet data, the metrics are defined on a rather sound basis. Accounting can rely on a high number of largely mandatory laws, regulations, and best practices. To a specific extent, such metrics have been empirically verified [Dr99]. In validating accounting ratios, publicly available data from financial reporting can be helpful.

Other accounting ratios can be used for assessing profitability, e.g.:

- (Net) Return on assets: $ROA = \frac{Net\ \ income}{Average\ \ total\ \ assets}$
- Return on equity: $ROE = \frac{Net\ \ income}{Average\ \ stockholders'\ \ equity}$

Return metrics are used in sophisticated models, e.g., the *Capital Asset Pricing Model* (CAPM). This model proposes a relationship between the risks associated with the return of a stock and the appropriate risk-adequate expected return of this stock [RWJ02].

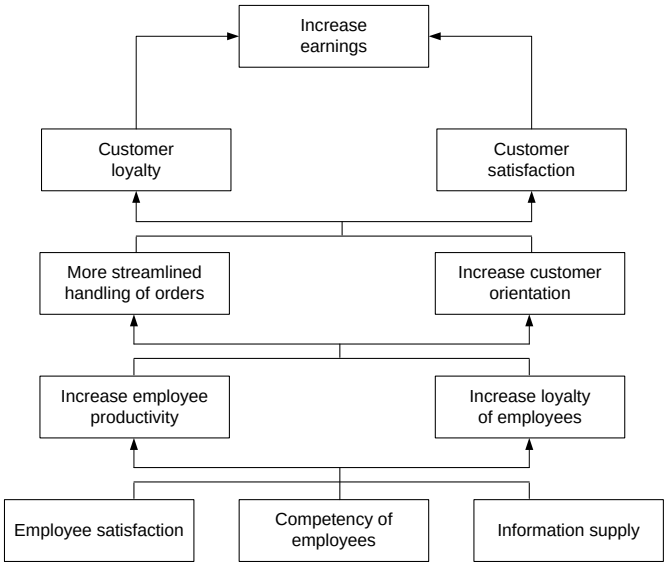


Figure 2.14.: Example of a strategic linkage model (adapted from [SK04])

2.2.2.2.2. The Balanced Scorecard

The balanced scorecard is a metrics-based approach for operationalizing and enforcing strategies. A core concept of a balanced scorecard is a strategic linkage model, showing strategic objectives and how they influence each other. Based on the strategic objectives, metrics are selected, operationalizing abstract strategic objectives by making them measurable. Figure 2.14 shows an example of a strategic linkage model, with the different strategic goals in the boxes, and the arrows indicating (positive) influences.

Two kinds of metrics used on a balanced scorecard are basically distinguished, with a good balanced scorecard containing a mix of them.

- Lagging indicators: metrics showing the consequences of actions previously taken; they measure historical performance, typically focusing on a specific space in time, e.g., the number of sales or revenue.
- Leading indicators: measure the drivers of lagging indicators; thus, these measure the reasons for changes in lagging indicators.

Explicating and operationalizing strategic goals as described above helps in communicating them in the organization, giving organizational units and their employees the information to behave conforming to the strategy. The structure of the balanced scorecard, with the strategic linkage model, is suitable for breaking goals down into sub-goals for specific organizational units and helps in propagating and communicating the strategy.

In this context, the balanced scorecard approach also encompasses setting goals, as target values for specific metrics, which are checked at pre-defined milestones. Together with

setting the target values, plans for reaching these values can be made. Another important aspect of the approach is feedback and learning, including adequate reactions, if target values are not reached at a milestone check.

[KN91] organized their goals and metrics into four perspectives: the financial perspective, the customer perspective, the internal process perspective, and the learning and growth perspective. However, these perspectives are not obligatory, and especially when focusing on IT, other perspectives might be more useful. [SK04] proposes the following perspectives for a IT balanced scorecard:

- **Contribution to business[5]:** concerned with the value IT creates by supporting business processes
- **Customers:** concerned with the customers of IT, e.g., its users, and customer orientation of the IT
- **Delivering IT benefits[6]:** concerned with operative value creation in IT, i.e., the processes of IT itself
- **Utilization of IT[7]:** concerned with what technology IT uses for value creation
- **Future:** concerned with the future, e.g., learning and building knowledge

When finding metrics for aspects in these perspectives, the application landscape may be especially of importance with *contribution to business*, *delivering IT benefits*, and *utilization of IT*.

With *managing application landscapes* described as the problem domain, and *metrics* set as the instrument to be applied to the domain, Chapter 3 discusses, in general, how metrics can be applied to managing an application landscape, and how this is currently done.

[5] In German: Unternehmensbeitrag
[6] In German: IT-Leistungserstellung
[7] In German: IT-Einsatz

CHAPTER 3

Applying Metrics to Application Landscapes

Contents

Chapter 2 introduced managing an application landscape as the problem domain, and outlined how metrics work as a solution instrument. Now, Section 3.1 explores the environment, in which metrics are currently employed to support managing an application landscape. Based on this, Section 3.2 proposes guidelines for constructing metrics and metrics-based methodologies targeted at this environment.

3.1. Environment Analysis and Related Work: Status Quo of Metrics Usage

The environment analysis rests upon information from different sources:

- Literature about metrics for application landscapes and related subjects, as software engineering and business administration. Literature mainly aids in exploring, what approaches are currently available.
- The *Enterprise Architecture Management Viewpoint Survey (EAMVS)* conducted by the Chair for Informatics 19 (sebis) of the TU München in 2007, in which practitioners were surveyed on how they approach EA management by guided interviews and an online questionnaire [Bu08b]. The environment analysis mainly relies on primary data to explore, to what extent and how practitioners want to use application landscape metrics, as there were, at the time of the work, no known surveys providing this information at the level of detail needed for an extensive analysis[1].

3.1.1. Surveying the Status Quo

In order to describe how the EAMVS collected information about EA management and metrics, the subject of interest in this work, the survey with its guided interviews[2] and online questionnaire is now described. Then, Section 3.1.2 presents the metrics-related findings.

3.1.1.1. EAMVS Interviews

During the EAMVS, 22 practitioners from 19 organizations were interviewed about how EA management is approached in the organizations they work for. The interviews, conducted as guided interviews, were meant to explore EA management, with its problems, stakeholders, concepts, and approaches [Bu08b]. Specific questions about metrics were included into the interview schedule[3]. This interview schedule is presented in Appendix A.1.1, both in the original German version, as this is the language in which the interviews were conducted, and as an English translation. Translated into English, the questions about metrics were as follows:

- Which properties of your current EA do you want to improve?
- Further clarification: please state the three most important quality attributes for an EA, and indicate the following:
  - Do you use metrics in this context?

[1] [In06] is a rare example of a survey taking into consideration EA management metrics, but only touches the subject with two questions.
[2] In German: Leitfadeninterview
[3] In German: Interviewleitfaden

Business	**Organizations**	**Interviewed Persons**
Finance and Insurance	7 (1 medium)	8
Information (here: Telcos)	2 (1 medium)	2
Manufacturing	6 (2 medium)	7
Professional, Scientific, and Technical Services	1	1
Transportation and Warehousing	3	4

Table 3.1.: Interviewed persons in EAMVS

- Could you imagine using metrics in this context?
- If not, what is the reason, and if yes, what do you expect from using metrics?

Table 3.1 gives an anonymized overview of the interviews, showing that representatives from a diverse set of businesses were interviewed. The focus was on larger organizations, with four middle-sized ones rounding off the set. The job profile of the interviewed persons was mainly centered on technical aspects of EA management and IT architects. The business classification used in Table 3.1 is organized along the categories from the *North American Industry Classification System (NAICS)* [EO99].

The size classification is based on the number of employees of the respective organizations, where organizations with fewer than 20,000 employees are classified as *medium*, and organizations with a higher number of employees are classified as *large*. If an organization is considered a branch or division of a larger organization, the size classification is tied to the size of the superordinate organization. In one case, an IT provider has been classified as large, although it has fewer than 20,000 employees. The reason is that this organization is the outsourced IT of a much larger banking organization.

The interviews took place from 1/2007 to 3/2007. They were performed by two interviewers at the sites of the respective companies with, in most cases, one, in some two or three interview partners. Each interview took around 1 to 1 1/2 hours. The interviews were recorded and then transcribed. Then, the interviewed persons received the opportunity to review the interview transcripts, to correct misunderstood issues or remove confidential details.

This work uses interview passages regarding metrics for application landscapes, which it has extracted from the interview results and summarized in tables. In this context, the interviews were classified along the categories *"Using metrics"*, *"Metrics usage planned"*, *"Metrics possible, but many issues are not clear at the moment"*, and *"Metrics not possible, makes no sense"*. These summaries are available in Appendix A.1.2, and are analyzed with respect to the status quo regarding application landscape metrics in Section 3.1.2

3.1.1.2. EAMVS Online Questionnaire

Section 3.1.2 also uses results from an online questionnaire, which was part of the EAMVS. This questionnaire was given to practitioners, to analyze how visualizations

and the methodologies using them support EA management [Bu08a, Bu08b]. Sections about application landscape metrics were introduced into this online questionnaire to support this work in its status quo analysis.

3.1.1.2.1. Questions about Metrics for Application Landscapes

The parts of the online questionnaire concerned with metrics and used in this work are subsequently given in an English translation. Appendix A.2.1.1 contains a copy of the original German language questionnaire sections.

Nr.	Question	Possible Answers	
Section IIX: Concluding Questions			
1	Visualizations in EA management		
1.4	How big is your application landscape (number of installations of business applications)?		number
2	Best Practices and Trends		
2.1	How would you characterize the process in which the goals for the evolution of the application landscape are set?	Options: Goals are not explicitly documented, but distributed, possibly also in the heads of the respective employees; precise goals exist in a document; No goals are set	
2.5	Are you satisfied with the data quality?	Checkboxes: No, they are not topical enough; No, correctness is insufficient; No, they are not complete enough; Yes, data quality is sufficient; No answer	
2.9	Where in your organizations are the goals for the future evolution of the application landscape set?	Options: Different persons/groups try to set goals, but have no formal power for enforcing them; A person/group (e.g., IT architects) tries to set goals, but has no formal power for enforcing them; A person/group with the necessary power for enforcing them sets goals; Different persons/groups with the necessary formal power set goals	
3	Metrics and Measures		
3.1	Did you (your organizations) use metrics in managing the application landscape up to now?	Options: Yes; Usage possible; Usage not possible	
3.2	How would you characterize the extent of metrics usage in your organization?	Options: Smaller usages (certain employees tried something); Some employees use metrics on a regular basis; Metrics are common in managing the application landscape; Metrics are central to managing the application landscape	
3.3	What is or would be your task in metrics usage in managing the application landscape?	Options: Collecting data; Reworking data; Validating and checking data and metrics; Analyzing data and metrics; Presenting metrics; Using the data for monitoring goals for projects/the application landscape; Planning metrics usage	
3.4	Which attributes of an application landscape would you like to examine using metrics?		
3.4.1	Maintainability: The maintenance efforts occurring in technical development should be small.		Five-point scale[4]
3.4.2	Flexibility: Making changes regarding the business functionality (customer- or market driven) should be swift and inexpensive.		
3.4.3	Testability: Tests with a given test coverage should be possible with as low an effort as possible.		
3.4.4	Performance: The systems of the application landscape should support certain service level agreements (SLAs).		

[4]Here implemented as: 1 (not necessary) to 5 (essential); no Answer

3.4.5	Scalability: The application landscape should support distributing load to new hardware.	
3.4.6	Operational risk: The risks posed by the application landscape should be low.	
3.4.7	Cost advantages in operation: Operating the application landscape, regarding both technical and functional operation, should be inexpensive.	
3.4.8	Installability: Installations should be possible swiftly and at a low effort.	
3.4.9	Functionality: The functionality offered by the application landscape should support business and its process execution.	
3.5	Which units of observation would you want to measure?	
3.5.1	Application landscape as a whole.	Five-point scale[5]
3.5.2	Domains / Subgroups (e.g., ”life insurance”, ”warehousing”).	
3.5.3	Specific components of the application landscape (processes, services, business applications).	
3.6	How would you characterize the tasks you would like to support with metrics?	
3.6.1	Improve understanding of certain problems, to aid in finding a solution.	Five-point scale[6]
3.6.2	Predicting the effects of specific measures.	
3.6.3	Setting goals (as target values for specific metrics) and monitoring goal achievement (operationalizing architectural goals).	
3.6.4	Managing application landscape optimization: Showing the status quo and potential for improvement.	
3.7	What do you expect from metrics?	
3.7.1	I can describe and communicate certain issues in a better way.	Five-point scale[7]
3.7.2	I can understand certain issues more swiftly.	
3.7.3	I am better informed and can make more founded decisions.	
3.7.4	I can get an overview of certain issues more swiftly.	
3.8	What properties of metrics are important to you?	
3.8.1	Simple calculation procedure.	Five-point scale[5]
3.8.2	Employees do not feel controlled by the metric.	
3.8.3	The metric should be well founded and well understood.	
3.8.4	The meaning of the metric should be easily communicable.	
Section IX: Personal questions		
1	Your field of activity	
1.2	In which line of business are you concerned with EA management/managing an application landscape?	Checkboxes: Producers; Service Business; Trading; Agriculture and Mining; Finance; Information; Utilities; Real Estate; Transportation; Other (with text input)

[5]Here implemented as: 1 (not necessary) to 5 (essential); no Answer
[6]Here implemented as: 1 (no potential) to 5 (high potential); no Answer
[7]Here implemented as: 1 (no expectations) to 5 (high expectations); no Answer

2	Your Background/Your Education		
2.1	In which area did you receive your education?	Checkboxes: IT (e.g., Computer Science); Business (e.g., Business Administration); Between Business and IT (e.g., Information Systems); Philosophy or Psychology; Social Sciences; Natural Sciences or Mathematics; Engineering; Other;	
2.2	How do you rate your IT knowledge?		Five-point scale[8]
2.3	For how many years have you been involved in IT?		years
2.4	How do you rate your knowledge regarding the business you support?		Five-point scale[8]
2.5	For how many years have you been involved in this area (business)?		years
2.6	How do you rate your EA management knowledge?		Five-point scale[8]
2.7	For how many years have you been involved in EA management?		years
4	Maturity of the IT		
4.1	Which of the following statements characterize your organization best?	Checkboxes: IT is characterized by applications focused on supporting specific areas, for which it is optimized; A homogeneous, standardized infrastructure is the execution environment of the applications; IT supports standardized processes in the organization and enables these processes to use data owned by the different applications; IT is based on process- and application-components, which can be used in a modular way and thus provide flexibility	

Table 3.2.: Questions about metrics for application landscapes

3.1.1.2.2. Giving Account for the Metrics Questions

The questions from Table 3.2 rely on concepts from three areas: the *usage scenarios* in which the metrics are employed, *demands on metrics*, and the *environmental factors*, characterizing the circumstances in which the metrics are used.

The most basic questions about *usage scenarios* are questions IIX 3.1 and IIX 3.2, concerned with the possibility and extent of metrics usage. This is detailed by questions IIX 3.4.1 to 3.4.9, concerned with quality attributes. The respective set of quality attributes has been developed and refined together with practitioners from software development of a financial service company in an application landscape-related project. The *tasks* from questions IIX 3.6.1-3.6.4 are taken from [FP97].

[8]Here implemented as: 1 (none) to 5 (very good); no Answer

The practitioners have different demands they pose on metrics for application landscapes. Such demands are considered by questions IIX 3.7.1-3.7.4, and IIX 3.8.1-3.8.4.

Both the usage scenario for metrics and the demands on them may be influenced by environmental factors. Questions IIX 2.1 and IIX 2.9 are concerned with setting goals and goal explication. An influence of such characteristics on metrics-related subjects is reasonable, as [FP97] proposes metrics as a means for, among others, making goals measurable. Data quality aspects, of concern in question IIX 2.5, are also stated as a possible influence on metrics usage by [FP97]. As the attitude towards metrics and metrics usage of a practitioner might depend on what his specific task in using metrics would be, question IIX 3.3 asks for this information. In this context, a classification of different tasks connected to using metrics described by [FP97] is used. As using metrics might get more useful and necessary with growing complexity of the application landscape, question IIX 1.4 asks for the size of the application landscape. The questionnaire did, for reasons of simplicity, not try to address finer differences in the definition of a "business application" used by the different practitioners. Therefore, the size data was in the subsequent analysis used only as classifying the practitioners into size classes:

Class	small	medium	large
# business applications	< 500	500-1000	> 1000

For considering the possibility of different attitudes towards metrics in different businesses, question IX 1.2 relies on a line of business classification based on the NAICS [EO99]. Education and experience of the respective practitioner are more personal influence factors, and are targeted by questions IX 2.1 to 2.7. The classification of the fields used therein is an adaptation from the *Dewey Decimal classes* used for classification in libraries [De89].

Lastly, question IX 4.1 considers the *maturity* of the IT utilization and management in the organization under consideration. The maturity concept behind question IX 4.1 involves the stages proposed in [Ro03]. [FP97] suggests a similar influence of CMMI maturity [CM06, CM07] in software engineering.

3.1.1.2.3. Including the Metrics Questions in the EAMVS Questionnaire

The above questions were included in an online questionnaire made up of nine sections as follows [Bu08a]:

- Questions about 30 methodologies with their viewpoints (according to the EA management pattern approach from [Bu07a][9]), organized into seven questionnaire sections. These questions constituted by far the largest part of the questionnaire. The questions for an exemplary methodology are shown in Appendix A.2.1.2.

[9]The EA management pattern approach is shortly sketched in this work in Section 3.2.

- *Section IIX: Concluding questions*[10], with German originals of the parts containing questions about metrics available in Appendix A.2.1.1, while Table 3.2 presents the questions relevant to the analyses of this work translated into English. This questionnaire section is organized as follows:
  - *Visualizations in EA management*[11], concerned with the attitude of the answering person and its organization to using visualizations in EA management.
  - *Best Practices and Trends*[12], concerned with general aspects and trends with respect to how EA management is conducted in an organization.
  - *Metrics and Measures*[13] contains the questions directly concerned with metrics for application landscapes.
  - *EAM Pattern Catalog*[14] contains questions about the attitude towards a catalog of EA management patterns, as proposed by [Bu07a], which were however not relevant to this environment analysis.
- *Section IX: Personal Questions*[15], see Appendix A.2.1.1 for the original questionnaire form, or Table 3.2 for the subset of questions used in the metrics-specific analyses of this work.

3.1.1.2.4. Conducting the Survey

Tests lead to the conclusion that an average respondent trying to completely answer all questions would need about 3 hours of time to finish the questionnaire. Thus, it was made possible to answer it in separate sessions, by saving partial answers and resuming later.

The online questionnaire was given to 39 practitioners. However, not all questions were answered by each practitioner. Consequently, the sample sizes of the specific analyses in Section 3.1.2 vary, and are thus explicitly stated there.

3.1.1.2.5. Using the Survey Results

The actual sample sizes used in evaluations of the questionnaire data mostly range from 20 to 30. Thus, they might, in some cases, be too small to unassailably back some conclusions. However, the status quo analysis of Section 3.1.2 is more of an exploratory nature, firstly concerned with finding reasonable hypotheses about the status quo regarding metrics for application landscapes. Both interviews and questionnaire are thus used for

[10] In German: Abschließende Fragen
[11] In German: Visualisierungen zum Enterprise Architecture Management
[12] In German: Best Practices and Trends
[13] In German: Metriken und Kennzahlen
[14] In German: EAM Pattern Katalog
[15] In German: Fragen zur Person

- giving first confirmations to hypotheses. This can show that they are realistic enough to be worth further research, and give first data to base such research on more specific research questions.
- highlighting hypotheses likely to be wrong, thus also serving as inspirations for new, more promising hypotheses about the status quo.

3.1.2. Related Work and Status Quo of Application Landscape Metrics

Section 3.1.1 introduced the information sources used for the status quo analysis. Based on these information sources, key areas regarding application landscape metrics and their environment are now explored.

3.1.2.1. Maturity of the Field: Available Approaches and Related Work

When searching literature for approaches to metrics for application landscapes, discoveries are most likely located in neighboring domains. COBIT [IT05], for example, is a standard focused on *good practices* for IT activities. It contains metrics for these activities, or processes, e.g.:

- *number of budget deviations*, measuring, e.g., whether the goal *improve IT's cost efficiency and its contribution to business profitability* of the activity *manage the IT investments* is met.
- *percentage of successful data restorations*, measuring whether, e.g., the goal *optimise use of information* of the activity *manage data* is met.

COBIT touches the application landscape in the *Acquire and Implement* section, with counts as:

- *percentage of application portfolio not consistent with architecture*, measuring whether, e.g., the goal *respond to business requirements in alignment with the business strategy* of the activity *identify automated solutions* is met.
- *number of different technology platforms by function across the enterprise*, measuring whether, e.g., the goal *create IT agility* of the activity *acquire and maintain technology infrastructure* is met.

However, COBIT only slightly touches these subjects, as it focuses on IT as a whole. The application landscape as defined in Section 2.1.2 is not its core interest.

Approaches to creating an IT balanced scorecard exist, e.g. [SK04], or [SS07]. However, these approaches also focus on a more holistic level, and often do not delve into details on the application landscape and what exactly can be measured on it.

[BT00] presents an approach for *measuring the flexibility of information technology infrastructure*. Information technology infrastructure is in this context defined in two parts: a technical part, *pertaining to applications, data, and technology configurations*, and a

human one, concerning *the knowledge and capabilities required to manage effectively the IT resources within the organization.* However, this approach uses questionnaires administered to employees as the sole measurement instrument, not relying on counts on the application landscape structure or performance measures derived in its operation.

Regarding the subject of application landscapes itself, some recent papers discuss aspects as quality attributes or nonfunctional requirements. [AD05] introduce indicators for sustainability. However, they do not always provide precise measurement procedures, with the approach thus relying on expert judgments. [SW05] propose a set of goals relevant to finding the optimal degree of integration of business applications. However, the article explicitly points to difficulties in measuring the respective goal achievements, due to a lack of the necessary metrics[16].

Literature actually proposing metrics working on an application landscape is rather rare and recent, with an example proposed in [Ma05a], and an approach centered on workload and performance presented in [Ia05]. Another example is the *Resistance to Change*, a lagging indicator developed and used at Credit Suisse [Mu05].

The Abacus approach [Du05, Li06] to EA management is centered on quality attributes as *Openness*, *Evolvability*, *Modularity*, *Reliability*, and *Total Cost of Ownership*. The approach is based upon making decisions explicitly with respect to what effect they will have on the position taken in a quality space, having, e.g., the above quality attributes as dimensions[17]. However, how the tool implementing the approach exactly measures these qualities, is not directly public information [On07].

Most recently, metrics-based approaches in the area of Service Oriented Architecture (SOA) can sometimes be considered as contributing to application landscape management. Examples include [RSD07], which is mostly centered on lagging indicators/performance measures for services. [Ai07] introduces an approach for deriving a metrics system from business goals via the GQM methodology [BW82], but does not focus on specific metrics themselves.

A subject often considered by articles discussing nonfunctional requirements in the context of SOA is how nonfunctional requirements emerge in service composition. As an example, [BL06] describe the subject of service composition, thereby also motivating the issue of maintaining quality of service when offering composite services based on

[16] Two direct quotations in this respect are "Bedingung dafür wären genauere Analysen der Zusammenhänge zwischen der Komplexität der Inter-Applikations-Strukturen und der Agilität." (In English: "A condition for this would be more thorough analyses of the relationships between the complexity of inter-application structures and agility.") and "Benchmark-Studien scheinen hier der einzige Weg zu sein, um zu aussagekräftigen Steuergrößen zu kommen. Diese fehlen jedoch, vor allem, da das Thema der Steuergrößen für die Applikationsarchitektur noch nicht genügend weit etabliert ist." (In English: "Benchmark-based surveys seem to be the only way of realizing expressive controlling measures. However, such surveys are not available, as controlling measures for application architectures are not sufficiently common and established.")

[17] Also [Du05] points to a lack of such approaches: "It follows then, that the ability to synthesise solutions, analyse them and extract guiding metrics, is a highly desirable capability within enterprise architecture in practice. Recent survey's and the experience of the authors suggest that few organisations currently posses that capability."

inter-organizational business service networks. [YZL07] discusses algorithms for quality of service-aware service composition.

In addition, some attempts of practitioners not made public but sketched in the respective project documentation, likely in presentation slides, exist.

Less literature than about metrics themselves is available regarding theoretic foundations of application landscape metrics. This is contrary to software engineering or business administration, where works as outlined in Section 2.2.2 abound, and also literature regarding foundations or standards exists. Additionally, the existence of journals, e.g., the *Journal of Software Measurement*[18], points to a much more mature field, offering a higher number of more established approaches.

The interviews from the EAMVS confirm this finding. Among 19 organizations, whose practitioners were interviewed, nine use to some extent metrics in managing their application landscape, and two have specific plans for using them in future. However, only two approaches widely known and available in literature were mentioned: *Function Points* and the *Balanced Scorecard*. It highlights the scarcity of approaches, that both originated from domains different from application landscape management.

Summary Both the scarcity of literature and the relatively low awareness level in practice regarding this literature point to a rather young field. Consequently, there are no established "theoretical problems" for research to work on, but ones directly originating from practice, where there is a need of well-founded approaches and cross-organizational discussions. Also, research might introduce more rigor by "borrowing" foundations from other domains.

3.1.2.2. Acceptance and Spread in Practice

The previous section indicated that the first approaches have been developed in the young field of application landscape metrics. This section evaluates to what extent the idea of using metrics on application landscapes is known and employed in practice.

Of the 19 different organizations, at which interviews have been conducted during the EAMVS, nine use metrics to some extent on their application landscape. Two more indicated specific plans for doing so in the future. Seven consider metrics for application landscapes as basically interesting; however they see unclear issues in this respect. Finally, three practitioners judged metrics for application landscapes to be irrelevant or unpractical. Two of these practitioners belonged to the same organization, while the third one was from an organization, that had another practitioner falling into the *basically interesting* category. The specific statements are summarized and categorized in Appendix A.1.2.

The practitioners of the organizations using or planning to use metrics for application landscapes indicated several kinds of metrics used or planned to use. The following list

[18] See `http://jsm.cs.uni-magdeburg.de`.

contains metrics used, if not stated otherwise. The numbers in brackets reference the interviews listed in Appendix A.1.2.[19]

- Lagging indicators, e.g., metrics characterizing past performance
  - IT efficiency vs. business value in function points, measured from past projects: one organization (4)
  - Errors in software development: one organization (7)
  - Measures for performance of the processes IT supports: one organization (8)
- Metrics characterizing usage and users of business applications
  - Number of users of a business application: three organizations (1, 2, 5)
  - Number of installations of a business application: one organization (1)
  - Usage frequencies of a business application: two organizations (2, 5)
  - Number of licenses: two organizations (1, 8)
- Leading indicators measuring structural aspects considered drivers for performance aspects
  - Degree of functional coverage of a business process: two organizations (2, 8)
  - Business application counts: two organizations (1, 6)
  - Function points: one organization (4)
  - Interface counts, coupling indicators: one organization (6)
  - Number of architecture violations: planned by one organization (11)
- Cost aspects: one organization (2)
- Value created by business applications: one organization (8)

The practitioners, mainly the ones not clear about or not favoring metrics usage, stated issues they see with the subject:

- Problems in getting the necessary information: four organizations (4, 10, 16, 21, 22)
- Business value of architecture metrics not directly visible; the analyses are difficult and expensive: three organizations (11, 12, 21, 22)
- Employees would feel controlled[20] by the metrics: one organization (14)
- No knowledge/ideas about what metrics to use: two organizations (15, 16)
- No need for application landscape metrics: one organization (20)

[19] Consider that some practitioners belong to the same organization, as described in Appendix A.1.2.
[20] In German: überwacht

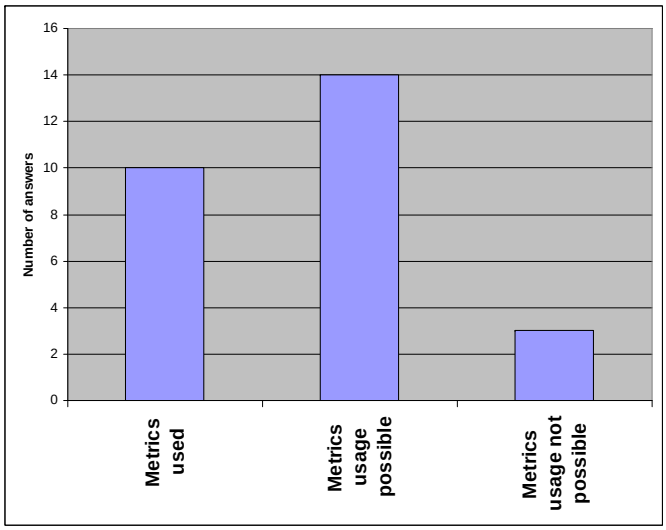


Figure 3.1.: Usage of metrics in managing an application landscape (n=27)

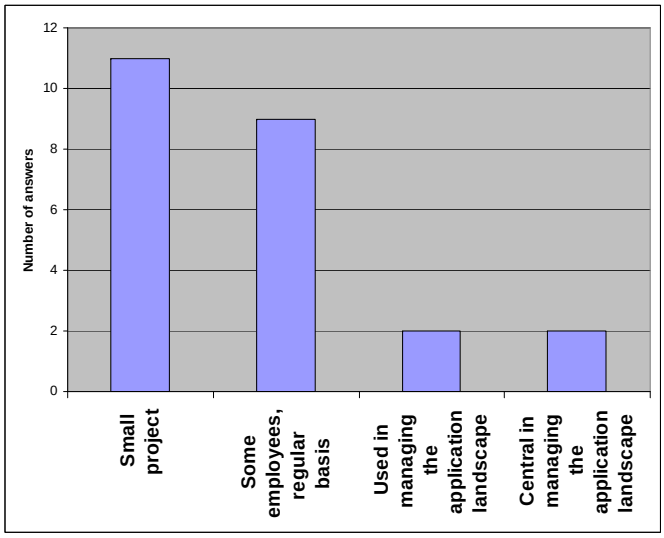


Figure 3.2.: Extent of metrics usage in managing an application landscape (n=24)

The online questionnaire basically confirms this picture. Answering question IIX 3.1 in Table 3.2, a large share of practitioners indicated they actually use metrics, while an even larger share saw them as a future possibility. Only a minority viewed metrics for application landscape as not sensible (see Figure 3.1).

Figure 3.2 shows most of the metrics usage in managing application landscapes as comparably low-key, while four practitioners indicated that they use metrics more extensively.

Thus, the questionnaire analysis also shows the field as rather young, with metrics not being ubiquitous and extensively used, but with potential, as a majority of the surveyed practitioners considered them as a future possibility.

Online questionnaire data have been analyzed more in-depth to reveal whether there are *environmental factors* characterizing environments more or less tending towards using metrics for application landscapes. Technically, this analysis is based on ordinal regression [FHT96], with detailed analysis reports available in Appendix A.2.2.1. The respective analyses have been performed using the *Ordinal Regression*[21] procedure from SPSS 14.

Firstly, personal characteristics of the interviewed practitioner have been tested, starting with the kinds of education received (question IX 2.1 in Table 3.2). Thus, as apparent from the analyses shown in Appendix A.2.2.1.1, an education in IT or in natural sciences seems to negatively affect the attitude towards metrics at a 0.05 significance level. At a higher significance level (0.15), an education in "Business" (e.g., Business Administration) seems to positively influence the attitude towards metrics for application landscapes.

When examining the influence of IT-, business- and EA management-knowledge (questions IX 2.2, 2.4, and 2.6), IT knowledge and business knowledge had influences at significance level $\alpha = 0.1$ (see the analyses shown in Appendix A.2.2.1.2): Growing IT knowledge was connected with less metrics usage, growing business knowledge with more metrics usage.

Testing the years of experience in business, IT, and EA management (questions IX 2.3, 2.5, and 2.7) (see the analyses shown in Appendix A.2.2.1.2), yielded a quite similar result; however, here only the positive influence of the years of business experience was significant at $\alpha = 0.05$.

After examining the personal characteristics, the influences from the organizational environment surveyed in the online questionnaire were tested. The education-specific variables which seemed most significant (education in IT, eduction in business, eduction in natural sciences) were always included in the model, in order to control their influence. The above discussion of personal influencing factors also point to including the IT knowledge and business knowledge, or the years of experience in business as alternative or additional factors. However, this is not done here. Adding these factors is considered overstressing the available data, even for an exploratory analysis. The dichotomous education variables are chosen, as they appear more significant (see Appendix A.2.2.1.1 and Appendix A.2.2.1.2), and do not rely on the assumption that a five-point scale variable is metric.

Concerning *data quality*, it was tested whether the attitude towards metrics for application landscapes was influenced by the number of data quality issues checked with question IIX 2.5. In this context, no significant influence could be detected (see Appendix A.2.2.1.3). However, it has to be noted that none of the surveyed practitioners indicated they were content with their data quality in answering question IIX 2.5.

The *size of the application landscape* (question IIX 1.4), aggregated to three size classes, as described in Section 3.1.1.2.2, appeared only as an influence on the attitude towards metrics at a significance level of 0.15, as demonstrated in Appendix A.2.2.1.4.

[21] In German: Ordinale Regression

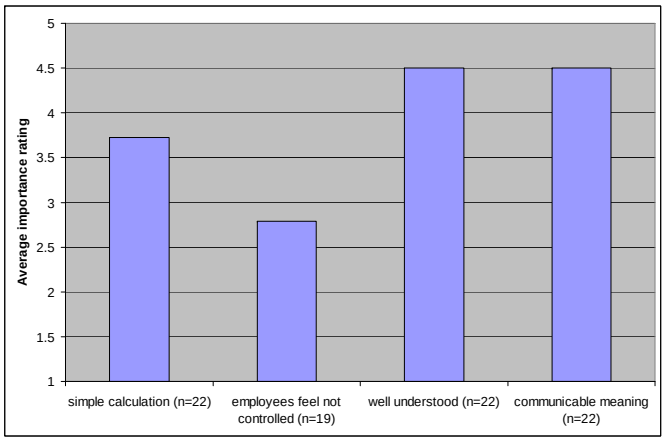


Figure 3.3.: Important properties of a good application landscape metric

The power and number of stakeholders (question IIX 2.9) did not show any significant influence on the practitioners' attitude towards metrics for application landscapes, as apparent from Appendix A.2.2.1.5. However, goal explication (question IIX 2.1) might be relevant (see Appendix A.2.2.1.6).

Significant at $\alpha = 0.15$ is the influence of the maturity level, as asked for by question IX 4.1, on the attitude towards metrics for application landscapes (see Appendix A.2.2.1.7). This leaves three organizational influence factors, application landscape size, maturity, and possibly goal explication, for which a positive influence on the attitude towards metrics (as operationalized by question IIX 3.1) might be suspected. As apparent from Appendix A.2.2.1.8, a correlation between these factors cannot be suspected based on the questionnaire data.

Finally, Figure 3.3 shows which properties (the surveyed) practitioners consider important for application landscape metrics. The diagram clearly points out two: The metric should be well understood, and its meaning should be easily communicable. Calculating the metrics, however, does not need to be as simple. This is contrary to the often used simple counts, which have only a vague connection to a quality attribute under consideration. Interestingly, the often heard aspects of employees feeling controlled by metrics appears least relevant here.

Summary Metrics for application landscapes are already used in a large share of organizations, however, in many cases only to a small extent. The potential for larger usage, especially when the field gets more developed, thus removing unclear issues, is likely to exist.

Differing kinds of metrics are currently used by practitioners. Metrics measuring aspects of the application landscape structure are, in most cases, simple counts, e.g., of business applications, or interfaces, which is however not without problems, as argued above (see Figure 3.3). Many metrics currently used on an application landscape are lagging indicators, measuring the past performance of the application landscape, thus, tying

such performance metrics to measures of the application landscape considered drivers, and thus leading indicators appears as an interesting use case.

Analyzing influencing factors on the practitioners' attitude towards metrics, three organizational influences appeared possible: the size of the application landscape, the extent to which goals are explicated, and the maturity of IT usage. Personal characteristics of the practitioners appeared as additional influences. Practitioners with a business education seem to be most inclined towards using metrics in application landscape management. This is a finding to be considered in creating metrics, especially when taking into consideration that practitioners view it as important that the meaning of metrics should be easily communicable and well understood. It presents metrics well understood and communicable ("sound and simple") also in business terms as an approach to integrate business more into EA management.

3.1.2.3. Usage Scenarios: How Metrics Are Employed in Practice

Above section examined, to which extent metrics for application landscapes are used in practice. This section explores *how* they are used.

The description of metrics usage in software engineering and business administration (Section 2.2.2) revealed many different usage scenarios, e.g., quality prediction regarding multiple quality attributes, guiding review processes in software engineering, or the numerous use cases in business administration, e.g., the balanced scorecard. Now, the following questions about the usage scenarios of metrics for application landscapes are explored:

- What *quality attributes* are relevant in managing an application landscape, and for what *evaluation objects* should metrics be able to assess them?
- In what specific *usage context* are metrics employed, and what *goals* do practitioners have in using metrics?

3.1.2.3.1. Quality Attributes and Evaluation Objects

Questions IIX 3.4.1-3.4.9 in Table 3.2 are based on a set of quality attributes developed and refined together with practitioners from software development of a financial service company in an application landscape-related project over multiple sessions. While Table 3.2 presents a short description for each quality attribute, the attributes are here presented in detail:

Maintainability Refers to the technical development and the maintenance efforts that arise in this activity. These efforts should be kept at a reasonable level. Possible sources for maintenance efforts are, e.g., errors or new versions of a software product already in use.

- Errors: Only a small number of failures should occur, only a small number of errors should be included in the system. Errors should be easily correctable.

- Releases: Adaptations to new software releases (e.g., of operating systems) should be possible at reasonable costs.
- Requirements: Efforts concerning new or changed requirements, affecting the functionality offered by the application landscape, are not relevant here.

Flexibility Changes concerning functional requirements (customer and/or market-driven changes) should be executed quickly and at reasonable costs.

- Parameterization: The mode of operation of the systems in the application landscape can be adapted with reasonable development costs by options, which have already been incorporated during development. This only affects changes, which were already explicitly anticipated during the development of the respective system.
- Modifiability: Functional requirements are changed and/or extended during the further development of the systems of the application landscape.
- Portability of (customer) interfaces: It should be possible to offer the interfaces (e.g., GUIs) on other platforms at reasonable effort.

Regarding changed and/or additional functional requirements, three aspects are of relevance: implementation at reasonable costs, time to market, and the effort occurring due to changes at the customer interface (backward compatibility). Another point to be considered is the risk that functionality desirable from a business point of view cannot be realized with justifiable effort within the existing application landscape.

Testability Tests with predefined test coverage should be realizable with reasonable effort. In this context, the functional coverage is important as well as the size of the test installation.

- The functional coverage of the test determines the tested amount of functionality, e.g., how many test cases of which size are executed?
- The size of the test installation is influenced by, e.g., the number of systems and interfaces taking part in the test, whether batch jobs are part of the test, etc. To minimize the effort, the tests should include as few systems as possible.

Performance The systems of the application landscape must be able to execute transactions respecting certain SLAs, e.g., time restrictions.

Scalability Adaptability to new requirements regarding quantity/capacity/time: The application landscape has to support the distribution of load on additional hardware in order to achieve goals such as performance, capacity (e.g. number of transactions), and quantities (e.g., number of users).

Operational risk The operational risk arising from the application landscape should be kept small. Operational risk refers to failures in the functionalities offered by the application landscape. In this context, both the failure probability (the availability) and the possible damage are relevant, as well as possibilities for risk mitigation.

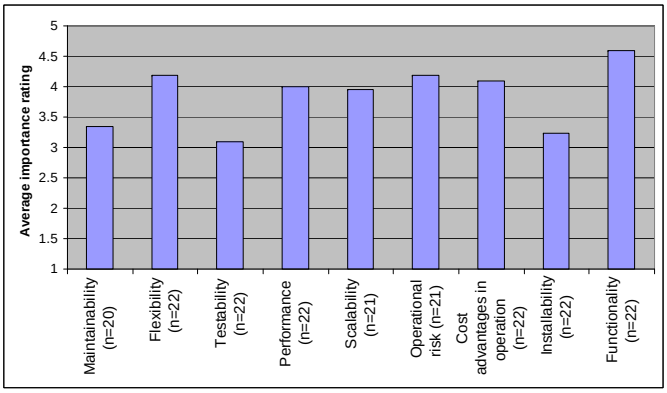


Figure 3.4.: Importance rating of quality attributes

Cost advantages in operation The technical operation of the application landscape has to be realized at reasonable costs. This quality attribute is concerned only with the current operation, without considering any changes. Also, the functional operation of the application landscape should be realized at reasonable effort, which involves particularly minimizing manual processing.

Installability Which efforts and what kind of restrictions have to be considered when installing a new version of a system? Apart from the effort, also the time, in which the installation procedure can be accomplished, is relevant as a temporal restriction. Hereby, it should be considered that the installation cannot be parallelized arbitrarily.

Functionality To what extent is the business supported in its activities and the execution of its processes by the functionalities of the application landscape? This basically covers two aspects: which functionality is available and how the functionality is provided.

- The aspect of available functionality is concerned with the fact which business activities are adequately supported by IT and which kind of activities have to be executed manually or are even not executed due to missing IT support. Provided functionality on the one hand means functionality, which is offered via a GUI, and on the other hand includes externally accessible APIs.
- How functionality is made available considers to what extent the architecture of the application landscape contributes to high accessibility of business application functionalities to users.

The practitioners surveyed by the online questionnaire rated the quality attributes as shown in Figure 3.4. *Functionality* has been rated as most important. *Flexibility*, *Operational risk*, and *Cost advantages in operation* also received rather high ratings, although less than *Functionality*. *Maintainability*, *Testability*, and *Installability* received the lowest ratings. A possible explanation consistent with the finding of Section 3.1.2.2, that metrics for application landscapes should have a meaning to business is that these three quality attributes are too technical and lack direct business relevance.

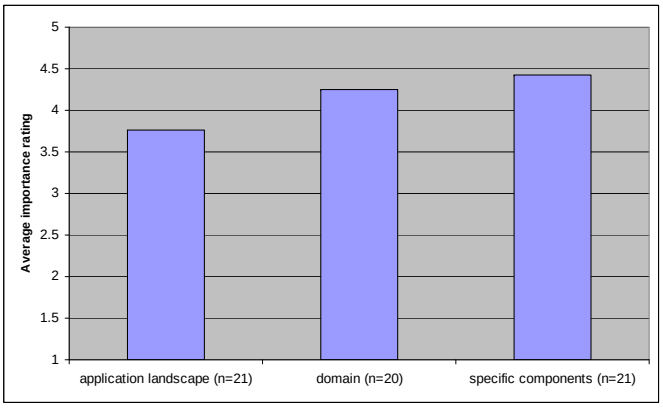


Figure 3.5.: Relevance of different granularity levels for application landscape metrics

Looking into the quality attributes the practitioners mentioned as relevant about a good application landscape in the EAMVS interviews basically revealed similar characteristics. The numbers in brackets reference the interviews listed in Appendix A.1.2:

- Flexibility aspects (easy and swift development of new functionality, simplicity of changes): mentioned by practitioners from three organizations (1, 4, 11)
- Scalability: mentioned by one practitioner (11)
- Business continuity aspects (risks posed by business application failures, recovery aspects): mentioned by one practitioner (1)
- Functionality, extent of support to business: mentioned by practitioners from three organizations (1, 2, 19)

In addition to these aspects, practitioners often answered this question by mentioning characteristics of the technical realization of the application landscape, which are usually considered drivers of qualities as mentioned above:

- Redundancy: mentioned by practitioners from three organizations (2, 18, 9)
- Homogeneity: mentioned by practitioners from three organizations (1, 6, 15)
- Dependencies: mentioned by one practitioner (5)
- Complexity: mentioned by one practitioner (11)

Quality is always quality of something, more specifically of an evaluation object. Based on questions IIX 3.5.1 to 3.5.3 from Table 3.2, Figure 3.5 shows at which level of detail the practitioners want to apply metrics to their application landscape. Metrics focusing on specific components of the application landscape, e.g., processes, business applications, or services, seem more important than metrics at the more coarse-grained level, especially ones calculating one value for the application landscape as a whole.

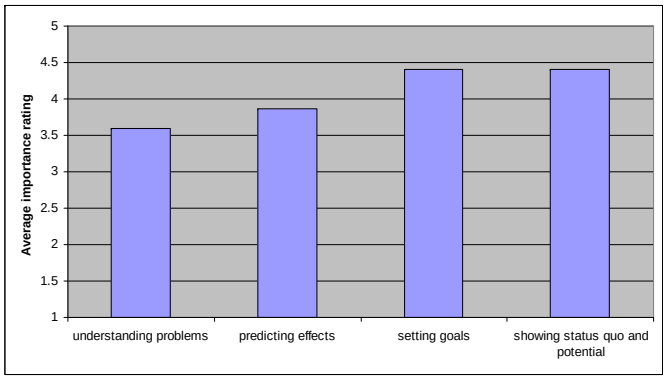


Figure 3.6.: Relevance of different kinds of usage scenarios for metrics (n=22)

However, looking at the questionnaire data underlying Figure 3.5 in detail reveals a more differentiated situation. Appendix A.2.2.2 shows results from three regression models, which try to link the three relevance ratings for the different granularity classes to the potential role of the answering practitioner (question IIX 3.3) and the size of the application landscape (question IIX 1.4). These analyses indicate that the importance of metrics at the application landscape level grows with the size of the application landscape targeted by the measurements ($\alpha = 0.1$). A possible explanation is that with growing size of the application landscape, keeping an overview gets more and more difficult, making aiding metrics more relevant. This highlights the importance of designing approaches for adequate aggregation of metrics[22]. A significant influence of the role taken by the practitioner on the preference with respect to metrics granularity could not be detected.

3.1.2.3.2. Usage Contexts and Goals of Metrics Usage

Questions IIX 3.6.1-3.6.4 in Table 3.2 surveyed how relevant different kinds of usage scenarios for application landscape metrics appear to practitioners. Figure 3.6 summarizes the results.

Interview statements about usage scenarios, here presented along the categories of Figure 3.6, are basically similar, but draw a more differentiated picture. Again, the numbers in brackets reference the interviews listed in Appendix A.1.2:

- Understanding problems:
  - Rethinking issues, e.g., licensing and focus on strategic business applications (1)
  - Identifying needs for action (3)
- Predicting effects: not mentioned in the interviews

[22]The case study in Chapter 5 also confirms this.

- Setting goals:
  - Managed evolution, i.e. tracking and controlling evolution with respect to certain qualities (4)
  - Planning/controlling (6, 8)
  - Performance metrics are tied to salaries (7)
  - Thoughts about integrating application landscape metrics into the IT-balanced scorecard already in use exist (16)
- Showing status quo and potential:
  - Rethinking issues, e.g., licensing and focus on strategic business applications (1)
  - Giving more detailed information about process support, summarizing information from detailed process diagrams (2)
  - Setting targets, checking whether they are met, and in case they are not met, exploring the reasons (8, 9)
  - Transparency, i.e. gaining and keeping overview (11)

Regression models used for exploring influences of the practitioners' role (question IIX 3.3) and the maturity of IT usage (question IX 4.1) on the relevance of the different kinds of usage scenarios (see Appendix A.2.2.3.1) revealed no significant influences.

Regression models for linking the practitioners role (question IIX 3.3) and the size of the application landscape (question IIX 1.4) to the relevance of the different kinds of usage scenarios (see Appendix A.2.2.3.2) hint to an influence of the size of the application landscape on "understanding problems". This may be explained by assuming that metrics get more important in understanding a problem, if it becomes more difficult to have a complete overview of the application landscape.

In each usage scenario, metrics can be employed to aid in addressing different problems. Figure 3.7 shows the questionnaire results from questions IIX 3.7.1-3.7.4 in Table 3.2, which were focused on the problems practitioners want to address using metrics.

In this context, communication appears as a rather important aspect, similar to Figure 3.3 in Section 3.1.2.2.

3.1.2.3.3. Summary

The above analysis of usage scenarios for application landscape metrics shows that largely differing usage scenarios for metrics are possible and also employed in practice. However, some findings from above might be helpful in identifying the adequate approach when developing a metrics-based methodology.

Regarding *what* to measure, qualities relevant to business, as *Functionality*, *Risk*, or *Flexibility*, seem to be more interesting to be supported by metrics than aspects more

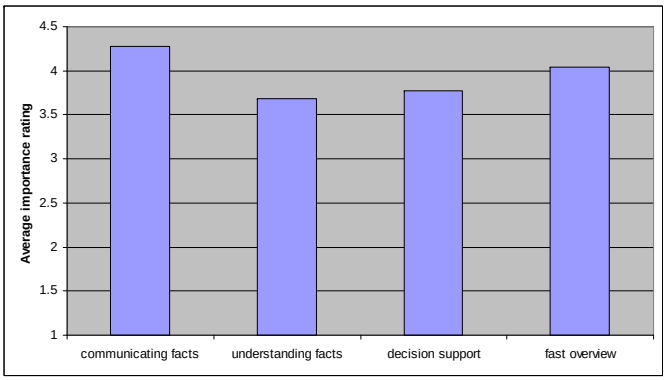


Figure 3.7.: Problems practitioners want to address with metrics (n=22)

rooted in technology, such as *Testability* or *Installability*. Measurement at the level of specific application landscape elements (business applications, services, etc.) is likely to be relevant, but especially with growing landscape size, more aggregated views become relevant, too.

In light of the findings above, one should especially consider using metrics for communicating facts and findings. This points to the relevance of having adequate metrics visualizations, and reinforces the need for metrics with a meaning easily communicable, also to business. Concerning approaches centered on metrics as predictors of certain effects or qualities, the above findings advise caution about whether this usage scenario really fits into the respective organization. On average, it was rated relatively low by the practitioners.

3.1.2.4. Conceptual Models/Metamodels

As outlined in Section 2.2.1, the process of measurement can be described as assigning numbers to objects of classes in a way to faithfully represent certain properties [Kr71]. Thus, measurement can be seen as based on a *conceptual model* of the respective domain, defining the classes and their relationships. These concepts determine *what* can be counted, summed up, etc. to derive a measure.

In the context of software engineering, this does not commonly appear problematic. In many areas, these conceptualizations have been readily available and strictly defined when the respective metrics have been built. Examples are control flow graphs, on which McCabes Cyclomatic Complexity [Mc76] relies, or the concepts of object orientation, on which, e.g., the MOOSE [CK94] and the MOOD [Ab95] metrics are based.

The same is true for several metrics in business administration. Accounting ratios, as mentioned in Section 2.2.2.2.1, can rely on a substantial body of regulations and guidelines in the field of accounting, which can also serve as definitions for concepts relevant to metrics in finance. Other fields in business administration can be seen as laying less stable foundations for the definition of metrics. When looking at a balanced scorecard,

which might use goals such as "customer satisfaction" or "employees per *", it becomes obvious that no exact and universally accepted definition of a customer[23] or an employee[24] exists.

This more resembles the situation regarding the management of application landscapes and the conceptual models used therein, which are subsequently called *information models*[25]. Unfortunately, no common information model for application landscapes exists, as described in [Bu07a]. Instead, EA management tool vendors, practitioners, and researchers create their respective models, with convergence towards accepted and established best practices only marginally visible today.

Defining such information models is aggravated by ambiguously used terms in this field. This is problematic for metrics definition, as it endangers the advantage of metrics being objective characterizations of the aspects under consideration: A metric is likely to conceal subjectivity by providing a perfectly objective measurement procedure, not taking into account subjective interpretations or ambiguous definitions of the concepts in the underlying model.

Summary [Bu07a] presents a pattern-based approach for constructing information models, which is applicable also in the context of application landscape metrics. This approach deviates from the creation of an information model covering all relevant aspects of the application landscape. Instead, the pattern-based approach focuses specific patterns on specific problems, and provides the concepts necessary for a solution. Regarding metrics definition, this means that an information model pattern defining the concepts for specific metrics is provided. Such patterns can then be used in constructing comprehensive, organization-specific information models.

This can be seen as congruent with the two domains alluded to above. After all, while there are many well-defined metrics in the context of software engineering, there is no single conceptual model of this domain. Object-oriented metrics, for example, rely on the concepts introduced in the paradigm of object orientation. Failure modeling, in contrast, might be more concerned with redundant and non-redundant units and their failure probabilities.

The situation is similar in business administration. Accounting ratios rely on the balance sheet as an abstraction of an organization, however, abstracting from the field of human resources, which may be of fundamental importance to certain metrics used in a balance scorecard.

When providing precise definitions of the concepts used in the information model, which is necessary to enable the provision of objective data for calculating metrics, two points should be kept in mind. Firstly, the importance of a shared understanding of the concepts, e.g., of a "part-time employee". Secondly, that the understanding should be useful in the context of the metrics and methodologies, in which it is employed.

[23] Consider, e.g., how to count institutional customers vs. private customers.

[24] Consider, e.g., part-time employees or temporary employees.

[25] This is the terminology commonly found in the field of application landscapes and used by the software cartography project at the Chair for Informatics 19 (sebis) of the TU München [Wi07, Bu07a].

In order to minimize subjective influence on the models built according to an information model underlying a metrics definition, it is useful to incorporate constructs into the information model, of which it can be easily and objectively verified, whether they actually apply.

3.1.2.5. Metrics Validation

In software engineering, the validation of metrics has not been without some misunderstandings and misconceptions, as detailed in [FP97]. In order to avoid such problems, this work uses the following terms introduced by [FP97]:

Internal Validation/Theoretical Validation is meant to demonstrate that a metric characterizes the attribute the metric is meant to measure [EE00]. This can be seen as proving that the attribute can be measured on a certain level of measurement.

External Validation/Empirical Validation builds on the internal validation and tries to establish that the metric is useful in predicting an important characteristic (also called an external attribute by [FP97]). This is relevant as the real benefit of a metric, which is meant to serve as a leading indicator, is the information the metric provides about this external attribute, which can be a lagging indicator.

A common approach for empirical validation is based on the construction of a model explaining the relationship between the metric under consideration and the respective external attribute/lagging indicator. Statistical hypothesis testing is then a possibility to confirm or refute such a model. This approach is propagated by literature about software engineering metrics, e.g., in [FP97] or in the "laws and theories" from [ER03].

Statistical studies for confirming such models are always subject to empirical data, on which the specific statistical techniques can work. Thus, difficulties can arise in the field of application landscapes, where data collection is known to be problematic [LMW05b].

In software engineering, several kinds of data can be considered easily available. Metrics parsed from source code can be considered only a calculation run away, and if the source code is under version control, metrics values covering the past evolution of the software are also available. Depending on the maturity of the software development process, effort data, data about failures and errors, etc. might also be available. Regarding validation studies, datasets from exemplary software projects are publicly available [Cu05].

In business administration, the situation depends largely on the kind of information. Data collected for accounting can be regarded as readily available, and is even publicly available for a considerable share of larger organizations. Other data might be more difficult to obtain, e.g., certain market and customer data.

Several aspects describe the situation regarding data collection in application landscape management:

- As mentioned above, data collection constitutes an important problem in managing an application landscape. Neither is "parsing the application landscape" easily possible[26], nor is detailed information about application landscapes commonly collected in other processes and publicly available, as with accounting. The fact that none of the 27 practitioners answering the respective question (IIX 2.5) in the EAMVS online questionnaire indicated that they were satisfied with the quality of EA data highlights this situation.
- Section 3.1.2.3 pointed out that a considerable share of metrics currently in use in application landscape management are performance metrics, or lagging indicators. Such information is necessary for validation studies regarding proposed relationships between leading and lagging indicators, which can thus profit from availability of such data.
- Logging data is generally a source of information about an application landscape. Especially middleware systems, message brokers, or systems management software can collect quite a wealth of data in the operation of the application landscape. Examples include exchanged messages, possibly how business applications react, if message exchanges fail, response times, or performance and load data. Similar data are much less readily available with single applications in software engineering. When trying to collect data, e.g., about calls to specific functions, one can rely on profilers, which are, however, usually employed only during development, and not in operation.
- Growing usage of technologies connected to *Service Oriented Architecture (SOA)*, e.g., service repositories or process definitions explicated in workflow languages as BPEL4WS [An03], or XPDL [Wf05] may create a new source of data about an application landscape. This might, in a limited sense, offer an equivalent to "source code parsing".

Summary The above elaborations point to the importance of models explaining the relationship between a metric and an external attribute, between leading and lagging indicators.

Firstly, in the creation of such a model, assumptions regarding the conditions, under which the supposed relationship between the metric and the external attribute holds, become explicit. This enables reflection on how realistic these assumptions are, leading to refinement or refutation of the model, without conducting a statistical survey. Also, an underlying model should, according to [FP97], be present as a basis for a sound statistical validation.

Moreover, a useful description of a methodology employing a metric should encompass the assumptions, under which a metric is a valid predictor of the respective external

[26] Tools for automatically determining which business applications are deployed in an application landscape exist, e.g. the *Mercury Application Mapping* mentioned in [se05a], which is now offered as the *HP Discovery and Dependency Mapping software* [He08]. However, they are, to the experience of the author, not common in data collection for EA management.

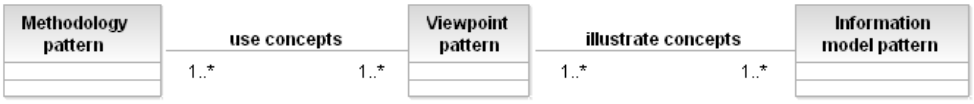


Figure 3.8.: Three kinds of EA management patterns [Bu07a]

attribute. The methodology has to make sure that the assumptions are true under the usage conditions. A model as described above simplifies this.

Also, theoretical validation should not be neglected when defining metrics and the methodologies using them. It should be verified whether a metric is consistently defined. Thus, issues in the metric can be found without an empirical validation, or without finding them via potentially costly errors in practical use.

3.2. Guidelines for Application Landscape Metrics

A metric in itself is only a means to pursue specific ends, and, if used in an engineering-like manner, a methodology describes how it can be employed to pursue such an end, addressing a concern in a planned way. In order to focus on the concern at hand, and how to address it, this work proposes to have the description of application landscape metrics connected to methodologies that employ them to address the respective concern.

Therefore, the work relies on the pattern-based approach mentioned in Section 3.1.2.4, detailed in [Bu07a], which uses the three kinds of patterns shown in Figure 3.8, which are defined as follows:

- An *information model pattern* contains the conceptual model providing both the classes, on which the metrics are defined, and the metrics characterizing objects of specific classes, e.g., as attributes.
- A *viewpoint pattern* describes, how the concepts and the metrics can be visualized.
- A *methodology pattern* is a description how to systematically address a concern, here focusing on how the metrics values are derived and how these values should be employed in addressing the respective concern; Section 3.1.2.2 already discussed the importance of having explicit goals when using metrics. Explicitly stated concerns are a means of explicating goals.

The metrics-related issues of these three concepts are explored in the subsequent sections, followed by Section 3.2.4, which closes the chapter by discussing an aspect especially important to metrics: models explaining the relationships between leading and lagging indicators, and the assumption under which these relationships exist.

3.2.1. Constructing Metrics-based Methodologies

As the status quo analysis about usage scenarios in Section 3.1.2.3 revealed, there are many different ways to benefit from metrics in methodologies for application landscape management. Five kinds of usage scenarios, derived from software engineering or business administration literature, or encountered in practice, are subsequently sketched.

3.2.1.1. Estimating and Tracking the Effect of Changes

Metrics can be used to track how specific qualities evolve over time, as the application landscape is constantly changed by possibly large numbers of projects. They can examine, whether certain projects conducted to add functionality to the application landscape endanger nonfunctional requirements, e.g., fault-proneness, or maintainability, making future functionality enhancements more expensive and cumbersome. This is along the usage scenario *showing status quo and potential*, which has been shown important to practitioners in Section 3.1.2.3.2. In this usage scenario, metrics might e.g. help in addressing the problem of getting a *fast overview* of a structure, of which Section 3.1.2.3.2 also highlighted the importance to practitioners.

Such tracking can be done using leading indicators, as in [LTC02], the example from software engineering already mentioned in Section 2.2.2.1.

Another possibility is using lagging indicators, which, while likely to measure the relevant quality attributes more exactly, can only reflect changes already made. An example from the field of application landscape management is the *Resistance to Change* [Mu05]. Using metrics in such a way can support the information roles taken by a manager (see Section 2.1.1).

3.2.1.2. Choosing an Alternative

Metrics can serve as a means of giving decision support regarding different alternatives, e.g., for realizing business support. Taking into consideration the importance and complexity of the application landscape alluded to in Section 1.1, such decisions should not be totally based on ad-hoc arbitrations being made in a state of insufficient information about their potential effects.

Metrics constitute an evaluation technique, which is able to systematically estimate the effect of changes (leading indicators), or evaluate them ex-post (lagging indicators). Metrics can present information in a way more suitable for supporting a sound decision, e.g., by preparing, enriching, or summarizing. This can be especially important, when, besides functional requirements, multiple nonfunctional requirements (e.g., a subset of the quality attributes described in Section 3.1.2.3) have to be considered. Thus, metrics are able to support the *Decisional Roles* of a manager alluded to in Section 2.1.1.

3.2.1.3. Depicting Scenarios and Communicating Their Benefits

Sections 3.1.2.2 and 3.1.2.3 indicated that metrics are suitable for facilitating communication, especially to stakeholders in business. This can exemplarily be highlighted by Figures 3.9 and 3.10. Figure 3.9 shows an exemplary project proposal with its technical details, trying to convey a message such as the following: "We are reducing the complexity of the application landscape by performing changes in business applications a, b, c, etc., which result in the dependencies highlighted in red being removed from the application landscape." However, details about specific dependencies might not be relevant to business, which might be more interested in what changes in the support the application landscape offers.

On the contrary, Figure 3.10 uses a metrics visualization to focus its message more on what is relevant to business: "By removing specific dependencies from the application landscape, the values of metric A are reduced by more than 20% at the business applications highlighted in the diagram. Metric A is a driver for the business applications' failure rate."

Figure 3.10 does not overwhelm stakeholders, especially from business, with technical details, of which these stakeholders might not even be able to see the implications. Instead, it relies on metrics as an understood technique to derive and directly show these implications. Thus, metrics are able to support the *Information Roles* of a manager as described in Section 2.1.1.

3.2.1.4. Setting Targets for Further Evolution

In setting targets for the evolution of the application landscape, metrics can be used in operationalizing goals and controlling goal achievement. Thus, metrics can support the management function called *controlling* or *monitoring and evaluating* in Section 2.1.1. Section 3.1.2.3 (see Figure 3.6) highlighted the importance of this kind of usage scenario.

Examples for communicating goals via visualizations are shown in Figures 3.11 and 3.12. Figure 3.11 visualizes a set of metrics values describing a specific evaluation object in a Kiviat diagram, then expressing the goal to bring the metrics to values located within the red circle in the diagram. Figure 3.12 visualizes a specific metric for a set of business applications, and demands that the metrics values are brought below a specific limit at the business applications highlighted by an exclamation mark.

By setting goals via metrics, one can abstain from prematurely prescribing a specific realization approach, but leave finding the best approach open to the respective experts. Such target values might even be included in the agreement on objectives for employees. In fact, this has been done with respect to certain performance metrics characterizing software development at one of the organizations covered by the EAMVS interviews (interview 7 in Appendix A.1.2). This usage scenario is realized in the balanced scorecard approach described here in Section 2.2.2.2.2, which might benefit from including application landscape metrics.

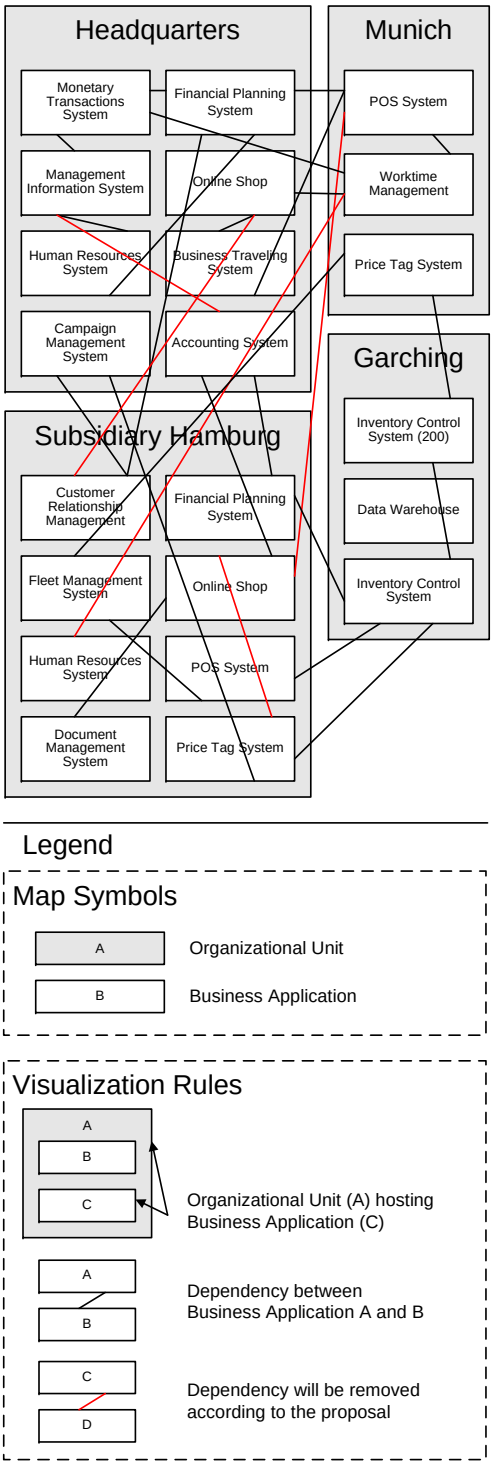


Figure 3.9.: Technical details of a project proposal

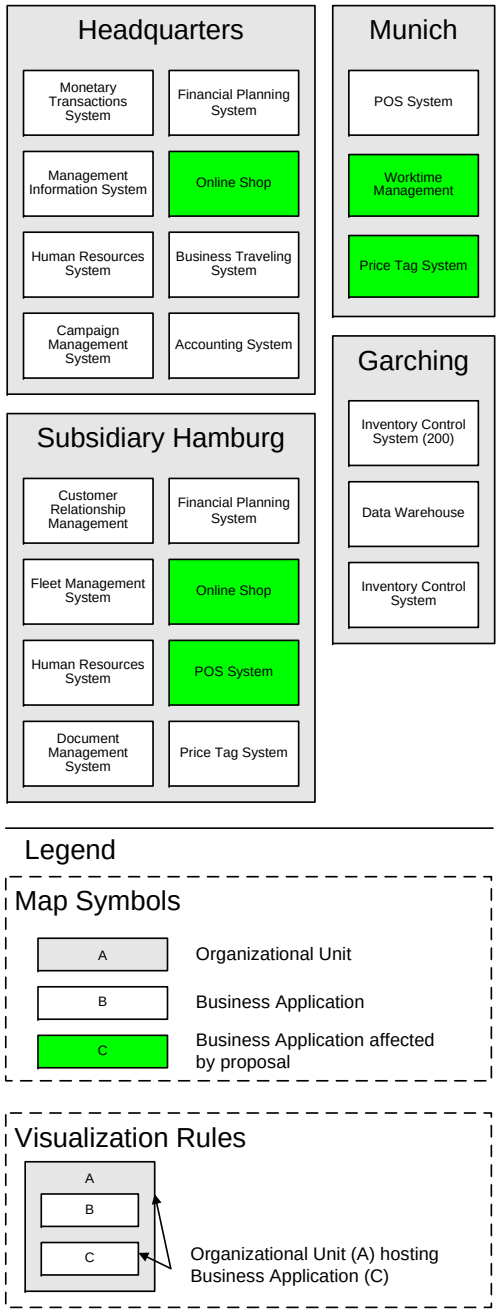


Figure 3.10.: Proposal effects highlighted by a metrics visualization

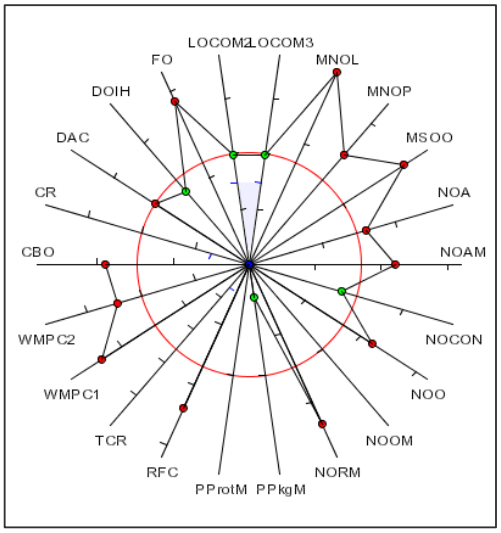


Figure 3.11.: "Move inside the red circle"

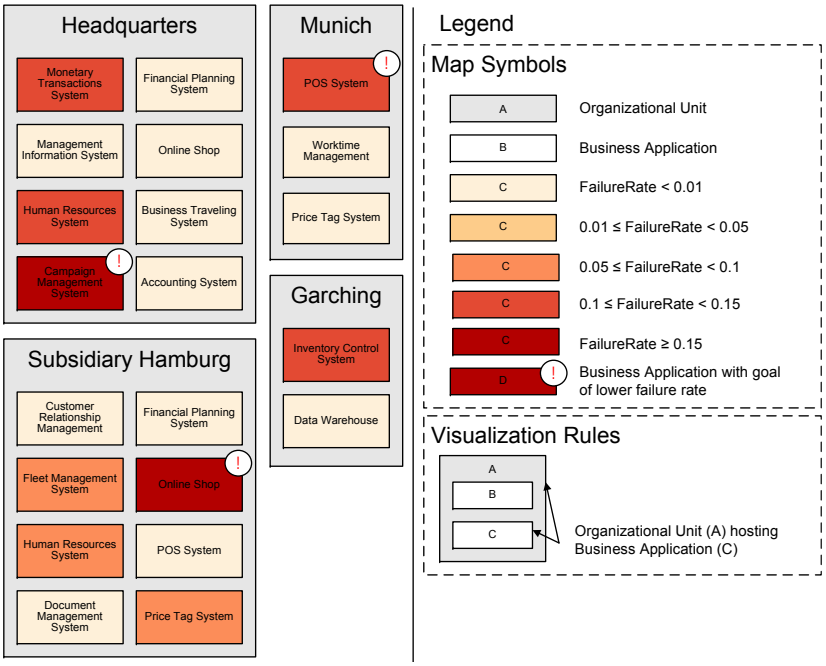


Figure 3.12.: "Reduce metrics values highlighted by exclamation mark"

3.2.1.5. Guiding Reviews

Section 2.2.2.1 mentioned [Ve05], an approach for supporting reviews in software engineering via metrics. Metrics may be used to detect areas of the application landscape where analyses determining potential for improvement can be expected to be especially fruitful. They may guide the effort of the experts conducting reviews to the problems and questions actually in need of expert reviews, and try to sort out other questions by a previous metrics analysis. Thus, this kind of usage scenario combines expert opinions and metrics as a formal evaluation technique.

The above described possibilities, from Section 3.2.1.1 to Section 3.2.1.5, should be taken into account when designing a metrics-based methodology, also considering that not every aspect of a methodology has to be addressed strictly by using a metric. "Not everything that counts can be counted, and not everything that can be counted counts", can serve as a proverbial summary here.

3.2.2. Visualizing Metrics

The above description of usage scenarios has already encompassed several visualizations. Section 3.1.2.3 highlighted the importance of using metrics to facilitate communication between stakeholders, and pointed to the role visualizations can take therein. This section gives an overview of different approaches for visualizing metrics, also supplying pointers to the respective literature. Its focus is on visualizing application landscape metrics on software maps as introduced in Section 2.1.3.2.

3.2.2.1. Metrics Visualization in General

Information visualization in general is treated, e.g., by Tufte [Tu01], [Tu90]. Tufte identifies general principles guiding design, editing, analysis, and critique of data representations. These guidelines are meant to explain design excellence, guiding to visualizations that are instruments of reasoning about information, and do not distort or cloud the respective facts. The covered visualizations range from pictograms over different kinds of charts and graphs to thematic maps.

Different kinds of diagrams, e.g., line charts, bar charts, Kiviat diagrams, or bubble charts, are important in metrics visualization. Portfolio matrices, similar to bubble charts, are frequently used in business administration.

[LKO97] constitutes an example of metrics visualization from software engineering. It uses, among others, bar charts, surface charts, and Kiviat diagrams to visualize the distribution of specific metrics values, checking for patterns, which point to the absence or presence of design problems. Several approaches for visualizing software metrics on representations of software systems exist. [LS02] describes an approach centered on 3-D visualizations of software. [LD03] visualizes metrics on graphs with nodes, e.g., representing specific classes, quite similar in appearance to UML class diagrams. [DL05]

proposes a so-called *class blueprint* for depicting the internal structure of a class, also visualizing metrics characterizing the elements making up the class.

3.2.2.2. Visualizing Metrics on Software Maps

Section 2.1.3.2 introduced software maps as visualizations of application landscapes. As elaborated on by [Wi07], software cartography is, in its approach to visualization, inspired by thematic cartography [HGM02, Sl05]. Thematic cartography uses a plethora of approaches for visualizing quantitative data describing specific geographic locations (see besides the above-mentioned publications on cartography, e.g., [ES90], [ES01], or [Be08] for examples).

[Wi07] touched on the subject of visualizing metrics on software maps. [Be04] explored it more in-depth, mainly in the context of metrics from the *IT Infrastructure Library* [OGC00a, OGC00b] and the balanced scorecard approach. Below, some basic approaches to visualizing metrics values on a software map are shown, presented with exemplary software maps and complemented by examples from cartography. Cartography literature contains more advanced examples and is thus suitable to provide inspirations.

3.2.2.2.1. Mapping Metrics to Color

A metric value can be used to determine the color of a graphical object representing the actual object the metric value describes. Figure 3.13 is an example from cartography, using this approach to visualize the average annual loss caused by different kinds of natural disasters (e.g., hurricanes, earthquakes) [ES01].

When using this approach to color objects on a software map to represent metrics, two basic decisions have to be made. First of all, the range of colors to be used has to be determined. [Ro95] and [Sl05], for example, present various aesthetic aspects to be considered in choosing a color scheme for an intuitive visualization. [Br08] presents a tool providing different color schemes and aiding in selecting an appropriate one. The tool is also available as a Java library. Then, the assignments between metrics values and the respective colors have to be made. This assignment may not necessarily be a linear one. Figure 3.12 contains a software map using this approach.

3.2.2.2.2. Mapping Metrics to Symbol Size

Metrics may be visualized by using them to change the sizes of certain objects on the map. In cartography, this is known as a *cartogram*[27], of which Figure 3.14 provides an example, in which the size of each territory shows the relative proportion of the world's population living there.

When sizing objects on a software map according to metrics values, the values can be scaled before they are used to scale the dimensions of the objects. In this context,

[27] In German: Kartenanamorphote

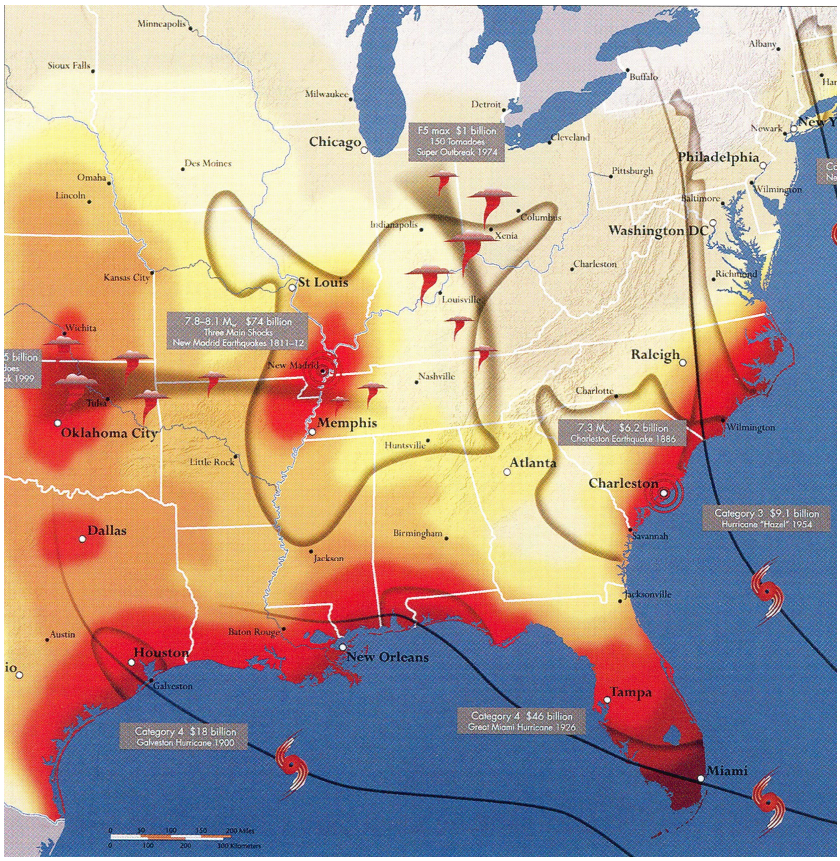


Figure 3.13.: Loss caused by natural disasters [ES01]

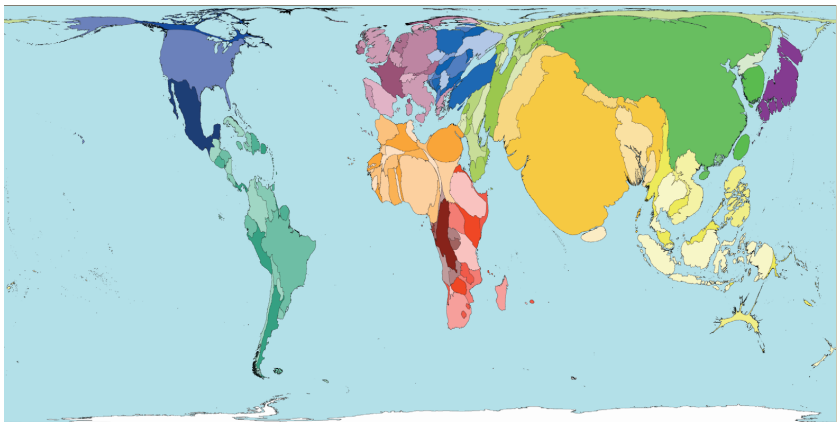


Figure 3.14.: Cartogram displaying populations on territories [Do08]

however, a basic rule should be taken into consideration: If it makes sense to add metrics values, it should make sense *adding up* the sizes of the respective symbols, too. A symbol depicting a metric value x, should have the same size as a set of symbols, for which the

metric values add up to x, together. More formally:
$\sum_{o_i \in objects} metric(o_i) = metric(o) \iff size(symbol(o)) = \sum_{o_i \in objects} size(symbol(o_i))$

The height and width of a symbol representing object o could thus be scaled with $\sqrt{metric(o)}$. Figure 3.15 shows a software map exemplifying this.

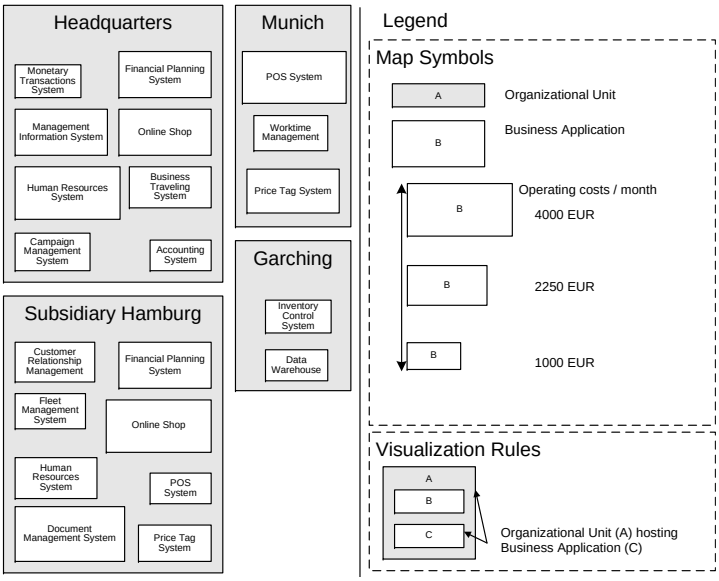


Figure 3.15.: Visualizing operating costs by symbol size

3.2.2.2.3. Mapping Metrics to Symbol Distance

In cartography, a *distance cartogram* [St06] is a map, which scales distances between certain points on the map to represent certain metrics. Figures 3.16 and 3.17 exemplify this for travel times on the London Underground network.

Figure 3.16.: London Underground: actual distances [Ca08]

Figure 3.17.: London Underground: distances indicate travel time [Ca08]

An exemplary software map using this approach is shown in Figure 3.18. This map indicates, via pictograms, what programming languages the different business applications

are written in. The distances from the center indicate a metric measuring the difference of used languages to a corporate standard, which demands business applications relying only on Java 1.4.

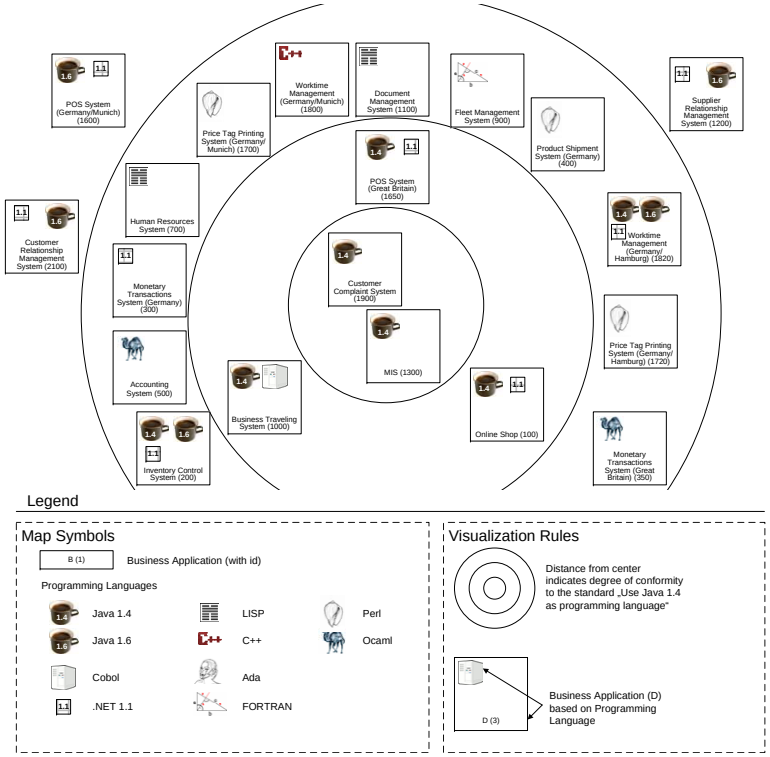


Figure 3.18.: Conformity to a programming language specific corporate standard, visualized by distance to center

3.2.2.2.4. Overlaying Additional Symbols

Placing symbols, e.g., bar charts or pie charts, on a map to visualize metrics characterizing a specific location is often used in thematic cartography. The approach also translates easily to software cartography, as shown by the exemplary map in Figure 3.19.

3.2.2.2.5. Outlook: Interactive Visualizations

Interactive visualizations can extend the insights into complicated data spaces, by allowing a user to select and explore specific aspects. A basic means of adding interactivity is the layering principle described in Section 2.1.3.2, with the possibility of showing or hiding different layers, focusing on a set of selected aspects at a time. Two examples of interactive software maps, which involve interacting with the symbols on the maps themselves, are shown in Figures 3.20 and 3.21. More sophisticated examples of interactive software maps could use clicking on the map to start more in-depth metrics calculations about specific aspects.

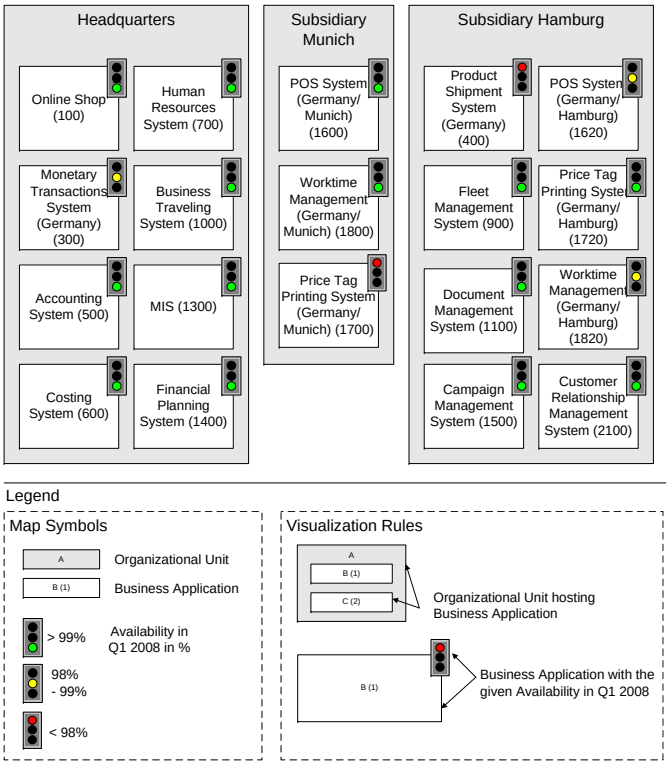


Figure 3.19.: Availability of business applications, visualized by a traffic lights symbol

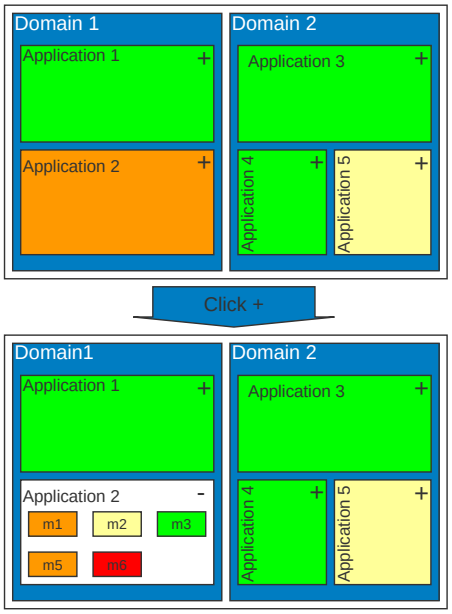


Figure 3.20.: Clicking on a domain reveals details

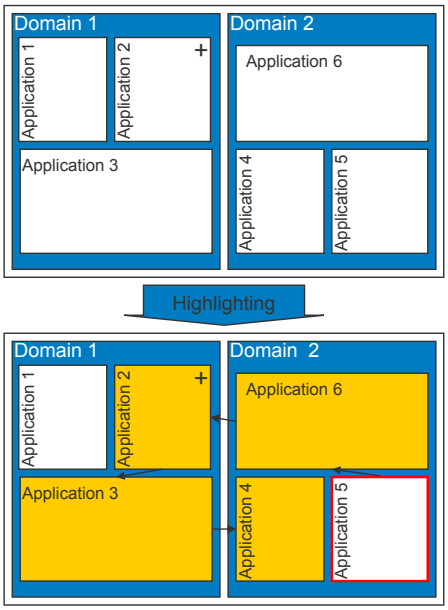


Figure 3.21.: Selecting applications to highlight successors

3.2.3. Information Models for Metrics-based Methodologies

To serve as a solid building block of a methodology, a metric should be defined in an unambiguous way. A metric thus defined can be subjected to internal validation, as outlined in Section 3.1.2.5. This ensures that the metric is consistently defined. Ruling out, or being able to control unsuitable behavior, is essential to avoiding systematic measurement errors, potentially flawing the methodology using the respective metric.

Furthermore, knowledge of the level of measurement enables efficient use of a metric in a methodology. It establishes which interpretation of the respective metric values are meaningful, and which are not [FHT96, Fa99].

The basis for a metric definition as described above is a suitable conceptual model or information model. As outlined in Section 3.1.2.4, this work proposes the conceptual model, on which the metric definition is built, to be based on an information model pattern.

Section 3.1.2.4 also describes the benefits of objectively enforceable modeling guidelines in information models. When, e.g., modeling business objects used by business applications, it is to a certain degree a subjective decision, at which granularity such objects are modeled. However, when considering a deployed business application, and a set of infrastructure services, it can be seen as rather objectively verifiable on which of the services the deployed business application depends.

In this work, metrics are modeled as attributes of the respective classes in the information model. Formal metric definitions are either specified in OCL [OM06b], or using notations common in mathematics, if the complexity of the definition requires this. To advance understandability of the metrics, both kinds of definitions are complemented by textual explanations.

3.2.4. Making Assumptions about Metrics Explicit

Underlying a metrics-based methodology are assumptions about how a specific metric connects to a specific concern. While different kinds of assumptions can be involved therein, the link between leading and lagging indicators, as described in Section 3.1.2.5, is often essential in metrics-based methodologies.

The models or theories giving this link should be explicated, if not in the description of the methodology itself, then in another document where they can be referenced. Such models should explicitly state under which conditions they are applicable. It then clearly has to be ensured by the methodology that these usage conditions are met when the methodology is applied. A core problem hereby is tied to the *objectively enforceable modeling guidelines* mentioned in Section 3.1.2.4. Due to insufficient objectively enforceable modeling guidelines, it may be possible that some precautions have to be taken to know with a specific degree of certainty that the application landscape is modeled in a way leading to usable results for the respective metric.

Data can be constantly kept in a state suitable for metrics analysis by appropriate update workflows, or checked before metrics are applied. If a metric is used to make predictions about a lagging indicator, the prediction accuracy can be checked ex post, if the respective values of this indicator are available. This can be done by adequate visualizations, contrasting the lagging and the leading indicator, or by statistical models, e.g., regression.

CHAPTER 4

Metrics-based Methodologies Addressing Availability and Failure Coupling

Contents

Chapters 1 to 3 introduced management of application landscapes, and described how metrics can be applied to it. This chapter presents specific examples of these issues by introducing two metrics-based methodologies for analyzing and limiting failure propagation in an application landscape. As proposed in Section 3.2, the methodologies are presented as EA management patterns.

Section 4.1 introduces the concerns to be addressed by the methodologies. Section 4.2 lays the groundwork for their introduction by presenting related work and the basic concepts the methodologies rely on. These concepts, including the metrics, are described in detail in Section 4.3, where the information model patterns the methodology patterns rely on are presented. Viewpoint patterns visualizing the concepts from the information model patterns are then presented in Section 4.4, while Section 4.5 finally introduces the methodology patterns themselves.

4.1. Concerns

With the basic structure of an application landscape a complex web of interdependent business applications, which exchange a plethora of information objects over a multitude of interfaces, it is not surprising that the failure of such an interconnected business application is likely to cause other applications not to perform correctly or not to perform at all. This behavior, called failure propagation, influences how the application landscape can support business processes, and therefore the business services provided by the enterprise itself.

Basically, failure propagation can be examined from two perspectives: On the one hand, one can look at a specific interface and ask to what extent its availability is negatively affected by failure propagation. On the other hand, one can look at a failing element of an application landscape, e.g., a business application, and examine the effects of such a failure on services rendered by the application landscape.

On a technical level, failure propagation results in lower availability of interfaces. However, decreased availabilities are likely to result in costs due to deteriorated process support, which is the underlying reason for organizations analyzing and trying to limit failure propagation.

Two concerns summarize above described aspects of failure propagation, of which one covers analyzing, the other one actually changing the application landscape.[1]

C-110: Analyzing Failure Propagation in an Application Landscape:

- What are the availabilities of interfaces via which the business applications in the application landscape offer specific services? How does failure propagation affect these availabilities?
- How is business affected by interfaces that fail due to propagated failures? What costs are induced by failing interfaces?
- How large is the impact of a specific business application failing? To what extent are interfaces offered by other business applications affected via failure propagation?

[1] Concerns, methodology patterns, information model patterns, and viewpoint patterns are in this work numbered to fit into the EA Management Pattern Catalog [Bu08a]. Thus, the numberings do not start at one.

- How is business affected by a failure propagating through the application landscape? What costs are induced by these failures? *Business Continuity Management*, as described by ITIL [OGC00a], is concerned with implementing measures enabling business to operate also in case of unforeseen large damages. Measuring the impact a business application failing has on business can help to identify relevant risks arising from the application landscape.

C-111: Deciding on Measures for Limiting Failure Propagation in an Application Landscape:

- How do changes to the application landscape limit failure propagation? To what extent can a project implementing specific changes increase the availability of interfaces and limit the impact of a business application failing?
- What is the value such improvements constitute to business, and are they worth the project effort for implementing them?
- How do different proposals compare with respect to limiting failure propagation and implementation effort? Which proposal is able to yield a specific benefit with the least implementation effort?

These concerns belong to the quality attribute *operational risk*, which has been described and found as rather relevant to practitioners in Section 3.1.2.3.

4.2. Related Work and Basic Concepts for Modeling Failure Propagation Structures

4.2.1. Related Work about Failure Propagation

Addressing the above concerns relies on modeling and evaluating failure propagation structures in an application landscape. Generally, various approaches for modeling and evaluating failure- and failure propagation-specific aspects exist. At the level of business processes, e.g., quality control is concerned with the subject, employing approaches as control charts, cause-and-effect diagrams, or QC process charts for process analysis [Is90]. Impact analyses, as commonly supported by EA management tools [se05a, Ma08], are able to determine sets of elements in an application landscape, which are, via specific kinds of relationships, including transitive ones, related to an element under consideration. More formal approaches to failure modeling, common in technical domains, include reliability block diagrams, failure trees, or Markov chains, which are able to model internal states of systems. These approaches are able to make quantitative statements about availability, failure times, etc. [Ge89, Wa03].

[GKC06] mention using interfaces between business applications in reliability analyses, however, not broadly discussing the subject and providing a quantitative approach.

Failure modeling has been used by [Ma05b] to optimize software architectures. However, the core goal of this article is to apply multi-objective design to automatically find optima

regarding specific architectural parameters. Failure modeling is only briefly introduced to provide an example of one optimization goal. Consequently, [Ma05b] does not elaborate on viewpoints and information models, instead relying on an ad-hoc defined box-and-line architectural description resembling reliability block diagrams. Specifics of application landscapes are not covered. The approach only mentions components and connectors, however, lacking, e.g., domains, interfaces, or business processes. Also, aspects of failure propagation besides availability, as impacts, failure costs, or risk of large failures are not discussed.

4.2.2. Basic Concepts to Address C-110 and C-111

The metrics-based approach developed here complements a model of the application landscape structure (see Figure 4.1) with application-internal models depending on one of the above techniques: failure trees (see Figure 4.2). While other approaches to failure modeling, e.g., the above-mentioned Markov chains, would allow more expressive models, they are not used here, as they would lead to too data-intensive models, as Chapter 5 is going to confirm.

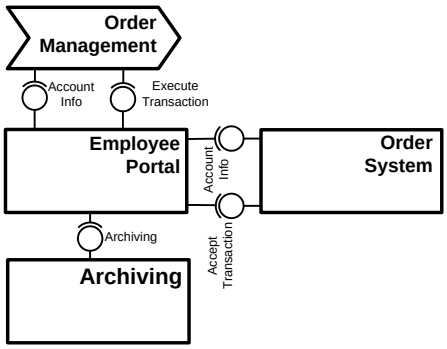


Figure 4.1.: Inter-application view

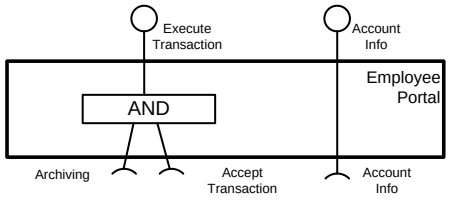


Figure 4.2.: Intra-application details

Information as exemplified by Figures 4.1 and 4.2 makes it possible to model which business applications need to be operational for a specific interface to be operational, i.e. able to render the services specified according to its definition. Metrics are then used to evaluate these structures, to characterize to which extent they are prone to failure propagation. Figure 4.3 exemplifies this.

Interfaces can be characterized by the probability that they fail due to business applications failing and subsequent failure propagation, as shown in Figure 4.3. *AccountInfo* and *AcceptTransaction*, together with *Archiving* fail only if the offering business application fails, and are thus colored green. Taking into account the internal structure of the *EmployeePortal* shown in Figure 4.2, it becomes obvious that *AccountInfo* of the *EmployeePortal* needs two business applications being operational, while *ExecuteTransaction* relies on all shown business applications, leading to yellow coloring for *AccountInfo* and red coloring for *ExecuteTransaction*.

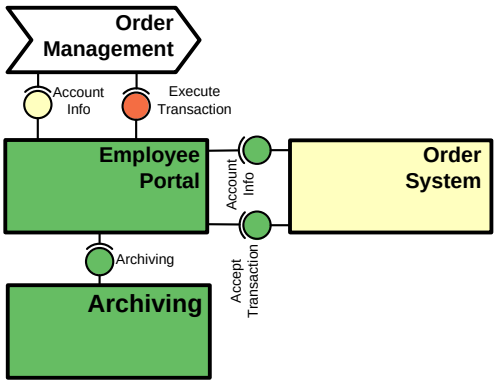


Figure 4.3.: Availability of services and impact of business applications

The business applications in Figure 4.3 are colored according to the impact they have on the interfaces in the application landscape. If the *Archiving*-application fails, its own *Archiving*-interface and the *ExecuteTransaction* interface of the *EmployeePortal* fail. If the *EmployeePortal* fails, the two interfaces it offers fail. Thus, these two business applications are colored green. If the *OrderSystem* fails, more interfaces are affected: *AccountInfo* of the *OrderSystem* and the *EmployeePortal*, *AcceptTransaction*, and *ExecuteTransaction*. Thus, it is colored yellow.

Such models, for which Section 4.3 provides the information model patterns and Section 4.4 the viewpoint patterns, are the foundation for two methodology patterns:

- M-40 addresses C-110, and utilizes metrics for *showing status quo and potential* (see Section 3.1.2.3), for increasing transparency with respect to failure propagation, its effect on availabilities and failure impacts. It is built around the usage scenario *estimating and tracking effect of changes* (see Section 3.2.1.1), but also includes aspects of analyzing the structures, which are made transparent in executing the methodology, then using metrics in *guiding reviews* (see Section 3.2.1.5).
- M-41 addresses C-111, by using metrics to compare different proposals for changing the application landscape with respect to change effort and effect on limiting failure propagation and improving availability. The methodology pattern uses metrics to aid in *depicting scenarios and communicating their benefits* (see Section 3.2.1.3), and to provide decision support in *choosing an alternative* (see Section 3.2.1.2). Especially *communicating facts* has been identified as a problem practitioners want to address using metrics in Section 3.1.2.3.2.

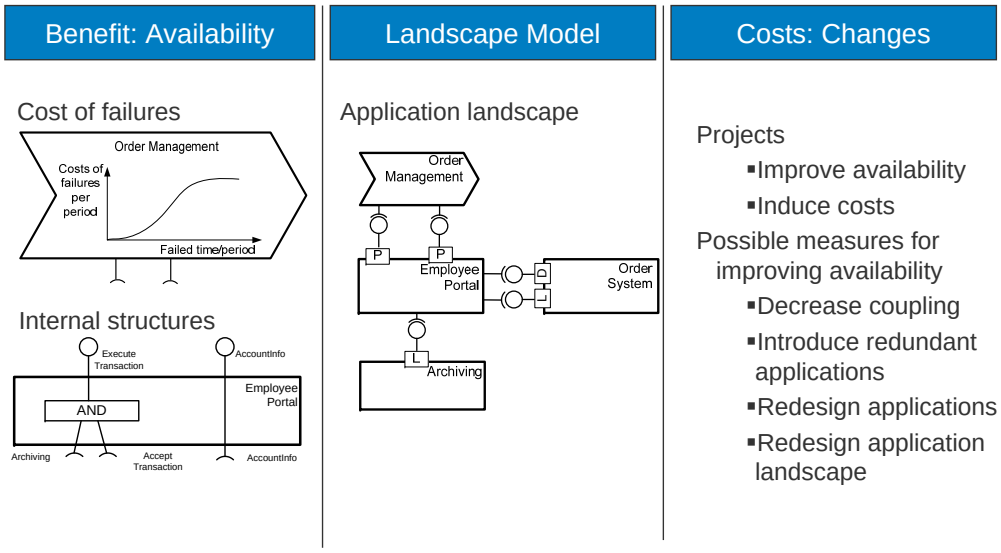


Figure 4.4.: Aspects the information model patterns should cover

4.3. Information Model Patterns and Metrics

In order to address the concerns introduced in Section 4.1, models are provided to cover three tiers, as shown in Figure 4.4.

- The middle part of Figure 4.4 shows the structure of the application landscape, taking up several of the application landscape elements introduced in Section 2.1.2. It revolves around interfaces offered via specific approaches (**P**rocess based coupling, **L**ogical coupling, **D**atabase, ...) by business applications and used by other business applications or business processes.
- In order to enable addressing the failure-propagation specific concerns C-110 and C-111, the middle part is complemented by details relevant to failure propagation (the left side in Figure 4.4): These are mainly concerned with the internal failure propagation structures of the business applications, and how failures of interfaces offered by business applications affect business processes. These models are able to capture availability-specific benefits of changes to the application landscape.
- C-111 encompasses changing the application landscape. The right side of Figure 4.4 covers aspects of changes, which on the one hand improve availability, and on the other hand induce costs.

For reasons of understandability, the respective information model patterns are introduced in two steps. Initially, an information model pattern focusing on essential, basic concepts is presented. Such an information model pattern might also be relevant to limit the data collection effort in a first attempt to introduce the respective models. Then, Section 4.3.2 introduces extensions to allow more realistic and expressive models of the application landscape.

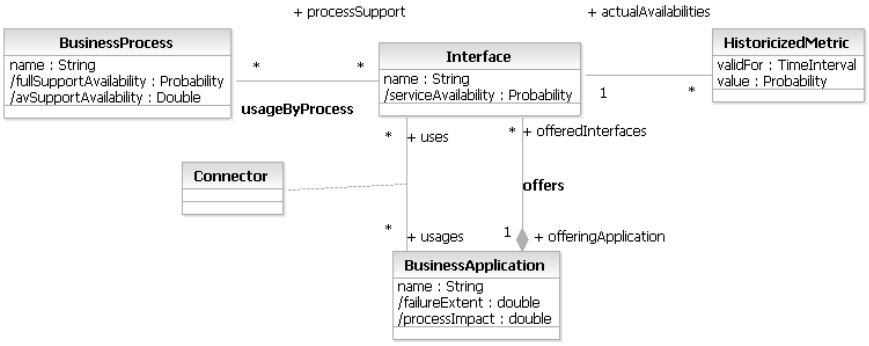


Figure 4.5.: Basic information model: application landscape structure

4.3.1. I-90: Basic Information Model

Information Model Pattern Overview	
Id	I-90
Name	Basic Failure Propagation Structures
Alias	A similar model has been proposed by [LS08]
Summary	This information model pattern provides basic structures for modeling failure propagation.
Version	1.0

As mentioned above, the basic information model defines concepts for modeling inter-application connections and process support, as shown in Figure 4.5 and described subsequently:

BusinessApplication A BusinessApplication in this context is a system, which is implemented in software, deployed at a specific location, and which provides support for or is specific to at least one business process. In performing the business support, a BusinessApplication may depend on other applications, which is modeled by two associations to the offered or used interfaces (offers and Connector, respectively). BusinessApplication denotes here an actual deployment of a software. It does not encompass social components (e.g., employees or stakeholders), or the technical infrastructure.

BusinessProcess A BusinessProcess is here understood as a sequence of individual functions with connections between them. A BusinessProcess as used in this information model should not be identified with a single process step or individual functions, but with high-level processes at a level similar to the one used in value chains. In executing a business process, a number of interfaces offered by business applications are used (see association usageByProcess), therefore effectively making the process execution dependent on the utilized BusinessApplications. Such

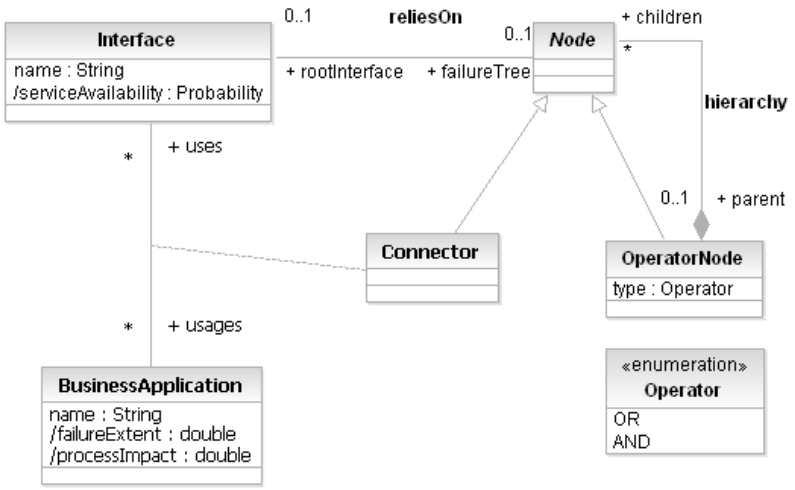


Figure 4.6.: Basic information model: application-internal details on failure propagation

a dependency is exemplified in Figure 4.3, indicating the interfaces used by the BusinessProcess "OrderManagement".

Interface An Interface is offered by a BusinessApplication to provide a service for external use by another BusinessApplication or a BusinessProcess (in which it could be a user interface employed in manual execution). An Interface can be connected to many using entities. This connection relationship is exemplarily visualized in Figure 4.1, showing Interfaces of BusinessApplications via the lollipop symbol. It resembles the *Logical Interface*[2] in Quasar Enterprise [En08]. In the Component-and-Connector viewtype of [Cl02], it would be called a *Port*.

Connector Signifies that a BusinessApplication uses an Interface offered by another BusinessApplication. It is similar to the *Connector* in the Component-and-Connector viewtype of [Cl02].

HistoricizedMetric An actual metric value that characterizes an evaluation object during a specific space in time (validFor). It is here used to capture actual availability measures for Interfaces made in the past. Availability is here defined as $\frac{Mean\ Time\ between\ Failures}{Mean\ Time\ between\ Failures\ +\ Mean\ Time\ to\ Repair}$ [Ge89], i.e. the share of time the respective Interface is operational.

As apparent from Figure 4.4, the above defined concepts are complemented by application-internal details about failure propagation, which mainly revolve around augmenting each Interface offered by a BusinessApplication with a structure indicating on which of the Interfaces used by the BusinessApplication it relies.

[2]In German: Logische Schnittstelle

Node A node represents the abstract basic element of a failure tree. It can be either a leaf node as represented by a Connector or an OperatorNode corresponding to a Boolean operation, as detailed below. A node can evaluate to different statuses regarding operation, i.e. operational and non-operational.

OperatorNode This node is connected to child nodes, which represent prerequisites for the OperatorNode to have an operational state. The node itself has an attribute referring a corresponding Boolean operation associated, determining whether all prerequisites (Operation AND) or at least one prerequisite (Operation OR) is necessary for the OperatorNode to be in an operational state.

Additionally, OCL rules for establishing the tree-like structure have to be incorporated:

```
context:  Node
derive:  ancestors=parent->union(parent.ancestors)
derive:  descendants=children->union(children.descendants->asSet())
inv:  descendants->excludes(self)
inv:  ancestors->excludes(self)
```

In the failure trees, only AND and OR are allowed as operators. XOR and NOT are not included, as these are considered rather irrelevant when modeling an application landscape, and the restriction makes evaluating the models more straightforward.

4.3.1.1. Metrics

The HistoricizedMetric introduced above represents measures actually derived from the operated BusinessApplications in the past, i.e. lagging indicators. Metrics which quantify aspects of the application landscape structure, which are here considered leading indicators, appear in Figure 4.5 as derived attributes, and are subsequently explained.

4.3.1.1.1. serviceAvailability

The serviceAvailability-metric of an Interface is calculated as the probability that the Interface is operational, based on the respective failure propagation structure, and assuming an availability A for each BusinessApplication (i.e. it is assumed that a BusinessApplication fails with probability $1-A$, leading to all its Interfaces being not operational). In this context, it is assumed, that, in the first place, BusinessApplications fail independently.

Figures 4.1 and 4.2 exemplify this: Assuming a BusinessApplication-specific availability of $A = 0.9$, and BusinessApplications failing independently, the following Interface-specific availabilities can be calculated:

- *ExecuteTransaction* is operational, iff[3] all three BusinessApplications are operational. Thus, $serviceAvailability(ExecuteTransaction) = 0.9^3 = 0.729$ follows.

[3] *iff* is shorthand for *if and only if*.

- *AccountInfo* of the *EmployeePortal* is operational, iff the *EmployeePortal* and the *OrderSystem* are operational.
  Consequently, $serviceAvailability(EmployeePortal :: AccountInfo) = 0.9^2 = 0.81$ holds.

Of course, the Interfaces no longer fail independently.

Formally, serviceAvailability can be defined as the sum of the occurrence probabilities of all distinct combinations of BusinessApplications failing, which lead to the Interface under consideration being operational. Formalizing this relies on the following concepts:

- $AppNr$ is the number of BusinessApplications in the application landscape.
- $b \in \mathbb{B}^{AppNr}$ assigns to each BusinessApplication i, whether it is operational ($b_i = true$) or has failed ($b_i = false$).
- $F_{oi} : \mathbb{B}^{AppNr} \mapsto \mathbb{B}$ indicates for each combination of BusinessApplications failing (as represented by a $b \in \mathbb{B}^{AppNr}$), whether Interface oi is still operational or fails. Thus, $F_{oi}(b)$ is dependent on the application landscape structure. It is defined in detail in Appendix B.1.1.
- $I(oi) = \{b \in \mathbb{B}^{AppNr} | F_{oi}(b) = true\}$, the set of all vectors b, which denote a state of the application landscape in which oi is operational
- $|b| = |\{i | b_i = true\}|, b \in \mathbb{B}^{AppNr}$, the number of positions b_i with $b_i = true$

Now serviceAvailability can be defined as:

$serviceAvailability(oi) = \sum_{e \in I(oi)} A^{|e|}(1 - A)^{AppNr - |e|}$

Depending on the actual models at hand, a tool might approach calculation differently, as the case study presented in Chapter 5 shows.

4.3.1.1.2. failureImpact

The failureImpact-metric, characterizing the impact of a BusinessApplication failing on the Interfaces in an application landscape, calculates how the serviceAvailabilities of the Interfaces in the application landscape degrade, if the BusinessApplication under consideration fails.

Figures 4.7 and 4.8 exemplify this. Figure 4.7 annotates the serviceAvailability-values at the Interfaces. Figure 4.8 does this, too, however, under the assumption that the *OrderSystem* has failed. Consequently, some serviceAvailabilities, which are thus shown in red, are reduced to 0.

failureImpact for the *OrderSystem* is now computed as the average of the deterioration over all Interfaces: $\frac{1}{5}(0.81 + 0.729 + 0.9 + 0.9 + 0) = 0.6678$

A formal definition and interpretation of failureImpact can be based on a "conditional serviceAvailability" and the expected number of operational interfaces. The number of operational Interfaces in an application landscape can be defined by introducing a binary

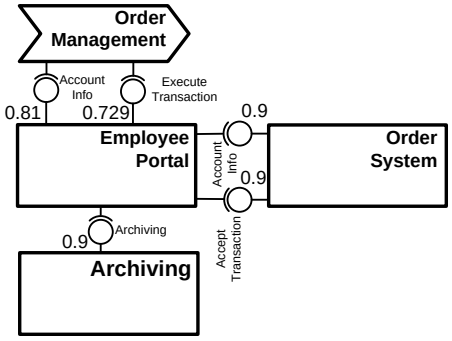


Figure 4.7.: serviceAvailability metrics

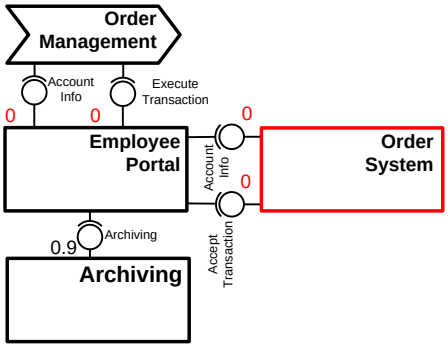


Figure 4.8.: OrderSystem has failed

variable for every Interface oi in the application landscape:

$$operational(oi) = \begin{cases} 1 & iff\ oi\ is\ operational \\ 0 & otherwise \end{cases}$$

Thus, the number of operational Interfaces is $\sum_{oi \in Interface} operational(oi)$.

Now, $E\left[\sum_{oi \in Interface} operational(oi)\right] = \sum_{oi \in Interface} E\left[operational(oi)\right]$ holds. $operational(oi)$ is a binomially distributed random variable, with $operational(oi) \sim B(1, serviceAvailability(oi))$, which leads to $E[operational(oi)] = serviceAvailability(oi)$. Thus, the expected share of operational Interfaces in the application landscape is $\frac{\sum_{oi \in Interface} serviceAvailability(oi)}{|Interface|}$.

Conditional serviceAvailabilities give the probability that Interface oi is operational, given BusinessApplication ba has failed:

$$p(oi\ working \mid ba\ failed) = \frac{p(oi\ working\ \wedge\ ba\ failed)}{p(ba\ failed)} =$$

$$= \frac{\sum_{e \in \{s \in \mathbb{B}^{AppNr} | F_{oi}(s) = true \wedge s_{ba} = false\}} A^{|e|}(1-A)^{AppNr-|e|}}{\sum_{e \in \{s \in \mathbb{B}^{AppNr} | s_{ba} = false\}} A^{|e|}(1-A)^{AppNr-|e|}}$$

Now, failureImpact of BusinessApplication ba can be defined as:

$failureImpact(ba) = \frac{\sum_{oi \in Interface} (serviceAvailability(oi) - p(oi\ working \mid ba\ failed))}{|Interface|}$, which is the reduction in the expected share of operational Interfaces in a situation where ba is known to have failed.

4.3.1.1.3. fullSupportAvailability

fullSupportAvailability of a BusinessProcess is defined as the probability that all Interfaces used by the BusinessProcess are operational. It can thus be seen as the share of the space in time during which the process has full support by IT.

Looking again at Figures 4.1 and 4.2 as an example, the BusinessProcess *OrderManagement* has all used Interfaces operational, iff all three BusinessApplications are in turn operational. Thus $fullSupportAvailability(OrderManagement) = 0.729$ holds.

Formally, the metric can be defined similar to serviceAvailability, however, only summing up the probabilities of the cases, where all Interfaces used by the BusinessProcess under consideration are operational, as formalized in Appendix B.1.2.

4.3.1.1.4. avSupportAvailability

While fullSupportAvailability targets the probability that all IT functionality supporting a process is available, the process, especially if it encompasses manual activities, might very well be able to operate in cases where some functionality is not available.

This is targeted by avSupportAvailability, which is the expected value of the share of Interfaces used by a BusinessProcess that are operational. This expected value can be calculated by using again the binary variables $operational(oi)$, this time to define the number of operational interfaces among the interfaces used by a BusinessProcess *bp*: $\sum_{oi \in bp.processSupport} operational(oi)$.

Using again the relationship $E[operational(oi)] = serviceAvailability(oi)$, $avSupportAvailability(bp) = \frac{\sum_{oi \in bp.processSupport} serviceAvailability(oi)}{|bp.processSupport|}$ can be deduced.

Taking again up the example from Figures 4.1 and 4.2, applying the above equation leads to $avSupportAvailability(OrderManagement) = \frac{1}{2}(0.81 + 0.729) = 0.7695$

4.3.1.1.5. processImpact

processImpact of a BusinessApplication resembles failureImpact, however, considering only Interfaces directly supporting a BusinessProcess. Thus, the metric does not consider Interfaces that are not directly relevant to business, their failure only leading to damage as it results in other Interfaces failing.

In the example from Figures 4.7 and 4.8, $processImpact(OrderSystem) = \frac{1}{2}(0.81 + 0.729) = 0.7695$ holds.

Formally, the metric is defined similar to the failureImpact: $processImpact(ba) = \frac{\sum_{oi \in BusinessInterface} (serviceAvailability(oi) - p(oi\ working \mid ba\ failed))}{|BusinessInterfaces|}$.

In this context, $BusinessInterfaces = \bigcup_{bp \in BusinessProcess} bp.processSupport$ holds.

4.3.1.2. Discussing the Basic Information Model

A model should be "as simple as possible, but not simpler"[4]. Several simplifications made by the basic information model are now discussed and introduced as a basis for the extended information model presented in Section 4.3.2.

4.3.1.2.1. Coarse-Grained View on Business Processes

The coarse-grained modeling of the BusinessProcesses does not allow capturing to what extent a BusinessProcess relies on a specific Interface it uses. Different Interfaces may be of different criticality to a specific BusinessProcess, e.g., if this process is robust to failures of a certain Interface. This may, e.g., be the case, if the BusinessProcess contains manual processing and the employees are able to find workarounds, if a certain Interface fails. The extended information model addresses this issue by introducing a loss function.

4.3.1.2.2. Errors in Estimating Failure Probabilities

The above model might **overestimate** failure probabilities at Interfaces. Figure 4.2 indicates that *ExecuteTransaction* relies on *Archiving* and *AcceptTransaction*. However, assuming that in reality the system only needs "archiving" a transaction in special cases, there might be successful calls to *ExecuteTransaction*, although the *Archiving*-Application is down. Also, one can imagine the connector between the *EmployeePortal* and the *OrderSystem* being based on messaging. In this case, it might be possible that the *EmployeePortal* is not affected by the *OrderSystem* failing, as it can put its messages into the respective queue, from which they are taken by the *OrderSystem* for processing, once it is operating again. Considering such cases, the basic information model can be understood as depicting a worst case estimation.

Contrastingly, there might also be cases where the basic information model **underestimates** failure probabilities. Consider, e.g., that the *EmployeePortal* tries to call *Archiving*. Due to a failure in *Archiving*, the call fails, sending the *EmployeePortal* itself into an undefined state, in which it is no longer able to correctly answer calls to its *AccountInfo* interface. However, it might be assumed that transactions in a Business-Application are isolated to an extent that makes the above described scenario unlikely and thus negligible.

Whether a set of assumptions is realistic enough to be applicable in a specific use case can be checked by comparing the metrics calculated from the application landscape structure to failure metrics collected in operating the respective business applications, as discussed with M-40 (Section 4.5.1).

Basically, the above described cases of over- and underestimating failure probabilities can be addressed by introducing additional probabilities into the models, as exemplified by Figure 4.9.

[4]"Make everything as simple as possible, but not simpler" is usually attributed to Albert Einstein.

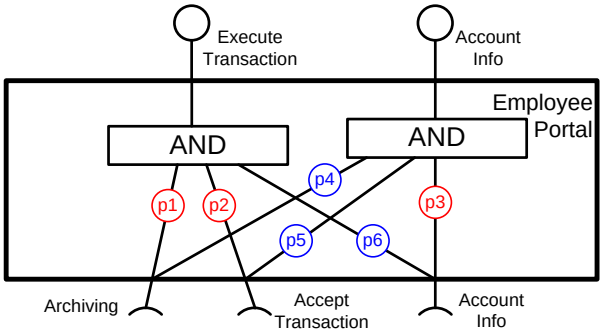


Figure 4.9.: Adding failure propagation probabilities to intra-application details

Basically, Figure 4.9 is built on Figure 4.2. However, the dependencies already present in Figure 4.2 are augmented by probabilities ($p1$, $p2$, and $p3$ in the red circles) indicating the probability that a failure propagates and impacts the state of the subsequent node. The above example of *Archiving* only being necessary in some calls to *ExecuteTransaction* could be represented by a $p1 < 1$.

Moreover, additional dependencies are added, with the respective failure propagation probabilities ($p4$, $p5$ and $p6$) added in blue for graphical contrast. This enables modeling that, by an implementation-specific mechanism, a failure propagates along a dependency that has no functional reason. The example of a call to a failed Archiving-Interface affecting AccountInfo could thus be modeled by a $p4 > 0$.

4.3.2. I-91: Extended Information Model

Information Model Pattern Overview	
Id	I-91
Name	Extended Failure Propagation Structures
Alias	
Summary	This information model pattern provides extended structures for modeling failure propagation.
Version	1.0

As hinted above, addressing the problems outlined in Section 4.3.1.2 relies on an extended information model, which includes the concepts exemplified in Figures 4.10 and 4.11.

Figure 4.10 introduces a *connectionAvailability*, representing the availability of the connection via which a BusinessApplication uses an Interface offered by another BusinessApplication. Moreover, details about the technical realization of an Interface, together with its probability to propagate a failure can be modeled. Thus, it can, e.g., be captured that an Interface relying on messaging is less likely to propagate a failure than one based

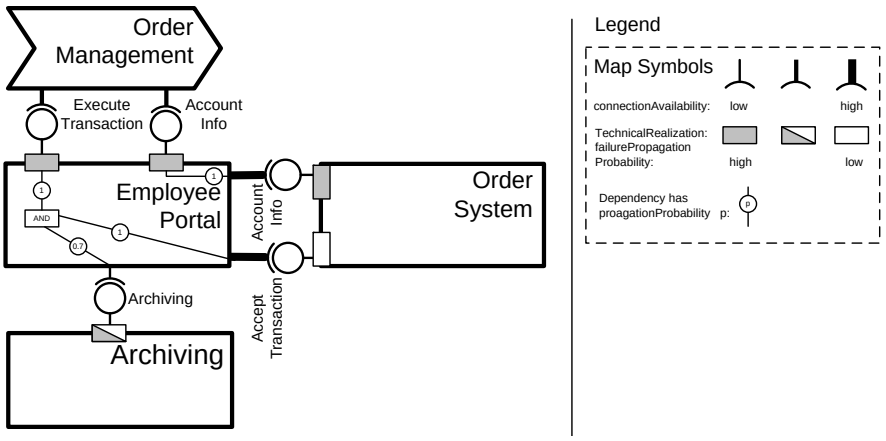


Figure 4.10.: Example of additional concepts in the extended information model

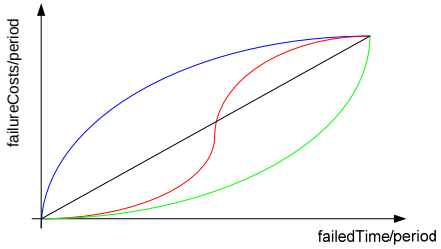


Figure 4.11.: Exemplary loss functions giving the loss induced by an Interface failing

on synchronous function calls. Failure propagation probabilities are also added to the application-internal dependency structures, as introduced in Figure 4.9.

Figure 4.11 presents three examples of loss functions, which are introduced for every Interface usage of a BusinessProcess, thus capturing what damage a failing Interface can induce in a BusinessProcess. The blue curve models a situation, where the costs grow quickly first, but the growth decreases. The green curve behaves the opposite way, with short failures being not very costly, but the costs growing faster with growing failed time. The red curve is a typical s-curve, with a critical point, at which the failure costs grow very fast.

4.3.2.1. UML Model and Concept Definitions

Figures 4.12 and 4.13 introduce the extended information model, with the concept definitions given below.

BusinessApplication, BusinessProcess, Interface, and HistoricizedMetric are defined as with the basic information model.

UsageByProcess and LossFunction As exemplified in Figure 4.11, a LossFunction assigns a monetary loss to a given downtime in a specific space in time, e.g., a month,

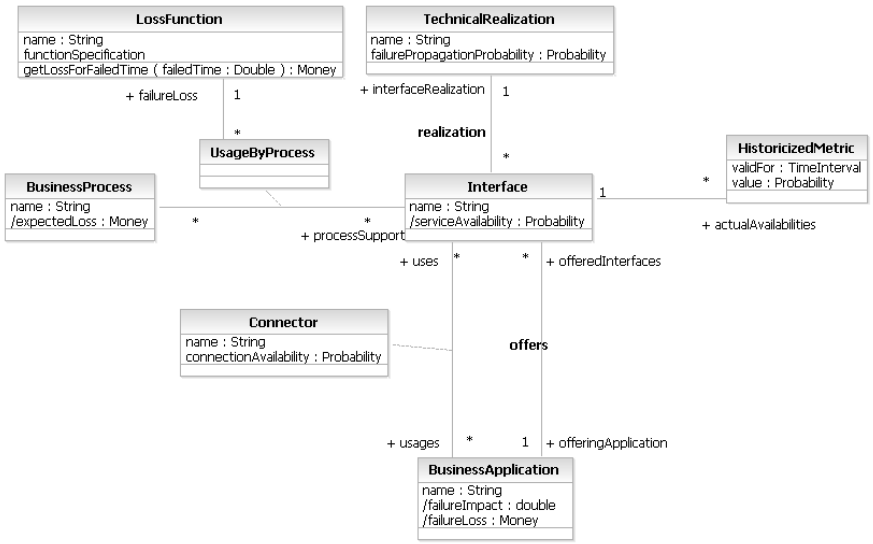


Figure 4.12.: Extended information model for the application landscape structure

which is modeled by the operation getLossForFailureTime. Such a LossFunction is assigned to every usage of an Interface by a BusinessProcess (association class UsageByProcess), in order to indicate to what extent a process is affected by its technical support failing.

Connector The Connector association class, as introduced in the basic information model, is here augmented by a name, which allows indicating, how the connection is realized (e.g., LAN, internet, mobile wireless), and a connectionAvailability. Thus, connectionAvailability indicates the probability that the respective connector is available and data can be sent over it.

TechnicalRealization The TechnicalRealization of an Interface indicates the technology employed in its realization, together with the probability that a failure actually propagates via the Interface. This work proposes an (adaptable and extendable) set of realizing technologies, based on [En08], ordered here by decreasing probability to propagate a failure. These concepts can be entered into an application landscape model as instances of the class *TechnicalRealization*:

Presentation The connection is based on a user interface. If another BusinessApplication uses such an Interface, screen scraping [SS03] is an approach to realize such a connector. In an average case, an Interface can be supposed to be at least as susceptible to a failure than synchronous calls, as the user interface it is likely to communicate with business logic via this technology.

Synchronous function calls Synchronous remote procedure calls to the business logic of a BusinessApplication.

Messaging A connection at the business logic layer, however, not relying on synchronous function calls but on messaging. Such an interface is less likely to

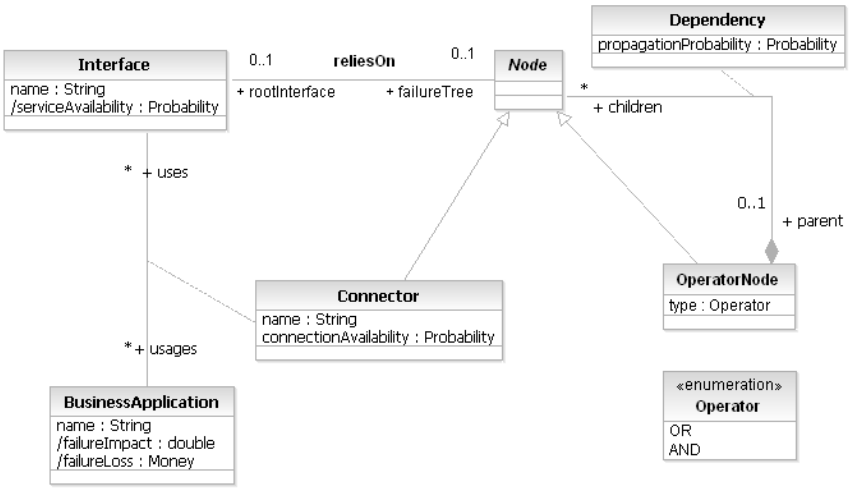


Figure 4.13.: Extended information model for application-internal details on failure propagation

propagate a failure, as in case of a failure, messages can remain in a message queue for later processing, once the respective BusinessApplication is operating again. Messaging also includes bulk data exchange via file transfer.

Database A connection realized via a shared database. The Interface is considered as offered by the BusinessApplication owning the respective database. Such an Interface can be supposed to have a relatively low failure propagation probability, as it does not rely on the BusinessApplication owning the database being operational.

The only extension made to the information model for application-internal details is the propagation probability added to the Dependency-association class.

4.3.2.2. Metrics

Due to above information model extensions, metrics definitions have to be adapted, and additional metrics are now possible.

4.3.2.2.1. serviceAvailability

As with the basic information model, serviceAvailability here signifies the probability that an Interface is operational, based on the respective failure propagation structure, and assuming an availability A for each BusinessApplication. However, calculating serviceAvailability now has to take into account the additional information provided by models following the extended information model:

- Connectors may fail, as described by the respective *connectionAvailability*
- *failurePropagationProbability*-values as indicated by the TechnicalRealizations of the Interfaces
- *propagationProbability*-values of the Dependencies of the BusinessApplication-internal views

Appendix B.2.1 presents a formal definition of serviceAvailability for the extended information model.

4.3.2.2.2. failureImpact

Similar to the serviceAvailability above, also the failureImpact has to be adapted to the extended information model, while it still keeps its meaning: How do the serviceAvailabilities of the Interfaces in the application landscape degrade, if the BusinessApplication under consideration fails?

Specifically, the adaptations are as follows:

- The serviceAvailability for the extended information model has to be used.
- An extended version of the "conditional serviceAvailability" essential to the definition of failureImpact has to be used.

A formal definition of failureImpact for the extended information model, including the explication of the conditional serviceAvailability is available in Appendix B.2.2.

4.3.2.2.3. expectedLoss

The basic information model provided three metrics for characterizing how failure propagation affects the IT support of a BusinessProcess (*fullSupportAvailability*, *avSupportAvailability*, *processImpact*). Basically, these metrics could be adapted to the extended information model. However, the extended information model uses the LossFunction to provide metrics with more expressive power, especially to business stakeholders.

expectedLoss for a business process is the sum of the losses induced at the BusinessProcess by failures at the supporting Interfaces, as given by the respective LossFunctions.

Formally this can be defined as $expectedLoss(bp) = \sum_{ubp \in bp.usageByProcess} ubp.failureLoss.getLossForFailedTime(1 - ubp.processSupport.serviceAvailability)$

4.3.2.2.4. failureLoss

failureLoss resembles the processImpact defined with the basic information model, however incorporating the LossFunctions. failureLoss for a BusinessApplication *ba* is defined as the increase in the sum of expectedLoss for all BusinessProcesses, given the information that *ba* fails in the period covered by the loss function.

A formal definition is given in Appendix B.2.3.

4.3.2.3. Discussing the Extended Information Model

The assumptions of the extended information model still restrict its expressive power. While these restrictions are here accepted, for reasons of simplicity and feasibility of data collection, they are nevertheless discussed subsequently.

A major simplification concerns the LossFunction. Firstly, it is deterministic and only dependent on the failed time in a specific period. However, situations can easily be imagined, in which, e.g., the loss induced by a failure is random, depends on the time at which the failure occurs, or depends on the length and number of failures, instead of only the total failed time.

Also the approach that losses resulting from different Interfaces are simply added up may be unrealistic in situations where, e.g., the damage is exceptionally high, if *all interfaces supporting a process fail together*.

However, for reasons of simplicity, considering that handling high-dimensional LossFunctions might be prohibitively complex in practice, the loss function approach introduced above is kept. Before delving into more complex modeling, one should validate the above model and check whether it is sufficient.

Moreover, the extended information model still does not consider possible information about dependencies between the basic failure events at *BusinessApplications* or *Connectors*. One can imagine failures, which lead to specific sets of connectors realized via the Internet failing together.

Additionally, the model is based on stateless abstractions of BusinessApplications. As with the above simplifications, verification has to ensure that the model is only applied in cases where the simplifications are possible. The methodologies presented in Section 4.5 take this into account.

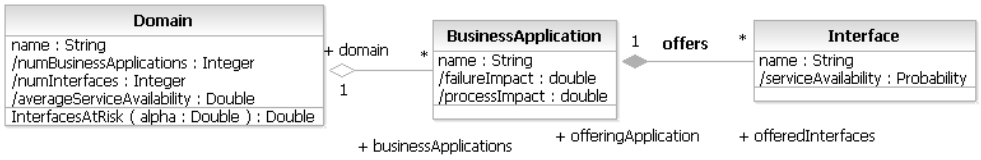


Figure 4.14.: Information model for aggregating business applications into Domains

4.3.3. I-92: Aggregating Metrics

Information Model Pattern Overview	
Id	I-92
Name	Aggregating Availability Metrics
Alias	
Summary	This information model pattern provides concepts for aggregating availability metrics from I-90 or I-91 using a domain structure.
Version	1.0

Dividing BusinessApplications into domains is common in managing an application landscape. Section 3.1.2.3 showed metrics at the domain level relatively important, and indicated, that metrics at the application landscape level become important with larger application landscapes. This sets the stage for approaches to aggregating metrics. Figure 4.14 presents a corresponding information model fragment. It constitutes a means of managing the complexity of possibly hundreds of business applications by a *divide-and-conquer* approach, grouping together BusinessApplications that are similar according to a given criterion, e.g., the kind of business activity they support [En08]. While this work does not focus on different kinds of grouping criteria, it provides approaches for aggregating metrics to derive Domain-specific values, as this allows the use of metrics also on the reduced-complexity, aggregated view created by the Domain structure.

The information model fragment shown in Figure 4.14 contains two size metrics: The number of Interfaces offered by BusinessApplications in a Domain, and the number of the BusinessApplications themselves.

4.3.3.1. Aggregated Metrics: Characteristics of a Failure Distribution

With respect to aggregating metrics, what usually crosses one's mind first is deriving simple averages. However, in aggregating serviceAvailability metrics, interpreted as the probability of an Interface being operational, a difficulty occurs: basically, it is not defined what an average of probabilities is.

However, this technical problem can be addressed quite straightforwardly, by using, similarly as with the *failureImpact* in Section 4.3.1.1.2, an indicator variable $operational(oi)$

EmployeePortal	OrderSystem	Archiving	Probability	# failed interfaces
false	false	false	0.001	5
false	false	true	0.009	4
false	true	false	0.009	3
false	true	true	0.081	2
true	false	false	0.009	5
true	false	true	0.081	4
true	true	false	0.081	2
true	true	true	0.729	0

Table 4.1.: Combination of business applications having failed or being operational in the example from Figures 4.1 and 4.2

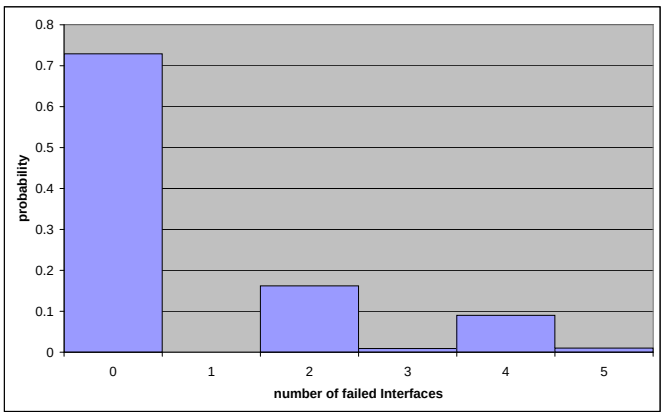


Figure 4.15.: Distribution of the number of failed interfaces in the example from Figures 4.1 and 4.2

for each Interface, being 1 iff the respective Interface is operational. Remember that $operational(oi) \sim B(1, serviceAvailability(oi))$ holds.

The number of operational Interfaces in a Domain d is $opInterfaces(d) = \sum_{oi \in d.businessApplications.offeredInterfaces} operational(oi)$, another random variable, which is however not binomially distributed, as the Interfaces do not necessarily fail independently, due to failure propagation. Consequently, the number of failed Interfaces in a Domain d, *failedInterfaces(d)*, can be defined as $failedInterfaces(d) = d.numInterfaces - opInterfaces(d)$.

Demonstrating this using the example from Figures 4.1 and 4.2, Table 4.1 derives the occurrence probability and number of failed Interfaces for every possible combination of BusinessApplications failing or operational in the application landscape. Figure 4.15 shows the results as a histogram.

Based on information about the distribution of the number of failed Interfaces, statistically meaningful aggregation is possible by deriving characteristics of the distribution, or of the distribution of operational Interfaces, if this is more convenient.

4.3.3.1.1. averageServiceAvailability

For the expected value, $E\left[opInterfaces(d)\right] = \sum_{oi \in d.businessApplications.offeredInterfaces} serviceAvailability(oi)$, holds. To allow cross-domain comparisons, this can be normalized to $averageServiceAvailability(d) = \frac{\sum_{oi \in d.businessApplications.offeredInterfaces} serviceAvailability(oi)}{|d.businessApplications.offeredInterfaces|}$.

Deriving other characteristics (e.g., the variance) relies on more sophisticated calculations, as they need more information (in case of the variance, e.g., covariances of the summands making up $opInterfaces(d)$). However, one characteristic, based on a quantile of the distribution of the number of failed interfaces, *failedInterfaces(d)* , and called here *Interfaces at Risk*, is subsequently detailed.

4.3.3.1.2. Interfaces at Risk

$IaR_\alpha(d)$, the *Interfaces at Risk* in a Domain d at confidence level α, is the α-quantile of the distribution of failed Interfaces. This metric, which is not only applicable to Domains, but also to the application landscape as a whole, is able to measure to what extent a Domain or the application landscape is prone to large failures affecting lots of Interfaces, which is exemplified by Figures 4.16 to 4.19 and Table 4.2.

Although both variants (Figure 4.16 and 4.18) have the same serviceAvailability-values at the corresponding Interfaces, leading to the same averageServiceAvailability in Table 4.2, their failure distributions in Figures 4.17 and 4.19 are nevertheless different. In variant 1, the risk of a large failure, e.g., affecting all interfaces, is much higher. This is characterized by the distinctly higher $IaR_{0.95}$ for the variant relying on merely one Archiving-System.

For reasons of comparability, it makes sense not to work with the absolute number of failed Interfaces, but a percentage, as this allows comparing the metrics for domains or application landscapes of different size.

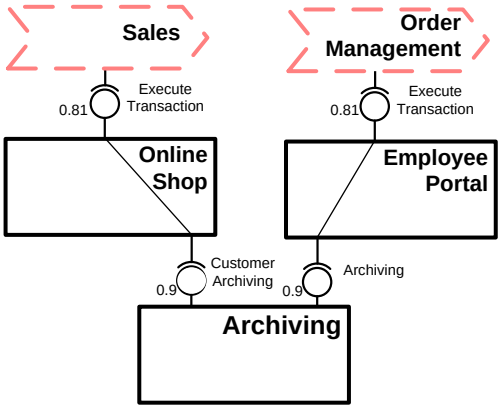


Figure 4.16.: Shared Archiving System

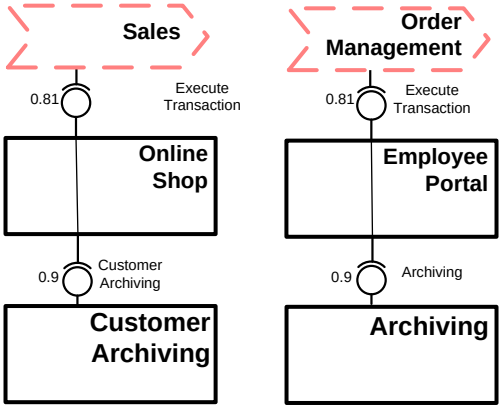


Figure 4.18.: Two Archiving Systems

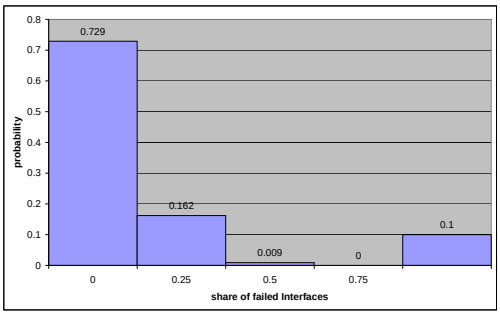


Figure 4.17.: Failure distribution for Figure 4.16

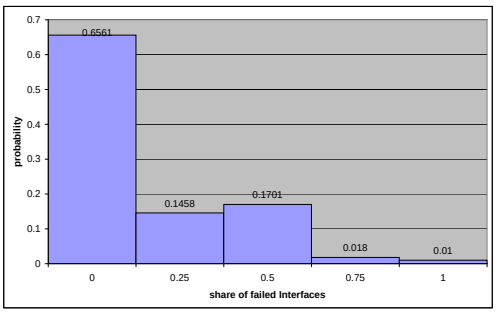


Figure 4.19.: Failure distribution for Figure 4.18

Metric	One Archiving System	Two Archiving Systems
averageServiceAvailability	0.855	0.855
$IaR_{0.95}$	1	0.5

Table 4.2.: Metrics characterizing the application landscapes in Figures 4.16 and 4.18

4.3.3.2. Focusing on Inter-/Intra-Domain Effects in Metrics Aggregation

The Domain structure introduced by the information model fragment in Figure 4.14 can also be used to focus the metrics on inter- or intra-domain effects, with two examples sketched below.

- The serviceAvailability can be modified to consider only failures originating from other Domains. This means that in calculating the serviceAvailability of Interfaces belonging to a BusinessApplication in Domain d, $A = 1$ is assumed for all BusinessApplications from d. Thus, the modified metric is able to measure, to what extent other Domains affect an Interface. The metric gives the maximum serviceAvailabilities achievable via decoupling the Domain from its environment.
- The failureImpact metric can be modified to consider only effects in other Domains. When calculating the failureImpact of a BusinessApplication in Domain d, serviceAvailability-deteriorations of Interfaces belonging to BusinessApplications in Domain d are not considered. A failureImpact modified like this is able to measure how a BusinessApplication affects availabilities in other Domains.

4.3.4. I-93: Comparing Scenarios

Information Model Pattern Overview	
Id	I-93
Name	Comparing Scenarios
Alias	
Summary	This information model pattern allows comparing scenarios of the application landscape created by proposals for limiting failure propagation.
Version	1.0

Addressing concern C-111 using the metrics introduced above relies on being able to compare different scenarios of an application landscape. Figure 4.20 introduces an information model fragment supporting such comparisons, which extends information model patterns I-90 or I-91, possibly combined with I-92.

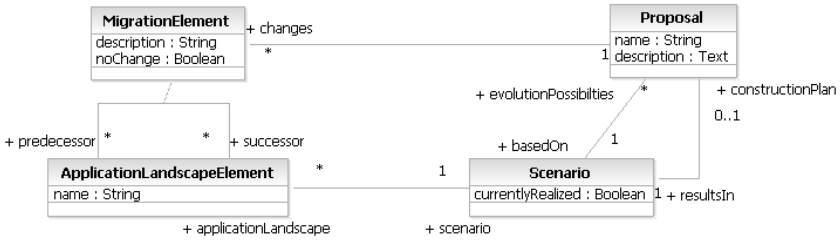


Figure 4.20.: Information model for comparing scenarios of an application landscape

The concepts introduced in Figure 4.20 are defined as follows:

ApplicationLandscapeElement These are the elements of the application landscape (Connector, Interface, BusinessApplication, ...) as described by I-90 or I-91, or aggregations of them, as modeled in I-92. In pattern integration, this can be considered by making these classes subclasses of *ApplicationLandscapeElement*.

Scenario This concept refers to a specific past, present, planned, or possible version of the application landscape, described by the respective ApplicationLandscapeElements.

Proposal This class represents a proposal for a project changing the application landscape from one Scenario to another.

MigrationElement These elements describe each one specific measure being part of a proposal for changing the application landscape. An element gives for a specific ApplicationLandscapeElement (predecessor) how it is modified, if the application landscape is changed (successor) according to a specific proposal. If an ApplicationLandscapeElement is connected to an element with identical information, and *noChange* set to true, the respective element is not changed by the proposal. In

such cases, the predecessor and successor information is still relevant to enable metrics comparisons.

Depending on the specific information model pattern used (e.g., I-90 vs. I-91), indicators of the project effort for realizing a proposal have to be adapted. However, some basic measures relevant in limiting failure propagation and associated counts helpful in estimating project effort can be given:

Removing or changing dependencies: Remove dependencies, or change them to technologies less likely to propagate a failure (e.g., from a synchronous function call to messaging).
Effort indicator: Number of removed/changed dependencies

Multiple deployments A software is deployed multiple times.
Effort indicator: Number of deployments added; Number of business applications for which multiple deployments are made

Redesign of BusinessApplications Business applications can be redesigned with the goal to make them less likely to fail in the first place, lowering *propagationProbability*-values, or enabling them to use backup services (i.e. introducing OR-nodes into the failure trees)
Effort indicator: Number of changed business applications, possibly with an effort estimation for each change

Redesign application landscape Other redesigns, as removing BusinessApplications, replacing two or more BusinessApplications by one, or introducing a new BusinessApplication are here not covered in more detail, as they are not specifically targeted on limiting failure propagation.

In addition to giving above effort indicators per proposal, they can also be used on a per Domain basis, if a Domain structure as introduced with I-92 is used.

4.4. Viewpoint Patterns

In introducing the above information model patterns, exemplary visualizations have already been used. This section presents the viewpoint patterns for visualizing information according to these information model patterns.

4.4.1. Visualizing the Failure Propagation Structure

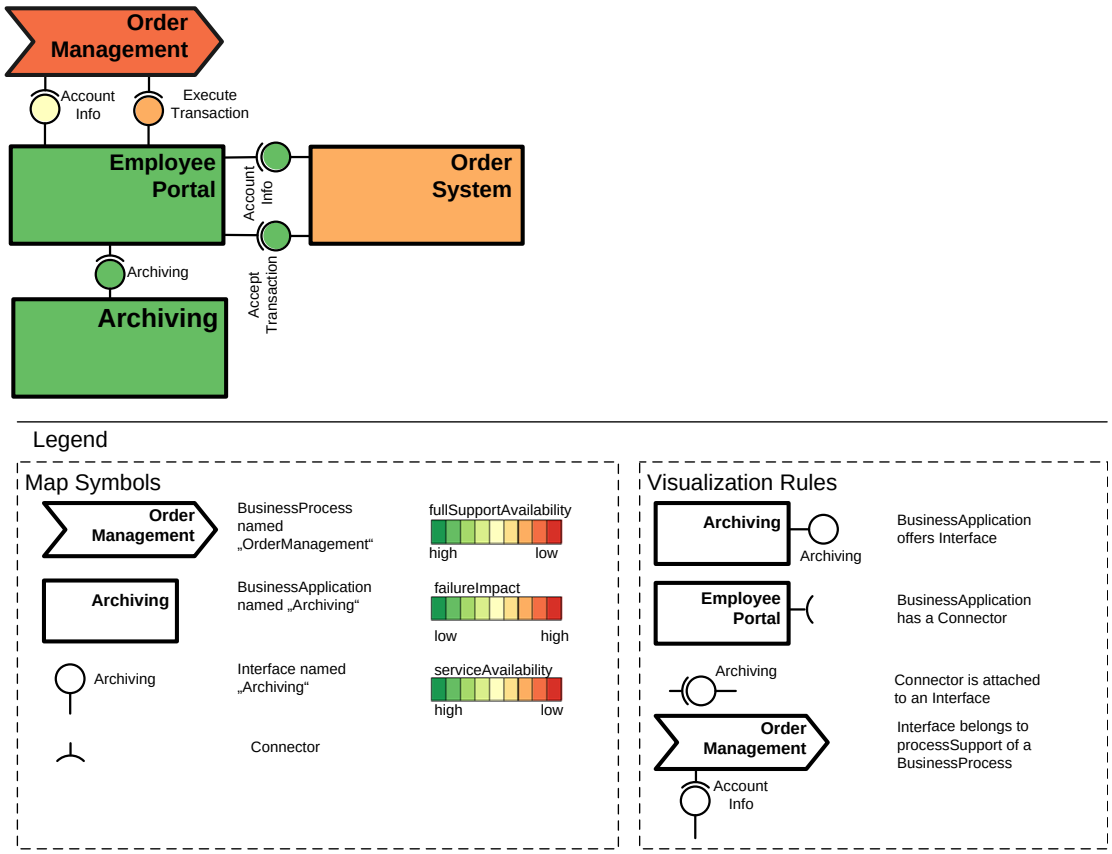
For visualizing inter- and intra-application aspects of the failure propagation structure of an application landscape, two sets of viewpoint patterns are supplied: one relying on the basic information model, and another one for the extended version.

4.4.1.1. Viewpoints Relying on the Basic Information Model

4.4.1.1.1. V-85: Basic Application Landscape View on Failure Propagation

Viewpoint Pattern Overview	
Id	V-85
Name	Basic Application Landscape View on Failure Propagation
Alias	
Summary	The viewpoint pattern visualizes the application landscape structure and how its elements are affected by or sources of failure propagation.
Version	1.0

Solution Section



The viewpoint pattern relies on information model pattern I-90. Regarding the metrics values displayed by the viewpoint pattern, the following variations are possible:

- Instead of *serviceAvailability* of an Interface, an *actualAvailability* (a HistoricizedMetric-value) may be visualized by the color of an Interface-lollipop
- Instead of *fullSupportAvailability*, *avSupportAvailability* may be visualized by the color of a BusinessProcess-chevron
- Instead of *failureImpact*, *processImpact* may be visualized by the color of a BusinessApplication-rectangle

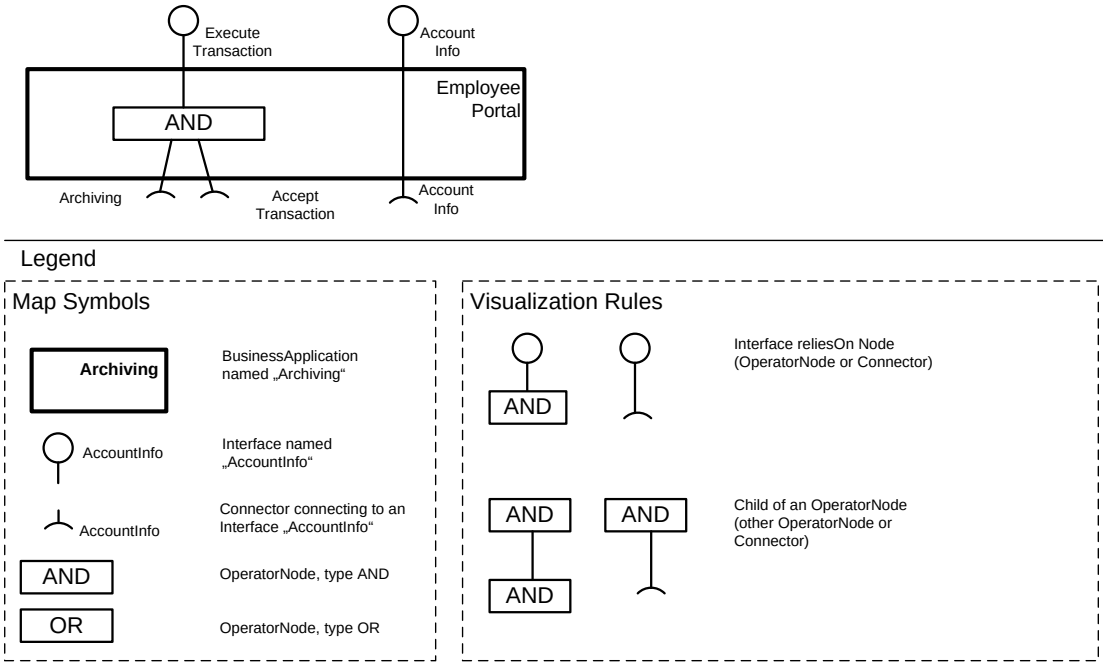
The layering principle, introduced in Section 2.1.3.2.3, may be applied to allow an user to switch, which of these metrics the map visualizes.

When specifying a map, ranges of metric values have to be assigned to color steps.

4.4.1.1.2. V-86: Basic Intra-Application View on Failure Propagation

Viewpoint Pattern Overview	
Id	V-86
Name	Basic Intra-Application View on Failure Propagation
Alias	
Summary	This viewpoint pattern visualizes how Interfaces offered by a BusinessApplication rely on Interfaces used by this BusinessApplication.
Version	1.0

Solution Section



The viewpoint pattern relies on information model pattern I-90.

V-86 visualizes intra-application details about failure propagation for a specific BusinessApplication by giving a failure tree for each Interface of the respective BusinessApplication.

4.4.1.1.3. V-87: Basic Intra-Application Failure Propagation Table

Viewpoint Pattern Overview	
Id	V-87
Name	Basic Intra-Application Failure Propagation Table
Alias	
Summary	This viewpoint pattern shows in a table how Interfaces offered by a BusinessApplication rely on Interfaces used by this BusinessApplication.
Version	1.0

Solution Section

Connector / Interface	Archiving	Accept Transaction	Account Info
Execute Transaction	X	X	
Account Info			X

The viewpoint pattern relies on information model pattern I-90.

The table contains a line for each Interface of a specific BusinessApplication. In this line, the Connectors, on which the respective Interface relies, represented as the columns of the table, are indicated via an ”X”. The OperatorNode in the respective failure tree has the type ”AND”.

Consequence Section Failure trees containing OR-nodes cannot be represented via V-87.

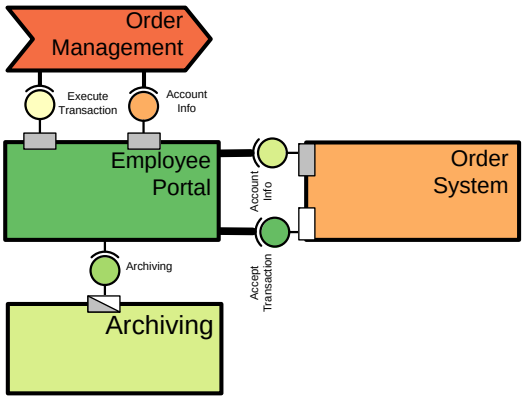
4.4.1.2. Viewpoints Relying on the Extended Information Model

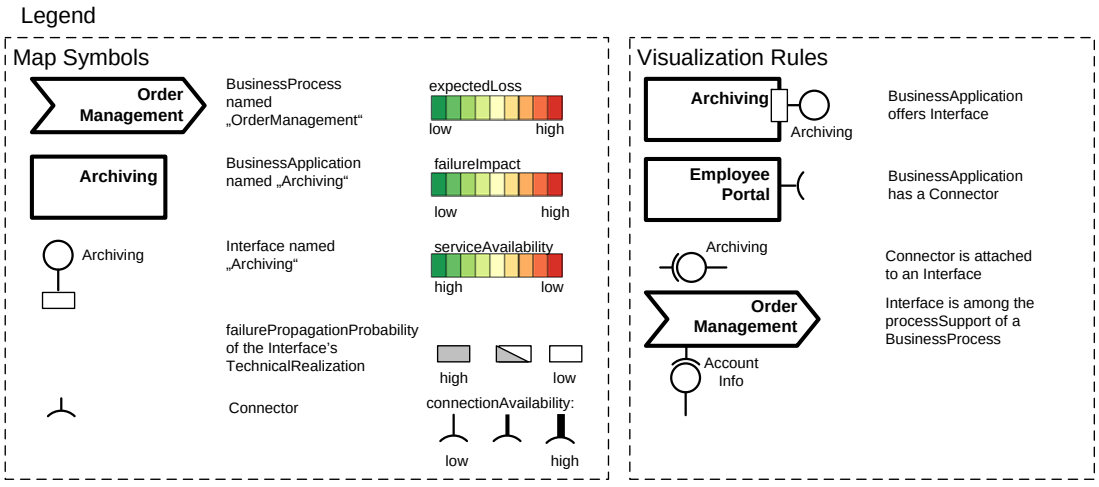
In order to represent information following information model pattern I-91, the viewpoint patterns need augmentations, which are basically centered on the graphical concepts already introduced in Figure 4.10.

4.4.1.2.1. V-88: Extended Application Landscape View on Failure Propagation

Viewpoint Pattern Overview	
Id	V-88
Name	Extended Application Landscape View on Failure Propagation
Alias	
Summary	This viewpoint pattern visualizes detailed information about the structure of an application landscape and how its elements are affected by, or sources of failure propagation.
Version	1.0

Solution Section





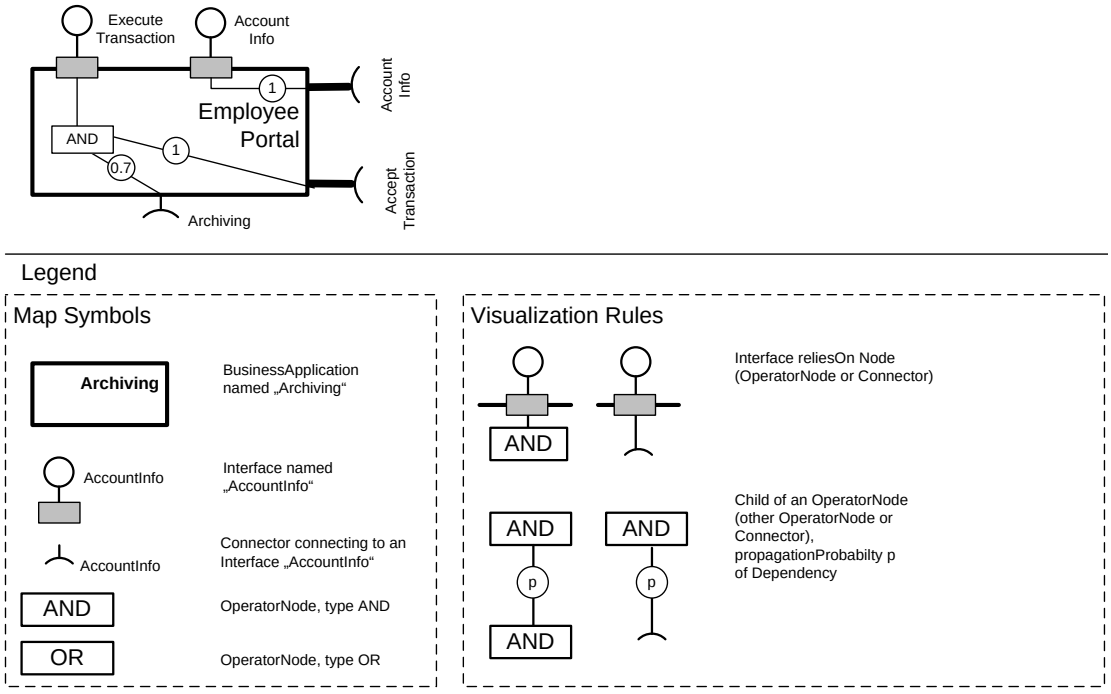
The viewpoint pattern relies on information model pattern I-91. Regarding the metrics values and parameter values displayed by the viewpoint pattern, some variations exist. Instead of *failureImpact*, the coloring of a BusinessApplication-rectangle could also visualize *failureLoss*, and of course, also here the layering principle is suitable to let an user choose which one to display.

Also here, ranges of metrics values have to be assigned to the color steps, which includes the input values *connectionAvailability* of the *Connectors*, and the *failurePropagationProbability* of the *TechnicalRealizations*.

4.4.1.2.2. V-89: Extended Intra-Application View on Failure Propagation

VPoint Pattern Overview	
Id	V-89
Name	Extended Intra-Application View on Failure Propagation
Alias	
Summary	This viewpoint pattern visualizes detailed information about how Interfaces offered by a BusinessApplication rely on Interfaces used by this BusinessApplication.
Version	1.0

Solution Section



The viewpoint pattern relies on information model pattern I-91. V-89 visualizes intra-application details about failure propagation for a specific BusinessApplication by giving a structure similar to a failure tree for each Interface of the respective BusinessApplication. This structure extends a failure tree by providing for each child of an OperatorNode the probability with which it propagates a failure to its parent.

4.4.1.2.3. V-90: Extended Intra-Application Failure Propagation Table

Viewpoint Pattern Overview	
Id	V-90
Name	Extended Intra-Application Failure Propagation Table
Alias	
Summary	This viewpoint pattern shows detailed information about how Interfaces offered by a BusinessApplication rely on Interfaces used by this BusinessApplication in a table.
Version	1.0

Solution Section

Connector / Interface	Archiving	Accept Transaction	Account Info
Execute Transaction	0.7	1	0
Account Info	0	0	1

The viewpoint pattern relies on information model pattern I-91. The table contains a line for each Interface offered by a specific BusinessApplication. The Connectors of the BusinessApplication under consideration to other Interfaces are the columns of the table. Then, each cell represents a Dependency from the respective Connector to a line-specific AND-OperatorNode, with a propagationProbability as given in the cell. The line-specific OperatorNode then describes the Interface represented by the line.

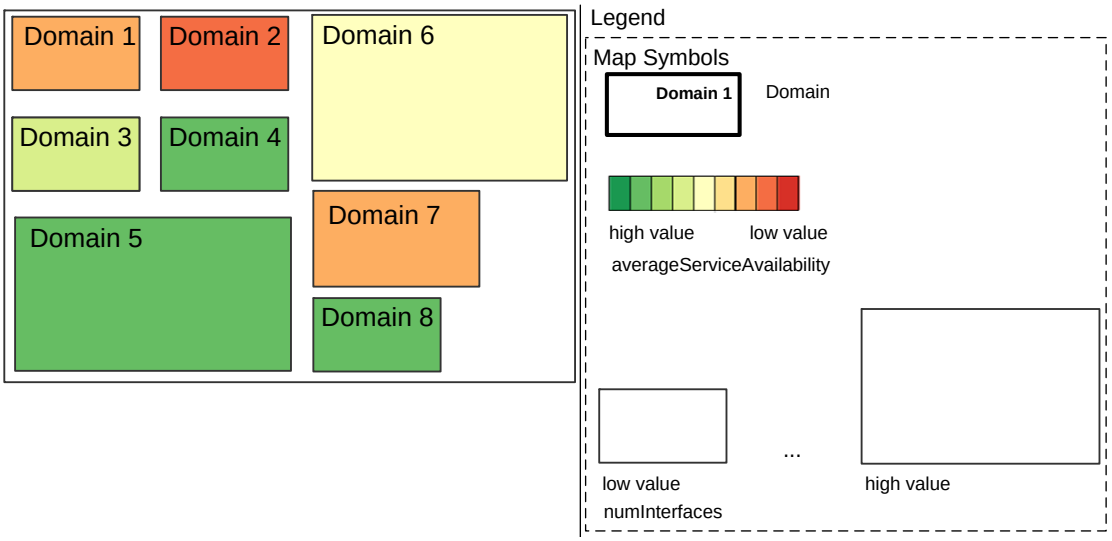
Consequence Section Failure propagation structures using OperatorNodes of type OR cannot be represented using V-90.

4.4.2. Aggregated Views for Large Application Landscapes

Views according to the above viewpoint patterns can get rather complex, if the number of BusinessApplications and Interfaces grows. In such cases, keeping an overview relies on viewpoint patterns working on a more coarse-grained level of detail by using a domain structure.

Viewpoint Pattern Overview	
Id	V-91
Name	Domain-Level Metrics Overview
Alias	
Summary	This viewpoint pattern gives an overview of different kinds of metrics values aggregated to the domain level.
Version	1.0

Solution Section



The viewpoint pattern relies on information model pattern I-92, together with I-90 or I-91. It displays metrics aggregated to the Domain level via the color of rectangles representing Domains. Besides the averageServiceAvailability, other aggregations, as described in Section 4.3.3, are possible. The layering principle again allows an user to interactively choose which of a set of metrics to display.

The size of the rectangle is used to indicate Domain size with two obvious metrics of Domain size being numBusinessApplications and numInterfaces, as contained in I-92. Conforming to Section 3.2.2.2.2, the rectangle sides are scaled by $\sqrt{metricvalue}$.

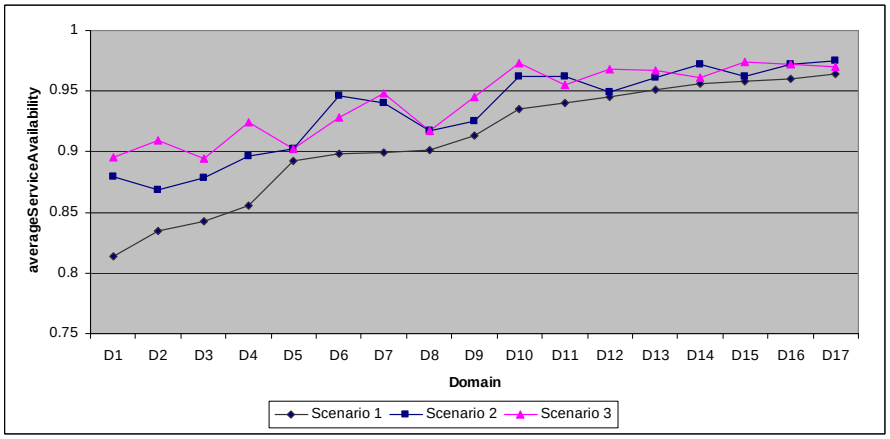
4.4.3. Comparing Scenarios

A methodology pattern for comparing different proposals aimed at limiting failure propagation needs viewpoint patterns for comparing different scenarios of an application landscape. Thus, two viewpoint patterns for comparing different scenarios of an application landscape are now introduced.

4.4.3.1. V-92: Scenario Comparison Profile

Viewpoint Pattern Overview	
Id	V-92
Name	Scenario Comparison Profile
Alias	
Summary	This viewpoint pattern compares values of a specific metric for a set of elements of the application landscape in different scenarios of the application landscape.
Version	1.0

Solution Section



The viewpoint pattern relies on I-93, integrated with I-90 or I-91, and possibly with I-92. It displays a *Scenario* as a graph, with the *ApplicationLandscapeElements* of a specific kind (e.g., Domains) on the x-axis and the values of the metric under consideration on the y-axis. Thus, an arbitrary number of scenarios can be compared with respect to one metric at a glance.

The ApplicationLandscapeElements are ordered according to their metric value in *one* of the Scenarios. Thus, the black line representing *Scenario 1* in the above example always moves upward from the left to the right.

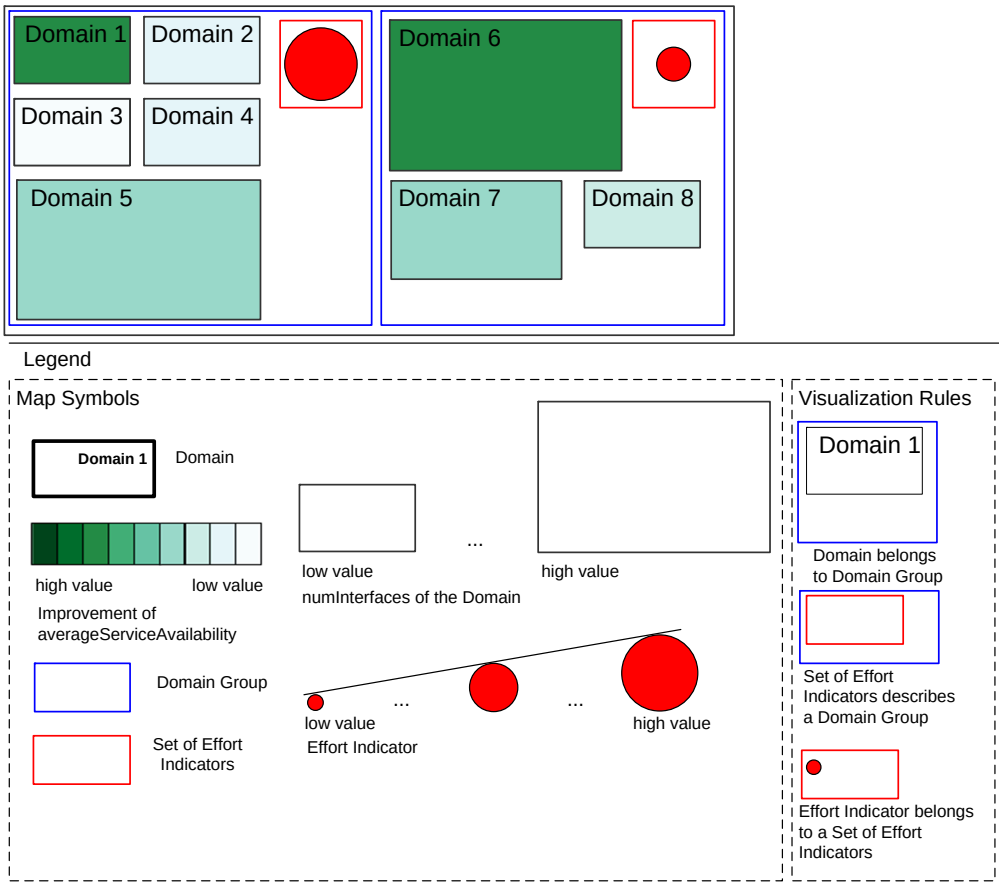
Consequence Section

The scenario comparison profile can be used in cases, where the Proposals transform the application landscape in a way that each ApplicationLandscapeElement to be displayed has exactly one successor in the Proposal's resultsIn-Scenario. Depending on the kind of elements to be displayed (Interface, BusinessApplication, BusinessProcess, Domain), this is more or less likely. While scenarios might differ regarding Interfaces, or perhaps BusinessApplications, it is less likely that they change the domain structure. Thus, using the scenario comparison profile depends on choosing an adequate level of granularity.

4.4.3.2. V-93: Scenario Comparison Map

Viewpoint Pattern Overview	
Id	V-93
Name	Scenario Comparison Map
Alias	
Summary	This viewpoint pattern compares two scenarios of an application landscape regarding metrics values of its elements and visualizes indicators of the effort necessary for changing from Scenario 1 to Scenario 2.
Version	1.0

Solution Section



The Scenario Comparison Map displays the differences in a specific metric between ApplicationLandscapeElements of the basedOn-Scenario and the resultsIn-Scenario of a

Proposal. The higher the improvement in the resultsIn-Scenario, the more green the respective map symbol is shown. If the resultsIn-Scenario is not better regarding the metric in all visualized ApplicationLandscapeElements, a diverging color scale, as used for V-91, can be employed.

While the map can only display a comparison regarding one metric at a time, layering can be used to allow a user to switch between different kinds of metrics.

In addition to metrics changes, other sizes are displayed: The rectangle size indicates, similar to V-91, a size metric of an object, which is especially useful if Domains are displayed. Sized circles can be used to display indicators of project effort (e.g., number of connections to be replaced).

Consequence Section

Basically, the scenario comparison map relies on an ApplicationLandscapeElement in the resultsIn-Scenario to have exactly one predecessor in the basedOn-Scenario to be able to calculate differences. However, this restriction can be avoided by specifically highlighting the objects only present in one of the scenarios, which can then be colored to indicate absolute metric values.

4.5. Methodologies

Section 4.1 presented two concerns from the problem domain "failure propagation and operational risk". Now, two methodology patterns addressing these concerns are introduced.

4.5.1. M-40: Failure Propagation Analysis

Methodology Pattern Overview	
Id	M-40
Name	Failure Propagation Analysis
Alias	
Summary	The methodology pattern uses metrics visualizations to analyze which BusinessApplications are sources and which Interfaces are affected by failure propagation.
Version	1.0

4.5.1.1. Problem Section

Methodology pattern M-40 addresses concern C-110 (see Section 4.1 for details), which deals with analyzing the effects and sources of failure propagation.

4.5.1.2. Solution Section

When using application landscape metrics, a considerable part of the effort occurs in introducing the metrics. Aspects of change management, such as sparking interest and getting approval of the respective stakeholders, belong to this effort. In this context, it should be detailed, in exactly which use cases the metrics are to be employed.

Another main effort in introducing metrics lies in getting and interpreting the necessary data. Thus, it has to be distinguished, whether data already available, e.g., in repositories already maintained for EA management, is to be used for metrics calculation, or if the data are yet to be collected.

If already otherwise collected and maintained data are to be used, it cannot be expected that they have been collected in a structure and according to definitions as presented with one of the respective information model patterns in Section 4.3. However, as these information model patterns contain rather basic concepts of an application landscape, it might very well be possible that at least some of them are present in the data, e.g., the (deployed) business applications. Other kinds of information, e.g., relating to the internal

failure propagation structures, might be less readily found in existing repositories. If it is not feasible to collect this information, rather general assumptions might be used, e.g., that all offered Interfaces of a BusinessApplication fail, if at least one of the Interfaces used by the application fails. However, when doing so, it should be discussed whether the assumptions are valid and useful in addressing the respective concern, and explore possibilities for this being not the case. The most basic decision in this respect is choosing, whether to rely on the basic information model (I-90) or using the extended one (I-91).

If data are yet to be collected, getting the support of stakeholders becomes even more important, but the advantage is that the definitions from the chosen information model patterns can most probably be directly used. However, also here, the collection effort might lead to using assumptions instead of more elaborate data collection.

In all cases, steps for validating the model have to be taken, in order to ensure that it is actually able to measure vulnerability to failure propagation in the respective use case. A very straightforward approach to achieve this is to contrast the results from the metrics with actual availability indicators collected in the operation of the application landscape. This can be done graphically and intuitively via an appropriate software map, e.g., V-85, V-88, or V-91, if configured to display both *serviceAvailability* and *actualAvailability* (HistoricizedMetric-values), or statistically. If it is only of relevance whether the metrics are able to establish an ordering of elements regarding their actual availabilities, Spearman's rank correlation coefficient [Fa99] can be used. If the goal is actually predicting availability, regression can be used to check the quality of the predictions.

Assumptions about failure propagation within business applications could be confirmed or refuted by source code analysis, or by analyzing log data about failures, if available. If assumptions have been made instead of actually collecting data, these data could be completed in cases where a validation reveals the assumptions to be problematic.

Finding the right value for the constant A (the assumed availability of each BusinessApplication) can be seen as a related problem. It should be set to a plausible value, to help interpret the metrics. However, the importance of setting A should not be overrated, as it is a rather arbitrary value, affecting more the absolute metrics values than their relations, which can be considered as most important in interpretation. However, it is important to leave A the same, once it has been set, in order to keep the metrics values comparable and focused on the application landscape.

Using the extended information model demands more attention in setting A, together with *failurePropagationProbability*-values of *TechnicalRealizations*, *connectionAvailability*-values of *Connectors*, and *propagationProbability*-values of *Dependencies*. The values have to realistically fit together.

After visualizing the metrics results on views according to, e.g., V-85, V-88, or V-91 (suitable especially for larger application landscapes), they have to be interpreted. As this relies on both metrics and domain experience, metrics and domain experts should interpret the results in shared meetings. The following interpretation hints might help:

- Interpretation can start with analyzing BusinessApplications having a high *failureImpact* and offering Interfaces with low *serviceAvailability*. Such applications appear critical, which could initially be confirmed by looking at actual availability metrics. Interpretation should explain the criticality, and assess whether it is plausible. Discussion of possibilities for improvement or necessary measures of risk management and business continuity management may follow.
- Another approach for discussing metrics results is finding and interpreting discrepancies between *serviceAvailability*-metrics and *actualAvailability*-metrics, especially with "critical" business applications. The guiding question here can be whether these discrepancies point to failure sources different from the application landscape structure.

4.5.2. M-41: Failure Propagation Specific Proposal Comparison

Methodology Pattern Overview	
Id	M-41
Name	Failure Propagation Specific Proposal Comparison
Alias	
Summary	The methodology pattern uses metrics visualizations to compare proposals for limiting failure propagation in an application landscape regarding their realization effort and failure propagation-specific benefit.
Version	1.0

4.5.2.1. Problem Section

Methodology pattern M-41 addresses concern C-111 (see Section 4.1 for details), which deals with supporting decisions with respect to measures for limiting failure propagation.

4.5.2.2. Solution Section

The methodology pattern relies on Viewpoints V-92 and V-93, together with the respective information model patterns, i.e. I-93, together with I-90 or I-91, and most probably I-92, to provide aggregated views.

The methodology pattern distinguishes two roles in a metrics-based analysis: *Metrics experts*, analysts responsible for applying metrics on the application landscape, and *domain experts*, which use the service of the metrics experts in actually addressing the respective concern. This methodology pattern gives basic steps of a metrics-based proposal comparison, with the role driving the respective step indicated in boldface.

1. For each Proposal, a Scenario describing the application landscape, in case this Proposal is realized, has to be created. In this context, it makes sense also to provide a model of the current application landscape (as-is). On the one hand, this allows improvements to the current situation to be calculated. On the other hand, models of Scenarios can be created by applying changes to the current application landscape. Possibly, also the concern has to be refined and concretized. [metrics experts, **domain experts**]
2. The scenarios have to be analyzed using metrics which means, that metrics are calculated and collected to give answers to stakeholders' questions and to support them in deciding between the respective Proposals. Basic types of relevant questions are: *How does a Proposal change the availability of a specific element of the application landscape, and how does this affect business? How do failures in an element of the application landscape affect other elements, and thus business? What effort is connected to realizing the Proposals?* [**metrics experts**]
3. Communicating the analysis results to stakeholders relies heavily on adequate visualizations, which the metrics experts create. These visualizations should include first ideas and hints regarding the interpretation of the measurements, as these can foster the discussion in the next step. [**metrics experts**]
4. Metrics experts present the analysis results to the domain experts, in order to discuss and interpret them. This introduces the important aspect of using both formal evaluation techniques (metrics) and experts (the domain experts). The discussions can produce refined proposals, or questions, which then leads to the process being restarted at 1. [metrics experts, **domain experts**]

Subsequently, some aspects of the above evaluation process are covered in more detail:

4.5.2.2.1. Comparing Proposals

Basically, comparing Proposals is built around the metrics-based prediction of availability, failure impact, and failure costs introduced in Section 4.3. Thus, a Scenario comparison relies on the following:

- the metrics being validated, as discussed with M-40.
- the Scenarios being comparable. Consider, e.g., a model of the current application landscape, which contains all existing dependencies. Now consider a model describing a Scenario of the application landscape, which however only considers the core functional dependencies. It has to be questioned, whether it makes sense to compare values of metrics as introduced in Section 4.3 for these two models. Thus, the models to be compared should describe the application landscape at the same granularity level. This may be achieved by creating Scenarios starting with a model of the current application landscape and applying the respective changes to it.

Viewpoints V-92 and V-93 are suitable for visualizing Scenario comparisons.

4.5.2.2.2. Effects of a Proposal on Failure Propagation

Projects can improve availability aspects, and thus induce costs. The benefits of a Proposal with respect to availability can be categorized as follows:

- Improved *serviceAvailability* of specific Interfaces, which leads to an improved *averageServiceAvailability* of the application landscape and reduced failure costs.
- Reduced *failureImpact* of BusinessApplications, leading to the risk of large failures involving a considerable share of Interfaces or leading to high failure costs. This can be advantageous in risk mitigation. The effect can basically occur separately from the above one, as demonstrated by Figures 4.16 to 4.19.

Possible Measures (from I-93 , see Section 4.3.4) that can be part of a Proposal, with their failure propagation-specific benefits, are listed in Table 4.3.

4.5.2.2.3. Putting Proposals in Relation: Exploring Proposal Spaces

While Scenarios can be compared to each other using metrics results compiled and visualized as described above, indicating which one is *better* regarding specific aspects, it is more difficult to arrive at more absolute judgments, e.g., a Proposal reaches an *acceptable* improvement.

Exploring proposal spaces answers this question starting with a specific *principle* of limiting failure propagation in an application landscape. By automatically generating a large number of variations of this principle, and applying the metrics to each variation, it is possible to explore what applying the respective principle basically is able to achieve. Figure 4.21 exemplifies this, visualizing each generated Scenario of the application landscape as a circle, with the x-coordinate indicating the averageServiceAvaialbility and the y-coordinate an indicator of the effort connected to realizing the respective Scenario.

The line in Figure 4.21 connects the points forming the *efficient set* of the Scenario cloud. These points represent Scenarios, for which all Scenarios having a higher *averageServiceAvailability* are also characterized by a higher *effortIndicator*. This is for example not the case for *Proposal Y*. The rightmost point on the efficient set line reaches a better *averageServiceAvailability* at a lower *effortIndicator*. Thus, taking into consideration only the information from the Proposal cloud in Figure 4.21, only the Proposals from the efficient set are sensible, while which one of them to realize depends on the desired *averageServiceAvailability* and the amount of effort one wants to put into increasing availability.

While the Proposals of the efficient set need not be feasible at closer look, e.g., also considering functional aspects, or other quality attributes, they reveal what is possible by applying the respective principle, which is an aid in interpreting specific (possibly manually designed) Proposals. Consider, e.g., *Proposal X* in Figure 4.21, which appears near the efficient set line and seems to makes use of the principle rather well. This may be not the case for Proposal Y, which is located rather above the efficient set line, which makes it a candidate for closer examination.

Measure	**Description**	**Effect**
Decrease coupling	Remove dependencies, e.g., by removing unnecessary dependencies to basic functionality, by shortening data supply paths, or changing them to ones relying on technologies less likely to propagate a failure (e.g., change from a synchronous function call to messaging)	*serviceAvailability* improves, failureImpact is reduced; thus, likelihood of large failures may be reduced
Split basic functionality	One or more BusinessApplications serving a large number of using BusinessApplications are deployed multiple times, with each deployment only serving a subset of the using BusinessApplications	The most important effect is reduction of *failureImpact*, with the likelihood of large failures thus decreasing; *serviceAvailability* of specific Interfaces may increase in some cases
Introduce backup capabilities	BusinessApplications are redesigned to have certain Interfaces failing only if all of several used Interfaces fail	The *serviceAvailability* of the affected Interfaces improves, *failureImpact* of the BusinessApplications for which a backup now exists is reduced
Reduce application failures	Intra-application changes to make a BusinessApplication less likely to fail	An underlying reason for failures is removed
Minimize service usages	Reduce *failurePropagationProbability*-values, via redesign of BusinessApplications to use external services only if really necessary	*serviceAvailaiblity* of affected Interfaces is improved, *failureImpact* of BusinessApplications called less often is reduced
Redesign application landscape	Depending on the actual redesign	

Table 4.3.: Effort and failure propagation-specific benefit of Proposals

In using explicit information models, and relying on metrics and clearly defined visualizations, the methodologies defined above give explicit tools for addressing concerns related to failure propagation. Thus, they move the respective activities from an art more to an understood and systematic activity.

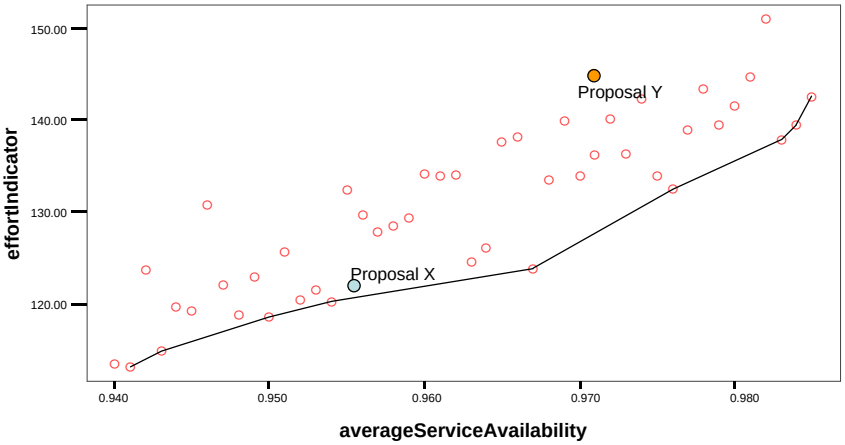


Figure 4.21.: Cloud of Scenarios automatically generated according to a specific principle and evaluated by metrics

CHAPTER 5

Validation in a Case Study

Contents

This chapter puts the metrics-based methodologies introduced in Chapter 4 into practice in a real-world case study. The goal of this case study is to demonstrate their utility, and gather experience in how to actually apply metrics to an application landscape. While this certainly focuses on the above introduced failure propagation metrics, the case study is also supposed to gather experience about metrics in general.

The steps and deliverables of the case study are presented roughly in chronological order. For reasons of confidentiality, the diagrams showing the evaluation results have the numerical values on the axes removed.

5.1. Initial Situation

The case study was conducted at a large bank, and encompassed a part of the application landscape supporting private banking, specifically the one which has its Applications located on the mainframe. This subset of the application landscape consists of 255 Applications, organized into 75 Subdomains, which are themselves organized into 18 Domains. Together, the Applications amount to about 12 million lines of PL/1 code.

The concerns to be addressed by the case study were centered on a quality attribute called *operational independence*, which the stakeholders define as follows: *Operational independence is an approach to reduce dependencies between components. Its intent is to further increase robustness and overall availability of the IT to bankers and clients.*

Specifically, operational independence influences the following:

Failure propagation Failures of an Application causing connected Applications also to fail, as discussed in Chapter 4.

Deployment of Applications Dependencies impede reusing specific parts of the application landscape on their own. Also, the dependencies aggravate deployments of new releases, decreasing flexibility as business has to wait until the next deployment cycle before it gets newly implemented functionality.

Data quality Failures in Applications can lead to data not being brought up to date, and incorrect data can be propagated across the application landscape.

Response times Failures in an Application can lead to responses being delayed (especially if message queues are involved). However, stakeholders consider this aspect to be less important, as messaging is used only where response times are not critical.

Possibilities to address these issues using metrics were discussed with the stakeholders, leading to the focus of the analyses being put on failure propagation. Besides failure propagation, other possibilities for application landscape metrics were presented to the stakeholders, which however involved some difficulties:

- Metrics examining data quality propagation were expected to need more detailed models than the available data could provide.

- Metrics examining aspects related to installability or testability also appeared to rely on more data than available, and possibly also time-consuming simulations.

Additionally, the failure propagation analyses turned out to be worth more research: Several cycles of refining concerns and deepening the analysis were conducted. Thus, the focus of the case study stayed specifically on this concern.

5.2. Introducing Failure Propagation Metrics

The first activity of the case study focused on introducing the metrics, giving stakeholders an overview of possible analyses, clarifying the concerns, and defining further analyses to be conducted.

5.2.1. Interpreting the Available Application Landscape Data

To create an initial metrics analysis, the case study proceeded along M-40.

As a basis for the metrics analyses, the stakeholders provided a dataset describing the above-mentioned subset of the application landscape. The data were structured as shown by the information model in Figure 5.1, and have been collected by automatically parsing the code deployed on the mainframe.

The stakeholders define the concepts introduced in Figure 5.1 as follows:

Domain A business Domain represents a coherent set of capabilities and responsibilities. It is an element of the functional decomposition of the banking business functions in the context of the service landscape. Business Domains are linked to certain skills and knowledge, which are clearly identifiable in the banking business. A business Domain belongs to exactly one business area and can be subdivided into Subdomains if needed.

Subdomain A Subdomain groups functionality, which is similar in business terms.

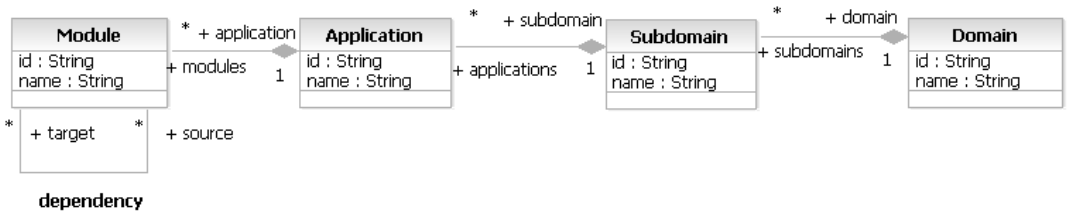


Figure 5.1.: Information model of the data initially available for the case study

Application A deployed group of Modules, which is executed in one process at runtime. This process may terminate due to several kinds of failures, with the Application no longer being able to render its services. These kinds of failures include the following:

- Errors in linking the Application at its deployment
- Errors in memory management
- Transaction loads too high for the Application to handle
- Divisions by zero

Module A specific file into which the source code is organized in PL/I.

In the examined subset of the application landscape, basically, four types of dependencies are used:

- Synchronous function calls,
- messaging, i.e. asynchronous information exchange via message queues,
- bulk data transfers via file exchange, and
- database-facilitated dependencies, with two or more Applications accessing a shared database.

The dataset contained only synchronous function calls, which are, however, the ones the stakeholders expected to be most critical with respect to failure propagation.

Comparing the information model presented in Figure 5.1 to I-90 (Section 4.3.1), or I-91 (Section 4.3.2), one recognizes *"Applications"*, which on first sight are similar to the *BusinessApplications* of I-90/I-91. However, the Applications have no Interfaces. Instead of this, *"Modules"* with dependencies due to synchronous function calls between them appear below the Applications. The Domain structure resembles I-92, with the difference of being organized into two hierarchy levels, Domains and Subdomains.

As described by [LS08], some variants of interpreting the data were tried in order to find an adequate interpretation. The kind of information available from the dataset easily hints at I-90 as the more adequate information model, as, e.g., information about different kinds of interface technologies, connection availabilities, or loss functions is not included.

The first interpretation consisted of taking the *"Applications"* in the available data as the *BusinessApplications*. The interpretation assumed that all services rendered by an Application fail (leading to the Application failing totally), if at least one of the Applications it depends on (as it uses, also transitively, a Module of this Application) fails. However, these assumptions turned out to be overly conservative with respect to how failures propagate. In spite of setting A to rather optimistic values, the metrics values were unrealistically pessimistic.

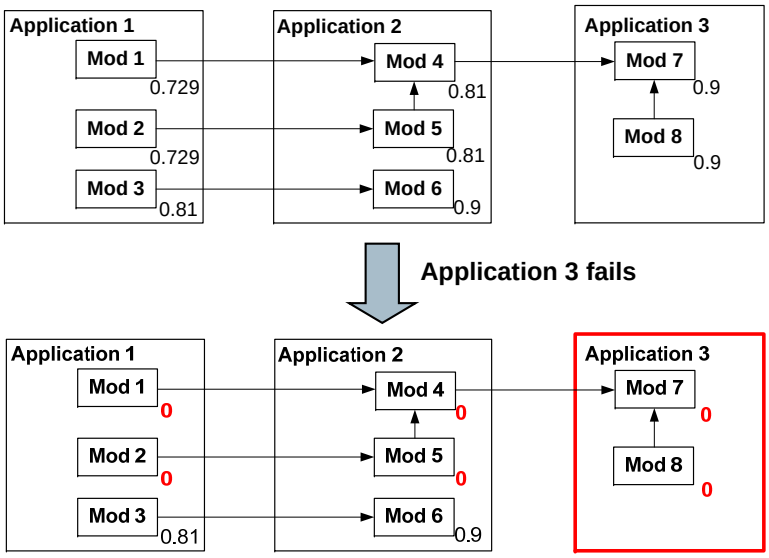


Figure 5.2.: Calculating serviceAvailability and failureImpact based on the case study data

The second interpretation took the Modules as BusinessApplications offering one kind of service (via a respective interface). It assumed that a Module fails if at least one of the Modules it depends on fails. Difficulties connected to this interpretation were as follows:

- It yielded only reasonable results, if the module-specific A was set rather high, and
- it is, as also opined by the stakeholders, more reasonable to consider Applications, and not the units in which their source code is organized, as the elements having the availability.

Thus, a more sophisticated interpretation was used, which relies on interpreting the Applications as BusinessApplications, and assumes that the functionality of each Module is offered via a (module-specific) Interface. These assumptions identify Modules with Interfaces, Application internal dependencies as the failure tree structure, and inter-application dependencies as Connectors.

This implies that *serviceAvailability* and *failureImpact* are calculated as exemplified in Figure 5.2. The arrows signify the *dependency* relationship, with the arrowhead indicating the *target-Module*.

This calculation assumes that a Module works, iff the Application the Module belongs to and all Modules it depends on work. This leads, e.g., to Mod1 relying on Applications 1, 2, and 3 being operational. This happens with a probability of 0.9^3, assuming, as described with I-90, independent failures of the Applications and $A = 0.9$. The remaining values annotated at the Modules in the upper part of Figure 5.2 have been calculated similarly.

failureImpact is then calculated as described in I-90 (Section 4.3.1.1.2): It is the average deterioration of serviceAvailabilities at the Modules, assuming that the Application under consideration failed. In case of Application 3, the *failureImpact* is $\frac{1}{8}(0.729 + 0.729 + 0 + 0.81 + 0.81 + 0 + 0.9 + 0.9) = 0.60975$.

Each Module is assumed to offer functionality due to the following reasons:

- Each Module might provide functionality directly used by processes (which ones actually do, is not available from the data; thus each Module is considered as an approximation).
- The Modules not externally used might hint that the Modules using them provide more functionality.

The actual calculations in the case study were performed with $A = 0.99$.

5.2.2. Tool Support

Calculating metrics as described above for the application landscape targeted by the case study relied on a prototypical tool, of which Figure 5.3 outlines[1] the basic architecture.

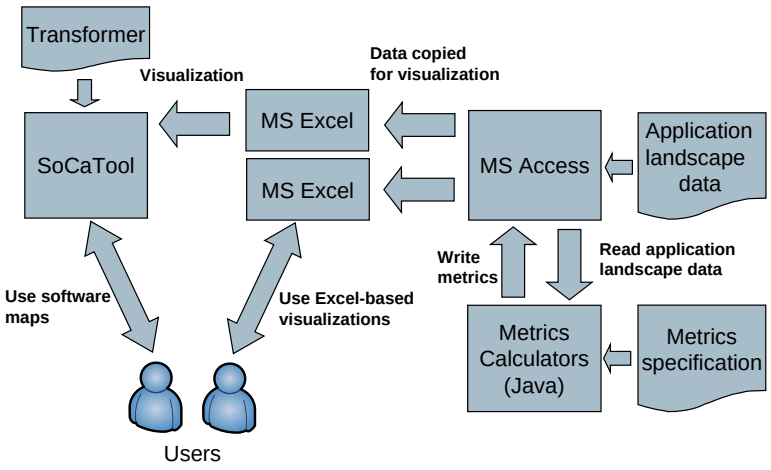


Figure 5.3.: Tool support used for calculating metrics

The *Metrics Calculators*, realized as standalone Java applications, read data describing the application landscape from an Access database, calculate the metrics values, and write them back into the database. Each metric is specified as a Java class containing the calculation procedure. For visualization, the application landscape data including the metrics values is copied into Excel spreadsheets. There, they can be visualized using the diagramming capabilities native to Excel, or constitute input data for software map generation using the SoCaTool. Each kind of software map produced by the SoCaTool is thus realized as a transformer implemented in Java [Bu07b].

[1] As the tools were throwaway prototypes, detailed and formalized diagrams, as well as actual source code, are not given due to reasons of brevity.

Algorithmically, the calculation procedure for the *serviceAvailability*-metric relies on performing a depth-first search [Se92] on the directed graph made up by the Modules and their dependencies. By deriving the set of Applications containing the Modules reachable by navigating along the dependencies from a depending Module to the used Modules, one gets the Applications that need to be operational for a specific Module to be operational. According to the above assumptions, *serviceAvailability* is then A^n, with n being the cardinality of this set. The application-specific *averageServiceAvailability*-values were then derived by a view in the database that stores the metrics values.

The algorithm for calculating *failureImpact* collects all Modules that are affected by a failure in the respective Application. This is achieved by taking each Module of this Application as a starting point for traversing the graph, following the dependencies in the opposite direction. A failure in the examined Application drops the *conditional serviceAvailailabilities* of the collected Modules to zero. Thus, *failureImpact* is the sum of the ex ante *serviceAvailabilities* of these Modules, divided by the total number of Modules in the application landscape.

5.2.3. Analysis Results and Feedback

A software map was used to display metrics calculated as described above (see Figures 5.4 and 5.5). The map basically follows V-91 (Section 4.4.2), but differs in some details:

- Metrics are not shown for Domains, but for Applications. Due to this, and the two-fold Domain structure of the case study data using also Subdomains, the map shows three levels of hierarchy.
- Instead of a size measure, *averageServiceAvailability* is mapped to the size of the symbols in the lowest hierarchy level. The coloring of these symbols is used to indicate *failureImpact*. This was done to support spotting critical Applications swiftly, as they appear on the map red and large.

After presenting the approach to stakeholders, they decided to employ it for comparing the as-is situation to two possible scenarios of the application landscape, which have been designed to improve operational independence. The possibility of swiftly spotting critical Applications was not used by the stakeholders.

5.3. Comparing Proposals with Respect to Failure Propagation

According to the wishes the stakeholders expressed after their first encounter with metrics for application landscapes, two proposals created by them relying on intuition were compared using metrics as described in M-41 (Section 4.5.2).

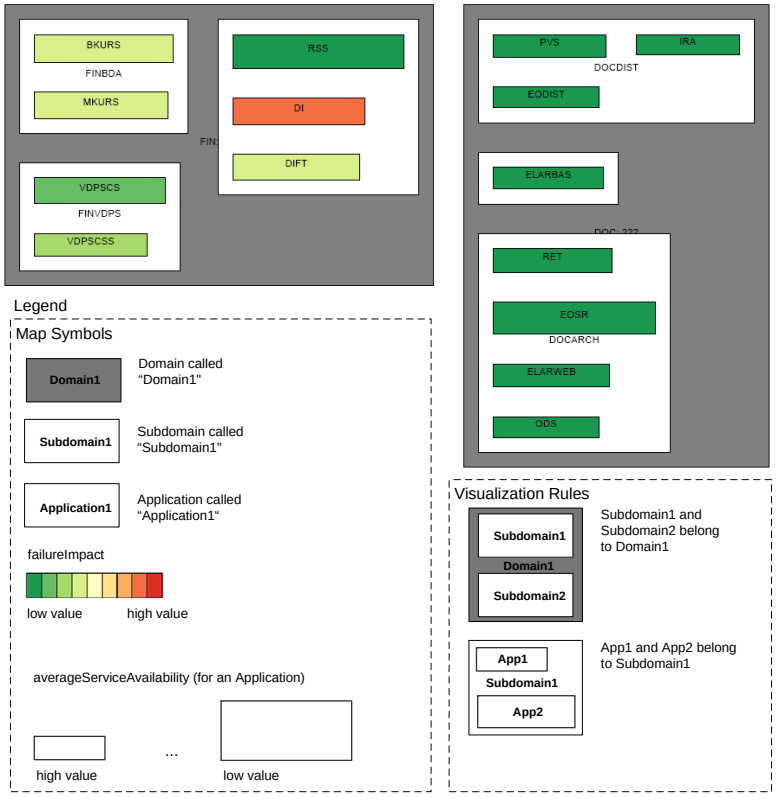


Figure 5.4.: failureImpact and serviceAvailability in the case study (excerpt)

5.3.1. Introducing Proposal I and Proposal II

Both proposals rely on making changes to how Domains are deployed and communicate. Thus, Figure 5.6 introduces the Domain structure of the application landscape subset under consideration. For the proposals, the Domains are organized into so-called Domain clusters along functional concerns: *Fundamentals*, providing basic data and services, *Money Business*, *Asset Business*, *Interfaces*, realizing the interfaces offering functionality to customers and suppliers, and *Complementary*, which contains non-banking functionality.

In the as-is landscape, one platform hosts all Domains. Contrastingly, the proposals distribute the Domain clusters to different platforms, as illustrated by Figure 5.7. In doing so, the Fundamentals-cluster is replicated in each platform. Thus, each platform hosts the Fundamentals-Domains, and another Domain cluster. The platforms are independent; therefore, only asynchronous communication is allowed between them. The data used by the Fundamentals-Domains is replicated between their different deployments by a mechanism not further specified by the proposals, which was thus not further evaluated in the analysis. Only one deployment of a Fundamentals-Domain is allowed to change its data; the other deployments are restricted to read-only access.

The two proposals differ in the number of platforms they create and how they distribute

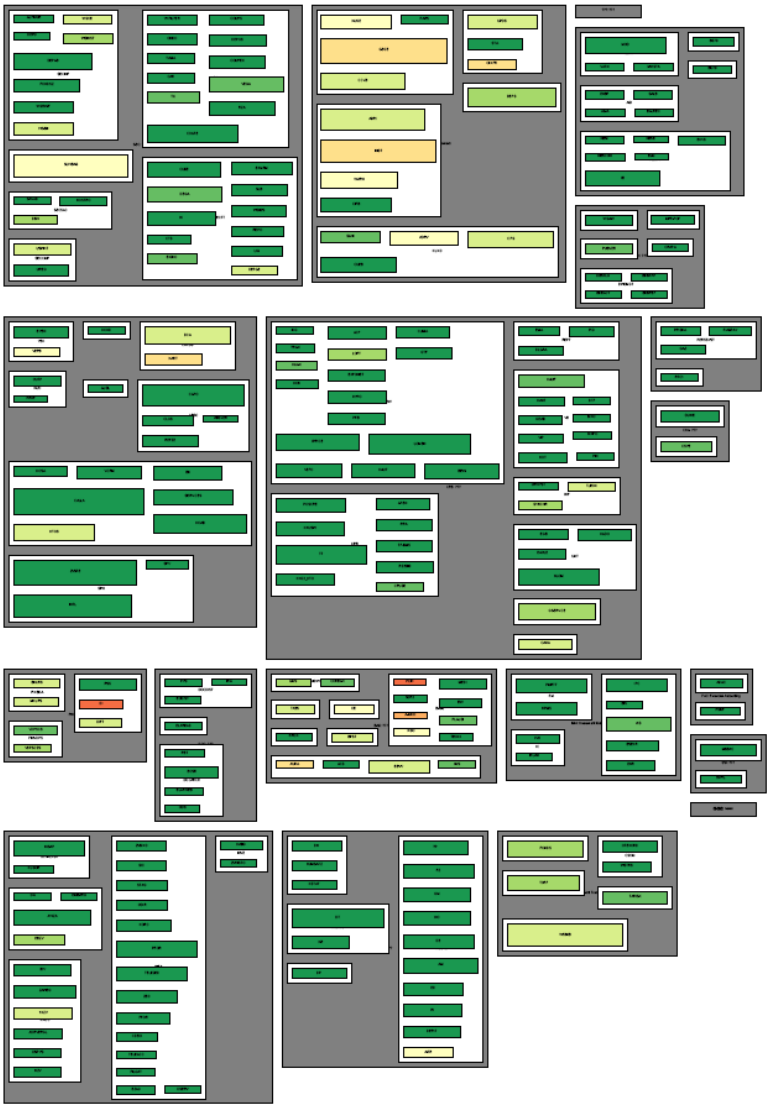


Figure 5.5.: failureImpact and serviceAvailability in the case study (small print for confidentiality reasons)

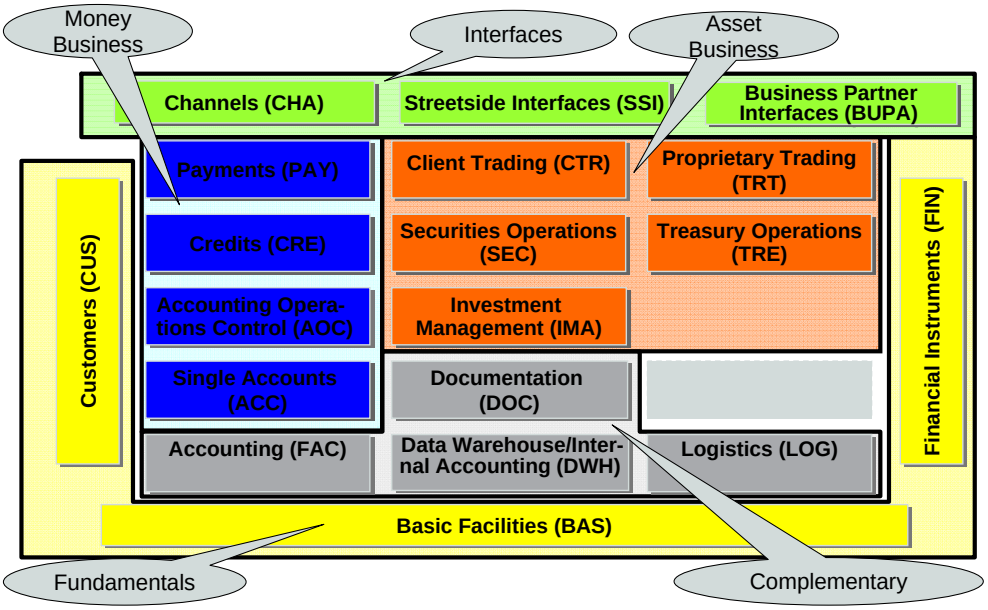


Figure 5.6.: Domains grouped to Domain clusters

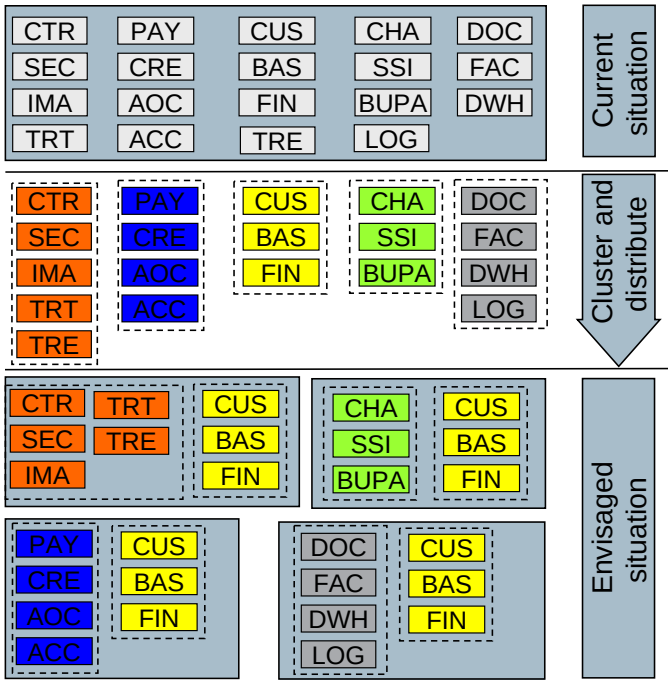


Figure 5.7.: Basic idea underlying Proposal I and Proposal II

the Domains on them. Figure 5.8 shows Proposal I, with four platforms on which it distributes the Domain clusters introduced above. Figure 5.9 introduces Proposal II, which uses an additional platform called *Trading*. The Fundamentals-Domains having write access to their respective data are identified by a thick border in Figures 5.8 and 5.9.

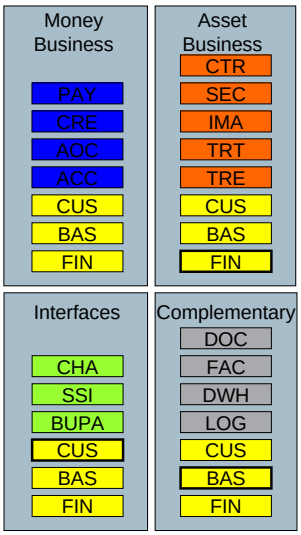


Figure 5.8.: Proposal I

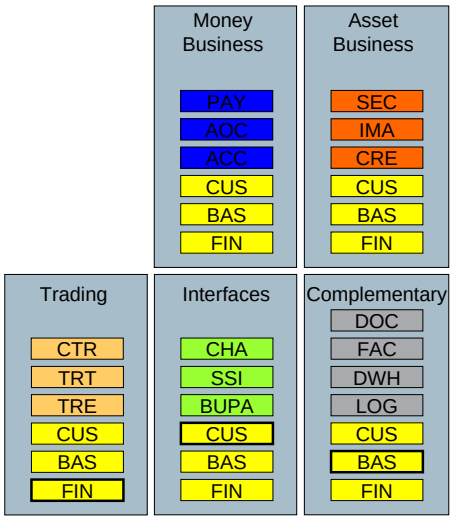


Figure 5.9.: Proposal II

5.3.2. Calculating Metrics to Compare the Proposals

As described in M-41, the proposal comparison focuses on comparing the proposals introduced above with respect to their ability to limit failure propagation, and the effort necessary to realize them.

5.3.2.1. Assumptions in the Metrics-based Proposal Comparison

Several assumptions are necessary to use the information about the application landscape and the two proposal introduced in Section 5.3.1 as a basis for metrics allowing a meaningful comparison. These assumptions are used to prepare the data describing the as-is application landscape and the proposals.

5.3.2.1.1. Changes to the As-is Landscape/Asynchronous Dependencies

The application landscape data already used for the evaluations in Section 5.2 constitute the basis for calculating the metrics on which the proposal comparison is based. Two datasets, each describing one of the proposals, are derived from this data set by making the following changes to the as-is data:

- The platforms are added, together with the information, which Domain is deployed on which platform, including the multiple deployments of Fundamentals-Domains, which are specifically marked by a flag.
- Dependencies between platforms are removed. The proposals basically assume that they are replaced by asynchronous dependencies. However, the analysis neglects asynchronous dependencies, assuming that
  - their contribution to failure propagation is insignificant.
  - neglecting them, while possibly affecting absolute values, does not basically change the results of comparing the different scenarios of the application landscape.
- Dependencies from a non-Fundamentals-Domain to a Fundamentals-Domain always point to the Fundamentals-replication in the respective platform.

5.3.2.1.2. Domains Not Considered in the Proposals

Two Domains, BUPA and TRT, are not considered in the proposal comparison, as they have no or only a small amount of code on the mainframe. Thus, the available data about the application landscape do not describe these Domains.

A modification preparing the data describing the as-is landscape as well as the proposals for the comparison concerns Domain TAP, which was basically contained in the as-is data, but is not considered by the proposals (see Section 5.3.1), as it contains no banking but technical Applications, e.g., monitoring tools. Therefore, the proposal data cannot consider this Domain, and to ensure comparability, it was removed from the as-is data as well. This was deemed possible, as the focus of the analysis rests on comparing the proposals, and not on the absolute metrics values, and removing the Domain is assumed to affect each of the scenarios to be compared roughly equally.

5.3.2.1.3. Effort Estimation

The proposal effort estimation relies on the count *removing or changing dependencies* from I-93 (Section 4.3.4). An effort estimation per platform uses the above introduced assumption that each dependency crossing a platform border is replaced by asynchronous communication.

Therefore, each changed dependency is both counted at both the platform of the source and the target Module: Changing the dependency is assumed to need establishing a message queue at the target Module, and changing from a function call to sending a message at the source Module. Thus, each dependency is counted, even if several dependencies target the same Module, which might thus only require introducing one message queue. Counting these dependencies is supposed to reflect that providing a shared message queue might induce an increased effort.

5.3.2.2. Results of the Metrics-based Proposal Comparison

After deriving, as described above, the datasets for the as-is landscape and the scenarios created by the two proposals, the metrics calculator already used in the initial metrics calculation (see Section 5.2) was applied. Views according to V-92 and V-93 presented the results to the stakeholders.

Figures 5.10 and 5.11 compare Proposal I and the as-is landscape with respect to averageServiceAvailability. Figure 5.10 shows the improvement of *averageServiceAvailability* reached by the Domain deployments of Proposal I. These deployments are listed on the x-axis, labeled as *<Domain><Platform Domain is deployed on>*. The curve for the as-is landscape displays the *averageServiceAvailability* for the corresponding Domain in the as-is landscape, leading to the curve having the same value at some points, due to the replicated Domains.

Figure 5.11 shows the same information on a software map. It visualizes the application landscape created by Proposal I, coloring each deployment of a Domain to indicate its improvement in terms of *averageServiceAvailability* over the respective Domain in the as-is landscape. The more full the green coloring, the higher the improvement reached by the respective deployment. Besides this, the software map also includes the effort indicator described in Section 5.3.2.1.3, visualizing the effort necessary in each platform as a red circle.

Figures 5.12 and 5.13 visualize effort and improvement of *averageServiceAvailability* connected to Proposal II in a similar way. In this context, the total effort for realizing Proposal II (the sum of the effort indicators visualized by the red circles in Figure 5.13) surpasses the effort connected to Proposal I.

Figure 5.14 offers a direct comparison of Proposal I, Proposal II, and the as-is landscape. To allow a V-92-based visualization, the diagram uses average values for the multiple deployments of the Fundamentals-Domains in the graphs for Proposal I and Proposal II.

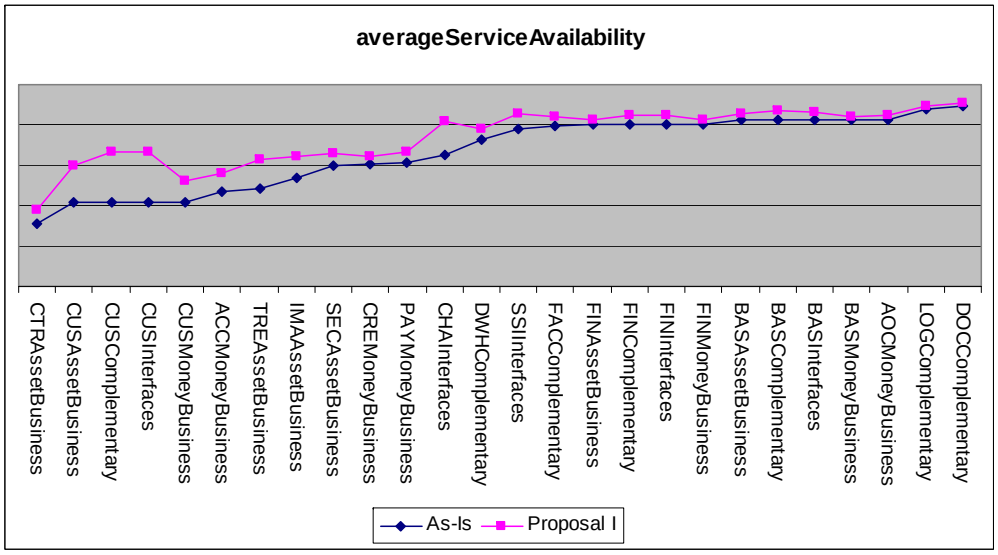


Figure 5.10.: Comparing Proposal I to the as-is landscape (V-92)

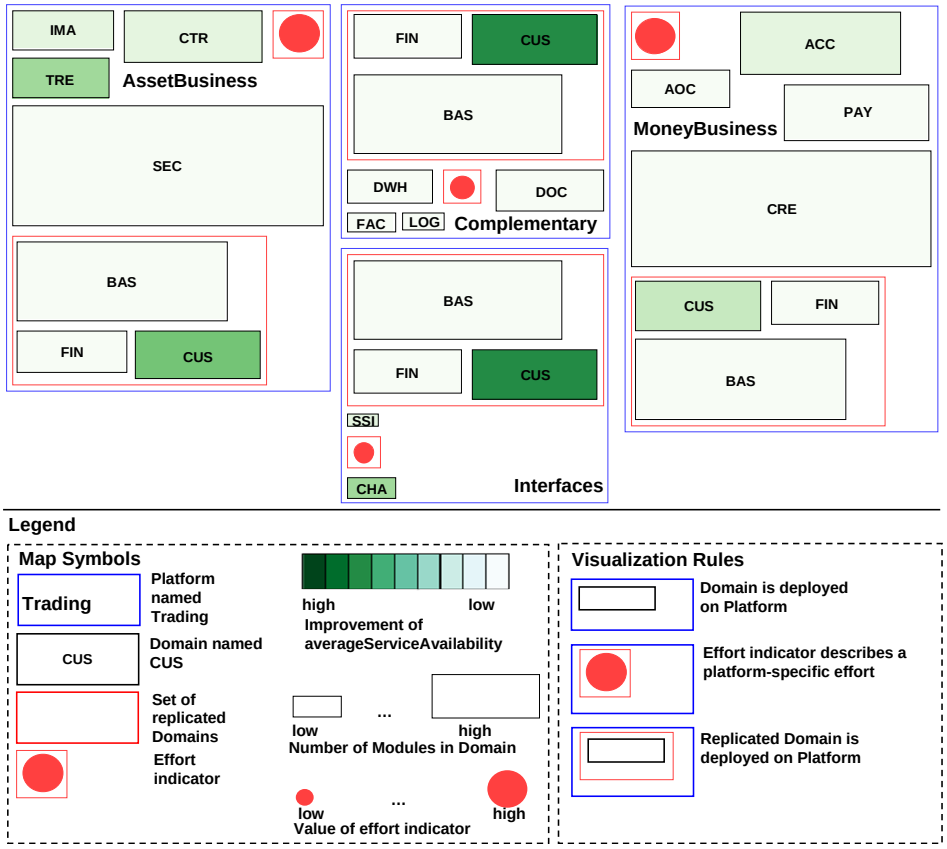


Figure 5.11.: Comparing Proposal I to the as-is landscape (V-93, including change efforts)

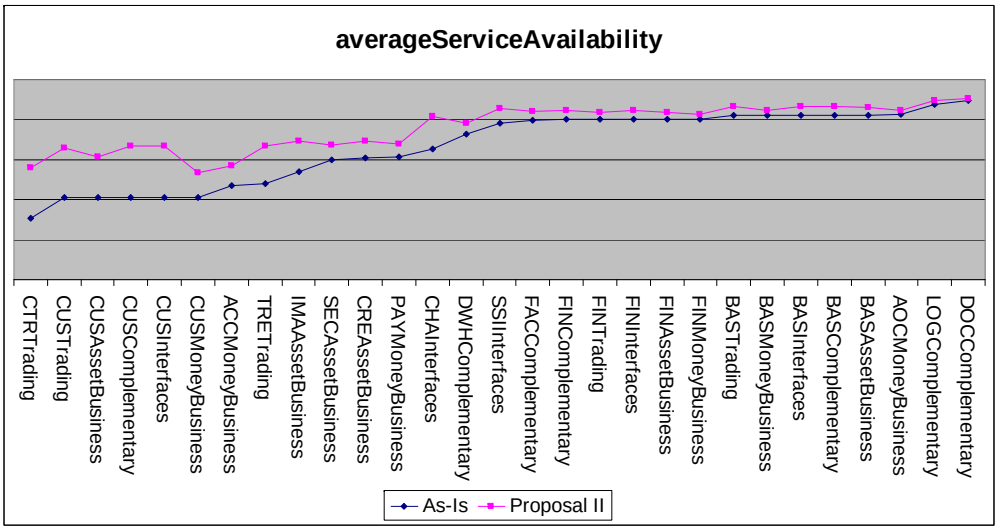


Figure 5.12.: Comparing Proposal II to the as-is landscape (V-92)

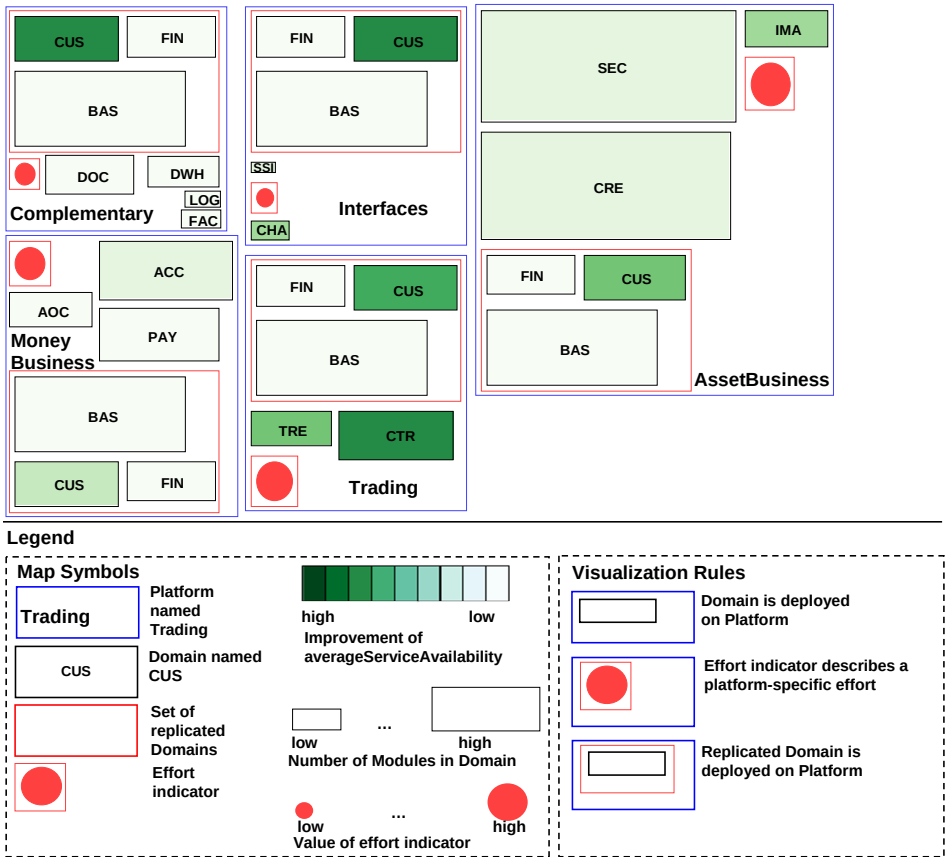


Figure 5.13.: Comparing Proposal II to the as-is landscape (V-93, including change efforts)

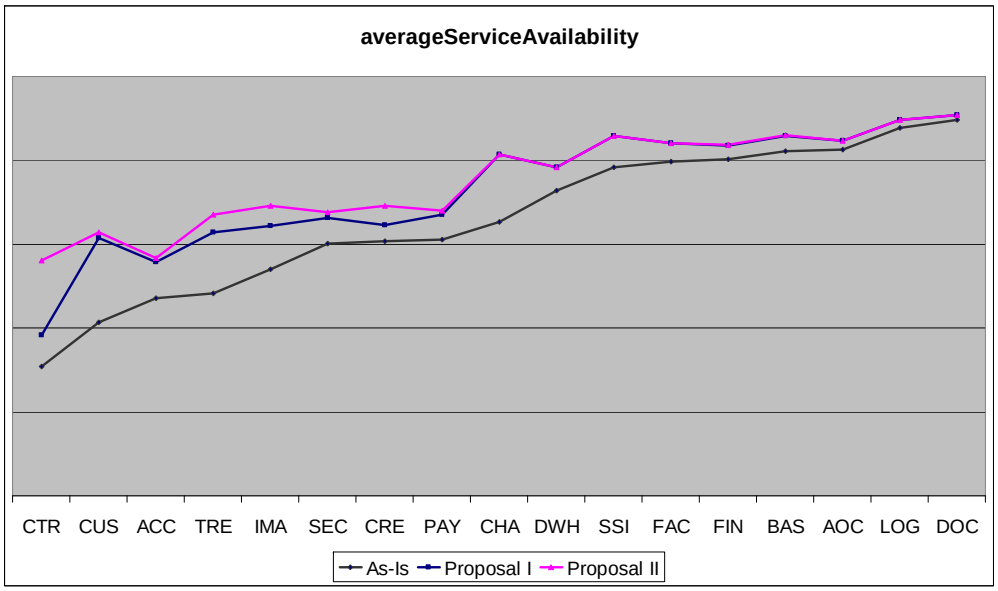


Figure 5.14.: Proposal I, Proposal II, and the as-is landscape

5.3.3. Exploring Options of Distributing Domains on Platforms: Proposal Generator

The above evaluations predict that both proposals increase availabilities, and that Proposal II, with higher ability to limit failure propagation, also requires a higher implementation effort. However, judging whether the proposals are "good" is still difficult. In this respect, M-41 proposes *exploring proposal spaces* (see Section 4.5.2.2.3). Consequently, the case study explored, how, and at which effort the *averageServiceAvailability* of the application landscape can be improved by proposals distributing the Domains on four independent platforms, as done by Proposal I.

5.3.3.1. Supporting Proposal Space Exploration with a Tool

The tool exploring the set of possible proposals distributing the non-Fundamentals-Domains on four platforms followed the architecture illustrated by Figure 5.15.

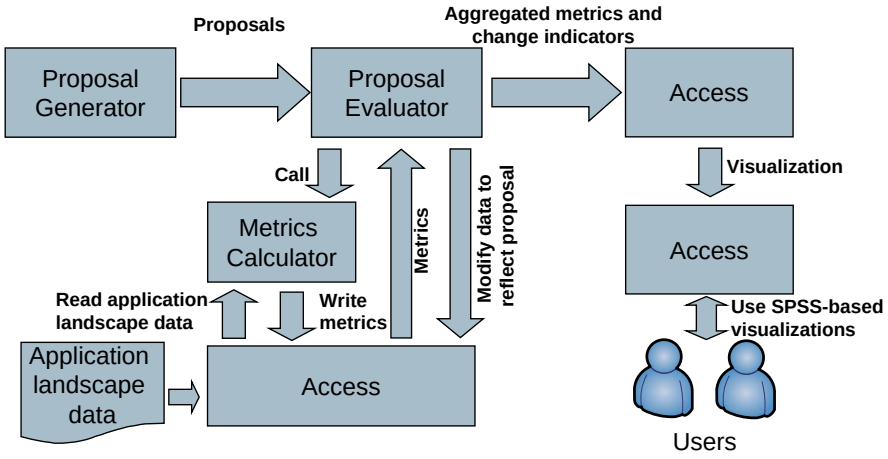


Figure 5.15.: Architecture of the Proposal Explorer

A *Proposal Generator* creates the (hypothetically) possible proposals, each of which the *Proposal Evaluator* evaluates. The Proposal Evaluator in turn relies on the Metrics Calculator already introduced in Section 5.2.2. After a generated proposal has been processed, the Proposal Evaluator stores the following results in an Access database:

- Application landscape wide *averageServiceAvailability* (subsequently called *landscapeWideAverageServiceAvailability*)
- Number of changed ingoing and outgoing dependencies per platform
- *averageServiceAvailabilities* of the specific deployments of the Domains per proposal

The proposals generated as described above can be seen as partitions of the set of $n = 14$ non-Fundamentals-Domains into $k = 4$ non-empty sets. Thus, the number of possible

proposals corresponds to the stirling numbers of the second kind [We08], leading to $S(14, 4) = 10,391,745$ possible proposals.

Thus, evaluating all proposals would have resulted in an unrealistically high computation effort. Therefore, the Proposal Generator evaluated only a sample of them. Appendix C.1 elaborates in more detail on algorithmic details of the Proposal Generator.

5.3.3.2. A Generated Proposal Space

Using the Proposal Generator, a sample of 6151 generated proposals was evaluated. Figure 5.16 displays this sample as a *proposal cloud*.

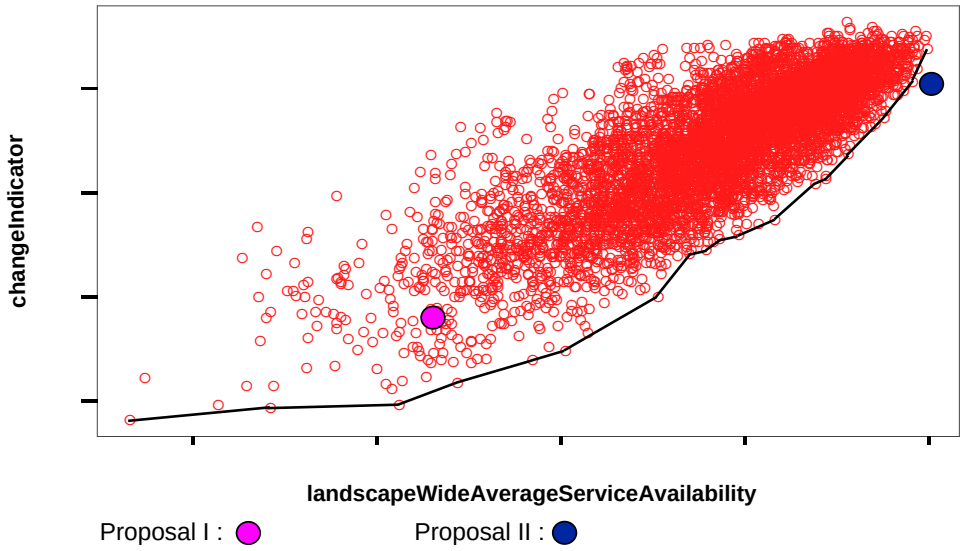


Figure 5.16.: Position of Proposal I and Proposal II in a set of automatically generated proposals

Figure 5.16 shows each automatically generated and evaluated proposal as a red circle, with the x-coordinate indicating the *landscapeWideAverageServiceAvailability*, and the y-coordinate the sum of the per platform counts of dependencies changed by the proposal (see Section 5.3.2.1.3). The two filled circles indicate Proposal I and Proposal II. The black line connects the proposals in the efficient set of the generated proposal cloud.

Both proposals appear relatively close to the efficient set of the cloud of generated proposals. Proposal I is not too far from the efficient set; there are not many generated proposals dominating Proposal I, compared to the size of the sample. This supports the assumption that the proposal uses the potential of "distributing Domains on independent platforms" rather well. Especially, it has to be considered that the automatically generated proposals forming the efficient set are not necessarily sensible from a functional point of view, or optimal with respect to attributes other than effort and availability.

Proposal II appears slightly below the efficient set. However, it is not part of the proposals using four platforms, but relies on five platforms.

5.3.4. Interpreting the Analysis Results

The proposal comparison, including the results from the proposal generator, have been presented to and discussed with the stakeholders, obtaining feedback about various aspects of the metrics-based proposal comparison.

5.3.4.1. Information Giving Decision Support to the Stakeholders

The stakeholders gained an improved overview of their proposals from the evaluation results. The stakeholders did not expect the different Domains to benefit as widely differently from the proposals (in terms of *averageServiceAvailability*) as apparent from the evaluation. Moreover, the stakeholders expected a higher increase in availability than indicated by the model, especially for the Domains SEC and CRE.

This demonstrated that proposals for increasing operational independence should be evaluated carefully before spending considerable effort on realizing them. Discussing why the proposal evaluation did not indicate higher improvements, the stakeholders pointed to an approach for limiting failure propagation not considered by Proposals I and II: removing dependencies to "Technical Platform-Modules". These are Modules providing basic functionality (e.g., printing or fractional arithmetic). However, these are distributed over quite different Applications, which thus can affect large parts of the application landscape when failing. Therefore, the stakeholders view an improved solution of providing these basic functionalities as an important step towards limiting failure propagation, which is however not considered by the above evaluation.

5.3.4.2. Discussing and Improving the Analysis

The stakeholders considered it a key finding of the case study that projects need improved data about the application landscape. They limited this statement not only to the above proposal comparison but also directed it at (IT-) projects in general.

Another discussion point related to the assumption that dependencies crossing platform borders can be replaced by messaging, and then be considered having no impact on availability. This assumption can be disputed. If a dependency reads data, the need for these data does not disappear when messaging is introduced. If the data are not returned within a specified time, this still has to be considered a failure. However, as the dependency information does not contain the directions of the data flows, the above evaluations are still used as an (possibly optimistic) approximation.

The stakeholders also suggested a way of extending the analysis by also considering Applications and dependencies not located on the mainframe platform. Some of these Applications are highly relevant to business, and there might be possibilities to estimate how much money they earn, thus being able to put a monetary value behind availability at these Applications.

Lastly, the above evaluation did not consider an effect M-41 lists for both *decreasing coupling* and *splitting basic functionality*: reducing the likelihood of large failure events, in the context of the case study ones affecting a high share of Modules. Altogether, this led to another evaluation step, refining the analysis presented in this section.

5.4. Refining the Proposal Comparison

Addressing the stakeholder feedback to the proposal comparison (Section 5.3.4.2), this section starts with extending the cloud of automatically generated and evaluated proposals to consider another approach to limiting failure propagation. Then two of the refined questions that were part of the feedback, are addressed: the issue of the Technical Platform-Modules, and the probability of large failures.

5.4.1. Creating an Additional Proposal Set: Failure Propagation Optimizer

The feedback presented in Section 5.3.4.1 mentioned that the two proposals did not improve the metrics values as expected. However, the proposals take a good position within the generated proposal cloud introduced in Section 5.3.3.2.

Exploring another approach for limiting failure propagation allows more meaningful comparisons than the above-mentioned scenario cloud alone. Thus, another set of proposals is created based on an optimization approach, which removes dependencies between Domains where this measure improves the landscape-wide *averageServiceAvailability* per removed dependency best. This approach proceeds as follows:

- Start with a set of pairs consisting of a source Domain and a target Domain (initialized with all possible pairs).
- Take each element of this set of pairs and try removing the dependencies from the respective source to the respective target Domain; select the pair with the highest per-dependency improvement of the landscape-wide *averageServiceAvailabilty*.
- Remove the dependencies between the selected source and target Domain permanently; remove the respective pair from the set of pairs to try.
- Resume at step 1 (using the remaining pairs).

Implementing and applying this algorithm to the as-is model of the application landscape led to results shown in Figure 5.17. Proposals III and IV are introduced to reference these generated proposals in more detailed comparisons. The axes in Figure 5.17 are the same as in the original proposal cloud introduced in Section 5.3.3.2.

Proposals III and IV are, regarding the characteristics on the diagrams axes, strictly better than Proposals I and II, respectively. However, it has to be noted that the two approaches for limiting failure propagation contrasted are not mutually exclusive. On the contrary, the generated proposals removed a considerable number of dependencies

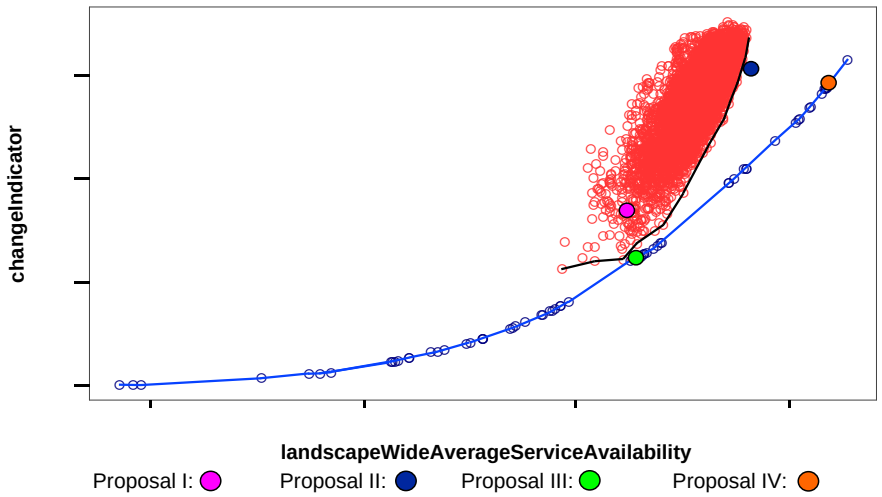


Figure 5.17.: Results of the Failure Propagation Optimizer, contrasted with the sample from the proposal space based on distributing Domains on four platforms

to Technical Platform-Modules, pointing to their importance in limiting failure propagation.

5.4.2. Technical Platform-Modules

Both the feedback to the proposal comparison in Section 5.3.4.1 and the results of the optimizer in Section 5.4.1 show that it is important to consider the Technical Platform-Modules in limiting failure propagation.

These are Modules providing basic functionality, e.g., printing or calculating with fractions. As these are contained in specific Applications, in some cases in the ones where the functionality happened to be implemented first, and are now used all over the application landscape, they lead to dependencies without any functional reason.

Failure propagation via these dependencies can be prevented by moving these Modules into a *basic library*. Technically, this means that they are deployed as a separate entity, in a way that they are accessible from all Applications, with the functionality provided similarly to operating system functionality. According to the stakeholders, this entity can be assumed not to fail.

5.4.2.1. Identifying the Technical Platform-Modules

The problem of deploying Technical Platform-Modules as a basic library is that currently the stakeholders have no complete overview about which Modules are Technical Platform-Modules. Thus, the case study employed a heuristic for identifying them, based on stakeholders' assumptions:

- Technical Platform-Modules are likely to be called from many other Modules, resulting in a high indegree in the call graph.
- About 1.5% of the Modules in the application landscape are Technical Platform.

Thus, the 1.5% of the Modules with the highest indegree were selected. Of these, the stakeholders checked a subset of 78 Modules. Fourty-five were actually Technical Platform, while the remaining 33 were false positives.[2]

5.4.2.2. Considering Technical Platform-Modules in the Proposal Comparison

The first decision in considering the Modules identified as described above concerns how to handle Modules called by one of the Technical Platform-Modules. On the one hand, one could classify all Modules called by a Technical Platform-Module also as a Technical Platform-Module, and re-apply this procedure, until no additional Modules are re-classified.

However, the case study makes a more conservative estimation. It takes only the Modules as Technical Platform, for which this has been actually confirmed by the stakeholders, assuming that their calls to other Modules have to be removed when the Modules are put into the basic library.

The second decision is that for calculating serviceAvailability-metrics, the Modules moved to the basic library are removed from the model on which the metrics are calculated, as the basic library is not assumed to fail.

Ten of the 45 Modules identified as Technical Platform above need not be removed, as they are in Domain TAP, which is not considered by Proposals I and II (see Section 5.3.2.1.2). Thus, a set of 35 Modules was removed from the model of the as-is landscape, and the corresponding deployments were removed from the models describing Proposal I and Proposal II. The dependencies originating from or pointing to these Modules were also removed.

5.4.2.3. Results of Considering Technical Platform-Modules

Modifying the models describing Proposal I and Proposal II, as described above, resulted in the evaluation results shown in Figure 5.18, which is an extended version of Figure 5.14.

The diagram reveals that both proposals benefit equally largely from moving only 35 Technical Platform-Modules into a basic library. The proposal cloud in Figure 5.19, which also shows the new versions of Proposal I and Proposal II, confirms this. The new versions are displayed in this diagram with an effort indicator increased by 35, as 35 Modules are expected to need some adaptations when moving them into the basic library.

[2] When randomly selecting 78 Modules, $78 \cdot 0.015 = 1.17 \approx 1$ correctly identified Technical Platform-Module can be expected at 1.5% Techical Platform-Modules in the application landscape.

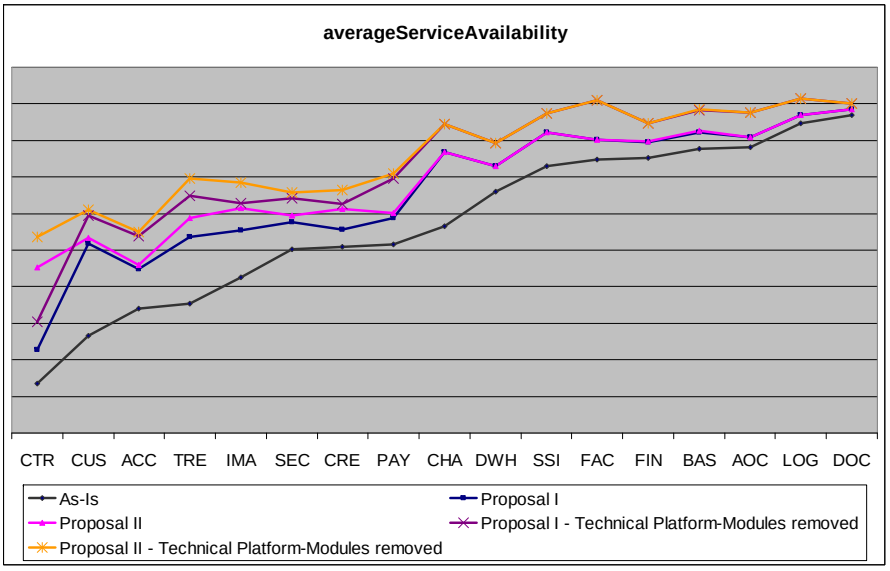


Figure 5.18.: Proposal I and Proposal II benefit largely from removing Technical Platform-Modules

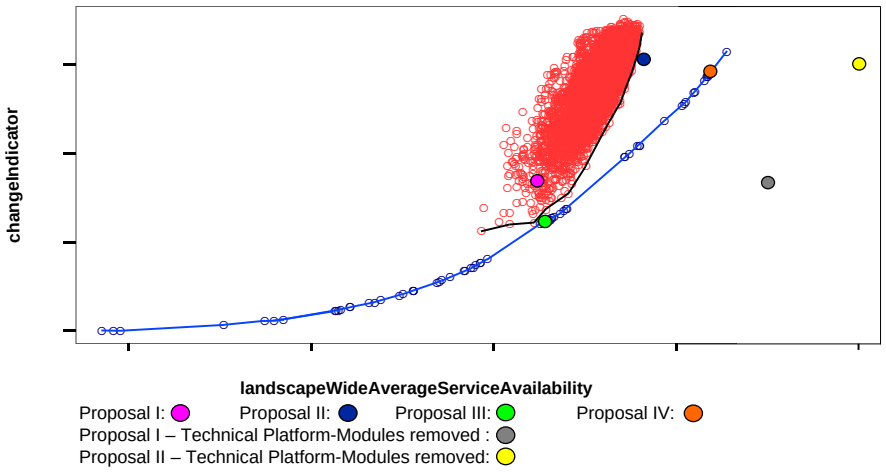


Figure 5.19.: Benefit from removing Technical Platform-Modules summarized

However, while removing only 35 Technical Platform-Modules led to a considerable improvement of *averageServiceAvailability* metrics, one can assume that these Modules are the most effective ones to remove: The removed Technical Platform Modules had the highest indegree, i.e. the highest numbers of dependencies pointing to them.

5.4.3. Estimating the Probability of Large Failures: Modules at Risk

Besides the Technical Platform-Modules, the probability of large failures constitutes another issue, where the analysis from Section 5.3 needs an extension. Thus, the case study estimated failure distributions and derived *Modules at Risk*-values, similar to the *Interfaces at Risk* outlined by I-92 in Section 4.3.3.1.2. Thus, the analysis is meant to more completely capture the benefit related to Proposals I and II. This is especially important, as the measure underlying these proposals, *splitting basic functionality*, has the *reduced likelihood of large failures* as its main benefit.

5.4.3.1. Calculation of Modules at Risk and Distributions of Failing Modules

As the metric calculations in this case study rely on interpreting the Modules as Interfaces (Section 5.2), this analysis calculates not an *Interfaces at Risk*, but a *Modules at Risk*-value, subsequently denoted as MaR_α. Similarly to the Interfaces at Risk, IaR_α, MaR_α denotes the α-quantile of the distribution of failed Modules.

Fully calculating the distribution of failed Modules, in order to derive MaR_α-values, is however not feasible, as it leads to a prohibitively high computation effort. Working through all combinations of failing n Applications as described in Appendix C.2.1 is of runtime complexity $O(n2^n)$, making a total evaluation prohibitively computation intensive.

Thus, the failure distribution is estimated based on a Monte Carlo simulation, as outlined in Appendix C.2.2, for the as-is landscape, Proposal I, and Proposal III respectively.

5.4.3.2. Resulting Failure Distributions and Modules at Risk

Figures 5.20, 5.21 and 5.22 show the distributions of failed Modules as histograms.

Comparing the three distributions, also by deriving and contrasting MaR_α-values for several $\alpha \geq 0.95$ shows[3] that Proposal I is able to reduce the probability of large failures. Proposal I also reduces the probability of large failures more than Proposal III, which has a comparable landscape-wide *averageServiceAvailability*, but does not rely on splitting basic functionality. This relativizes the fact that Proposal III reaches a similar landscape-wide *averageServiceAvailability* as Proposal I, demanding much less expected change effort. Proposal I reaches an advantage over Proposal III as it reduces the probability of large failures.

[3] Actual values are not included here for reasons of confidentiality.

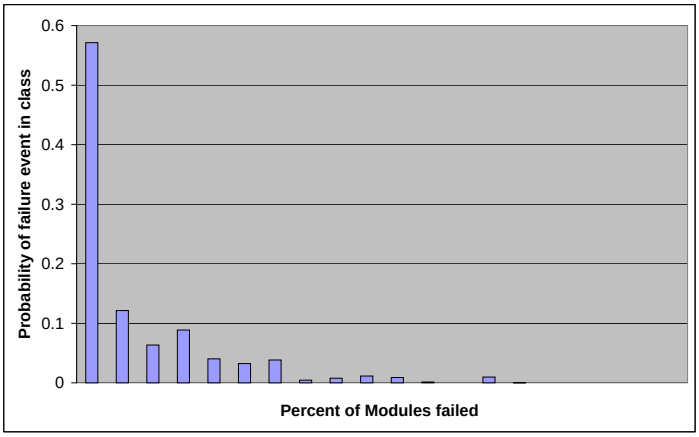


Figure 5.20.: Distribution of failed Modules for the as-is landscape

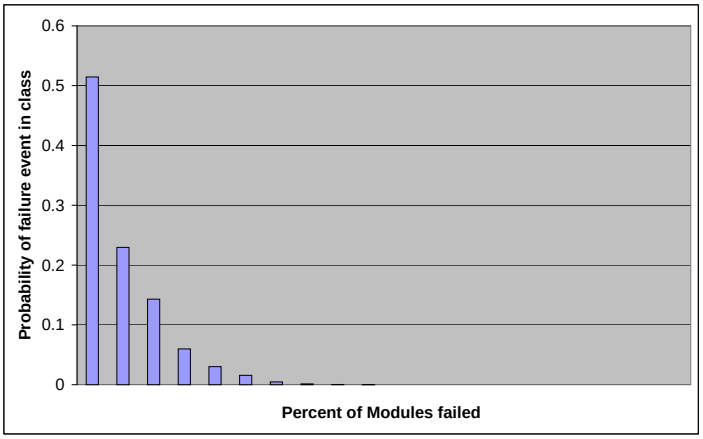


Figure 5.21.: Distribution of failed Modules for Proposal I

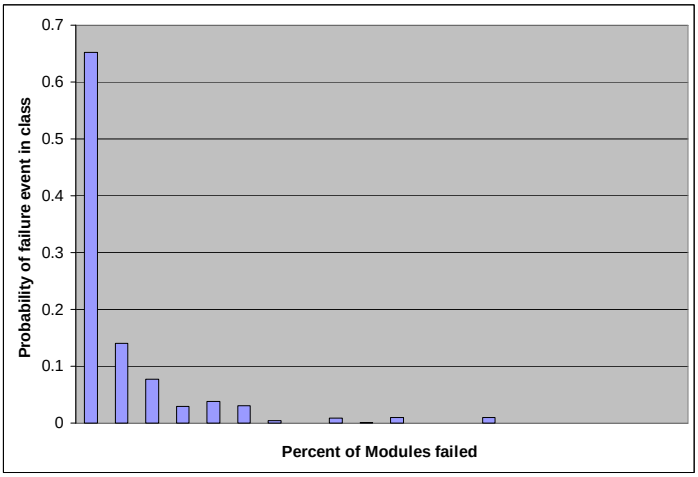


Figure 5.22.: Distribution of failed Modules for Proposal III

5.4.4. Outlook on Further Refinements of the Proposal Comparison

The methodologies described in Chapter 4, and the feedback the stakeholders gave to the proposal comparison, point to several directions, in which the models, the metrics derived from them, and the proposal comparison can be extended. Two are shortly discussed below.

5.4.4.1. Explicitly Modeling Hardware and Infrastructure Software

The extended information model for failure propagation structures, I-91, points to several situations where information about hardware and infrastructure software can be considered. This encompasses failures of network connections, and the non-synchronous dependencies with their technical realization.

In addition, the hardware and possibly infrastructure software, on which Applications are deployed, could be explicitly modeled as a source of failures. Especially in the context of Proposal I and Proposal II, where the four or five independent platforms could be operated on separate hardware, this extension could more fully capture the benefit of the proposals. However, these extensions would require additional data, which were not available at the point in time the case study was conducted.

5.4.4.2. From Modules to Transactions

Considering each Module in calculating the *averageServiceAvailability*, as done above, is certainly only an approximation. Most Modules are probably not directly relevant, at least not to business. They might not provide user interfaces, or be called externally by business partners, etc.

Moreover, different functionality might be of different relevance, which is modeled in I-91 (Section 4.3.2) via the LossFunction, but not at all considered in the case study.

However, in the future, information about how the transactions called by business map to Applications will be available. Then, these transactions can be interpreted as the Interfaces used by BusinessProcesses as introduced by I-90 or I-91. Additionally, information about how often these transactions are called will be available, which might even aid in constructing the LossFunctions for failures of the respective transactions.

5.5. Lessons Learned from the Case Study

Considering the case study as a whole, several lessons learned about application landscape metrics, tools to support their calculation and visualization, as well as how these metrics are applied and create benefit in practice can be summarized.

5.5.1. Data Supply and Data Quality

As many activities in application landscape management, metrics-based analyses rely heavily on adequate data, which is often difficult to collect and keep up to date.

In addition, data already available in an organization might not always strictly follow the requirements for calculating a given set of metrics. As exemplified by the failure propagation metrics in Chapter 4, metrics may be based on models with clear semantics, making statements falsifiable in reality, e.g., about failures in a specific BusinessApplication leading to an Interface of another BusinessApplication failing.

In practice, information having exactly these semantics is often not available, which can then lead to relying on assumptions and interpretations, as described in Section 5.2.1, however at the cost of probably introducing approximations into the metrics. Still, metrics can be considered as a powerful way of (re)using already collected application landscape data, drawing more benefit from the data often collected and maintained at a high effort.

The argument that metrics make improved use of available data points to the fact that they can also act as an incentive to provide or collect new data. Section 5.4.4.2 exemplifies this. The intention behind making the mapping from Applications to Transactions available for the metrics calculation is to refine the analyses made in the case study described here.

5.5.2. Tools for Calculating Application Landscape Metrics

If the data underlying a set of metrics are available, tool support is clearly necessary to derive and visualize the actual values. As apparent from the metrics introduction described in Section 5.2, tool support is already useful in the first stages of introduction, as it enables stakeholders to give feedback. In practice, one might easily encounter a situation where stakeholders are more likely to discuss an approach when they see it applied to their application landscape instead of discussing solely abstract elaborations.

When supporting metrics calculation with tools, a tension between ”developing individual tools based on an evolutionary process” and having standard tools appears. While standard tools enable swift creation of results, they might not always allow to appropriately use the data available in the often customized EA repositories of practitioners, or be able to respond to the demands and feedback of stakeholders in the iterative process of introducing metrics.

A possible approach for resolving this tension could be relying on highly customizable and integratable standard components in incrementally introducing application landscape metrics and their tool support.

5.5.3. Involving Domain Experts in Metrics-based Analyses

The case study shows that, in addition to data and tool support, it is essential to involve domain experts in providing useful metrics-based analyses. As in the basic steps of M-41 (Section 4.5.2.2), it is necessary that both metrics experts and domain experts are involved in introducing and using metrics for application landscapes. The domain experts guide the explorative process of refining concerns and the analyses to a considerable extent, making sure that their requirements are actually met.

5.5.4. Benefit of Application Landscape Metrics in Practice

Finally, the case study confirmed that metrics can be an aid in managing an application landscape. They provided the stakeholders with new insights into their proposals, among others:

- The different Domains benefit to a largely different extent from Proposal I and Proposal II (see Section 5.3.4.1).
- The Technical Platform-Modules are important in limiting failure propagation and have to be considered in the proposals (see Section 5.4.2).
- Proposals I and II not only increase the availability of specific functionality, but they also reduce the probability of large failures (see Section 5.4.3.2).

In addition to providing the stakeholders with information about these issues, the metrics-based analyses helped the stakeholders to refine the questions they ask about their proposals. For example, the reduced likelihood of large failures was not directly discussed by the stakeholders before the metrics analyses. This resulted in a more systematic discussion of approaches for limiting failure propagation, based on explicit assumptions and information about the application landscape. Thus, it also became obvious where additional information is necessary for improved analyses as a basis for deciding about the proposals.

CHAPTER 6

Conclusion and Outlook

Contents

This chapter summarizes the results of this work. From there, the chapter gives an outlook on future research about application landscape metrics, and how this field might influence application landscape management and EA management.

6.1. Conclusion: Introducing Metrics to Application Landscape Management

This work suggested metrics as an approach to application landscape management, introduced exemplary metrics-based methodologies, and demonstrated their usefulness in a real-world case study. Now, the research results are summed up, referencing the research questions posed by Section 1.3, which appear in boldface below.

The work gathered information about **which management situations metrics can support** via analysis of literature and questioning practitioners. Section 3.1 found metrics as a young but promising approach, with foundations and techniques still being worked out, but practitioners basically inclined to use it. Furthermore, the work identified quality attributes at the application landscape level, which practitioners especially wish to address via metrics, with functionality, operational risk, and flexibility being the ones most important to stakeholders. The work also identified the circumstances under which practitioners wish to use metrics, with communicating facts and findings, also to business stakeholders, being a major problem stakeholders want to address with metrics. This points to metrics being possible to contribute to more mutual involvement of business and IT in managing an application landscape.

Building on this, Section 3.2 introduced four guidelines showing, **how metrics and metrics-based methodologies for addressing concerns in the management of application landscapes can be constructed**. The first guideline proposes ways of incorporating metrics into methodologies for application landscape management, also covering aspects supposed to aid communication between business and IT. The second guideline discusses how to visualize metrics, heavily relying on software cartography. The *Heated Tree Map* implemented in the EA management tool *Troux 7.1* (Troux Technologies, Inc.) takes up one of these visualization guidelines [Pr08]. The third guideline elaborates on how to construct information models on which metrics are defined. The fourth guideline points to the importance of discussing the assumptions or theories underlying a specific metric to identify appropriate use cases.

The metrics introduced in Chapter 4 show, for the specific case of failure propagation, how **information about an application landscape can be supplied to management processes by metrics**. The work introduces information models for failure propagation structures in an application landscape, and builds metrics on them, which address the impact of failure propagation on availability, the probability of large failures, and the impact of specific applications failing on the total landscape. The metrics are complemented by viewpoints visualizing them. The methodologies using these metrics show how **metrics can improve the management of application landscapes**, supporting *Failure Propagation Analysis* and *Failure Propagation Specific Proposal Comparison*.

Finally, the case study demonstrated exemplarily how **metrics are used in practice**. Metrics were introduced to evaluate two real-life proposals for limiting failure propagation, also prototyping the tool support necessary for calculating the metrics. Then the metrics-based analyses were discussed with the stakeholders and iteratively refined in several steps. Thus, the stakeholders gained an improved overview of the effects connected to their proposals; on the one hand by the analyses themselves, and on the other hand by the refined questions about the proposals that were worked out during the case study. Differentiating between the availability of a specific functionality and the probability of large failures affecting major parts of the application landscape is one example for such a refined question. In providing this improved understanding of the situation at hand to the stakeholders, the case study showed that **metrics-based methodologies can be a successful instrument in managing application landscapes**.

6.2. Outlook: Using and Developing Application Landscape Metrics

In answering the above research questions, the work points to new directions for both research and application of approaches quantifying aspects of application landscapes.

6.2.1. Establishing Application Landscape Metrics

Currently, metrics-based approaches to application landscape management are not ubiquitously known, as demonstrated by Section 3.1.2.1. Evolving metrics for application landscapes into a mature approach depends on that they get an established technique in the field. This is essential for getting experience with the approaches, advancing the models and metrics, e.g., to improve their predictions, increase their understandability, or target them more precisely on the relevant subjects.

In order to establish the failure propagation metrics introduced in Chapter 4, the Chair for Informatics 19 (sebis) of the TU München will include the respective EA management patterns in future releases of its EA Management Pattern Catalog [Bu08a]. This catalog is a collection of best practices building on the EA management pattern approach, on which the metrics-based methodologies for addressing availability and failure coupling in Chapter 4 also rely. The catalog is based on approaches to specific EA management concerns found in literature and practice, which have been subjected to an extensive online questionnaire to identify methodologies and viewpoints, practitioners consider relevant and useful. The catalog will evolve further in a community process. Adding the metrics-based methodologies from Chapter 4 to the catalog is a step in the direction of introducing quantitative techniques in EA management, and a step toward the further development of methodologies for addressing failure propagation at application-landscape level.

Tools are essential in building acceptance for metrics-based techniques among practitioners. As mentioned in Section 5.5.2, tool support may be already indispensable in the first stages of introducing metrics in an organization. Without being able to actually derive metrics values in a practical situation, metrics are merely a theoretic possibility, and practitioners might have no feasible way of verifying, whether they are actually helpful in their specific situation. Thus, providing tools supporting calculating the metrics introduced in Chapter 4 beyond the prototypes used in Chapter 5 is essential in establishing these metrics in practice.

An important challenge in providing standard calculators for specific metrics is to allow users to apply the metrics calculators, despite using information models not directly matching, e.g., I-90, I-91, I-92, or I-93, without heavily adapting or re-creating the tools. Using such a differing information model was essential to the case study in Chapter 5, and can be expected to be central to using the metrics in practice, due to customized repositories being virtually ubiquitous in EA management or application landscape management [Bu08a, Ma08]. An approach based on views on the information, or model transformation languages (e.g., ATL [AT06] or BOTL [BM03]) might be useful here.

The assumption that e.g., an *Application* from an user-specific information model can be interpreted as a *BusinessApplication* of I-90 can be implemented by a corresponding transformation rule. Metrics calculators can then process the output of this transformation.

6.2.2. Refining and Developing Application Landscape Metrics

While the metrics introduced in Chapter 4 were able to support stakeholders in the case study presented in Chapter 5, they constitute only a first step in establishing metrics as a way of providing extensive support to application landscape management, ubiquitously used in the respective processes, and applied to a wide range of problems.

A first step into this direction could be extending the variants of the failure propagation metrics as supplied by I-90 and I-91 by ones that rely less on assumptions and allow more realistic modeling. [Br07], for example, introduce an approach for building detailed formal models of application landscapes based on libraries of pre-defined components hiding the complexity of formal modeling. These models are able to capture internal states of the business applications, and detailed semantics of the interconnections between the applications. Extending such models to consider failures and how they are propagated in an application landscape can be a step toward more expressive and realistic models.

However, using metrics more wholistically than just for examining failure propagation aspects relies on metrics for more quality attributes, as the ones pointed out in Section 3.1.2.3. To sketch some examples, one could try to build metrics measuring the impact of the application landscape on the changeability of specific business applications. Effort estimation models, e.g., COCOMO II [USC00], or Function Points [Lo08] contain influencing factors, some of which reflect characteristics covered by an application landscape model. Homogeneity aspects, with tendencies of projects needing expertise in many different technologies being less productive, are an example thereof. Building metrics this way would link them to the considerable knowledge base already available regarding effort estimation models.

Quantitative approaches for assessing the impact of application landscape structures on quality attributes are not limited to metrics. Simulations of application landscapes constitute another approach, possibly able to address a plethora of quality attributes based on rich, complex landscape models. A wide variety of simulation techniques are available to support a multitude of possible approaches in this direction, with two examples sketched below:

- The processes supported by the application landscape could be simulated, e.g., by creating corresponding (stochastic) petri nets [Li92, Je96] from an application landscape model, and simulating them to get insights, e.g., about latencies or loads at specific business applications or computation nodes.
- Agent-based simulations [Wo02] of how different stakeholders interact to evolve the application landscape could be conducted to explore the effect of architectural standards and different ways of enforcing them on changeability. This leaves the

domain of modeling purely technical systems, and extends the models to include also organizational aspects. Thus, such models constitute a step from application landscape management to full EA management.

6.2.3. Bridging the Gap between EA Management and Business

According to [ELW06], EA management "not only considers the information technology of an enterprise, but also business processes, business goals, strategies, etc. are considered in order to build a holistic and integrated view on the enterprise." In practice, however, EA management is often the domain of IT architects, and lacks the business involvement to comply with the above vision.

A hypothetical reason for this *gap* is that EA management is often concerned with technical information, e.g., application components, interfaces, hardware platforms, or architectural standards. Metrics, as presented in this work, constitute an approach for extracting the business implications of these technical information, e.g., changeability, or, as presented in Chapter 4, availability. Considering the findings from the status quo analysis in Section 3.1.1, metrics present themselves as a way of involving business more in EA management.

Moreover, using metrics quantifying aspects of an application landscape structure in approaches as the balanced scorecard, IT balanced scorecard, or IT controlling [Kü03, Kü05], would provide these domains with more foundation in the systems and structures they are managing. As an example, [Ju04] introduces *risks in IT operation*[1] as a kind of risk to be considered in *value-oriented risk controlling in information management*[2]. The metrics-based methodologies introduced in Chapter 4 constitute a means of analyzing risks of insufficient availability posed by the application landscape structure. Thus, using application landscape metrics, EA management can proceed another step in bridging the gap between business and IT.

As far as those approaches might seem from current practices, as put forward by EA management tools, or EA management frameworks, they point in a direction where EA management is less an art practiced by IT staff but a science providing documented, predictable approaches to experts supporting business with the IT business needs.

[1] In German: Betriebs-Risiken
[2] In German: Wertorientierte Steuerung von Risiken im Informationsmanagement

APPENDIX A

Status Quo Data in Detail

A.1. Guided Interviews

A.1.1. Interview Schedule

	German (original)	**English (translation)**
1.	Gesprächsbeginn	Start of interview
1.1	Begrüßung des Interviewpartners	Greeting the interview partner
1.2	Vorstellung der anwesenden Gesprächspartner	Introduction of the present dialog partners
1.3	Vorstellung der Ziele der Studie	Outline of the goals of the survey
1.4	Vorstellung des ungefähren Gesprächsablaufs	Sketching the course of the interview
1.5	Begründung der Tonbandaufnahme, Erläuterung, dass Transkript anonymisiert wird und nur Abteilungsart und Branche verbleiben sowie Klärung des Einverständnisses des Interviewpartners; Transkript wird dem Interviewpartner vor der Auswertung zur Durchsicht (vertrauliche Informationen) zur Verfügung gestellt.	Giving reasons for recording the interview; outlining, that the interview transcript is anonymized, only the kind of organizational unit and line of business of the interview partner is kept; getting the consent of the interview partner; the interview partner will have an opportunity to review the transcript and remove confidential information before it is analyzed.
1.6	Klären, ob der Interviewpartner noch Fragen hat	Clarifying further questions

2.	Fragen zur Person	Personal information
2.1	Wir haben Interesse an Informationen, aus welchem Blickwinkel und mit welchem Vorwissen Sie EAM betrachten. Wie sind Sie zu Ihrer aktuellen Tätigkeit gekommen und was haben Sie vorher gemacht?	We are interested in information about from which point of view and with which knowledge you view EA management. How did you achieve your current position and what positions did you hold previously?
3.	EAM im Unternehmen	EAM in the company
3.1	Welchen Aufgabenbereich haben Sie inne? Mit welchen Aufgaben, Tätigkeiten sind Sie regelmäßig beschäftigt? (eventuell Beispiele bringen; Vorsicht suggestiv; Granularität ähnlich Toolstudien-Szenarios [se05a] anstreben)	In which fields of activities do you work? Which tasks do you address regularly? (In order to concretize this questions, examples can be given, however abstaining from being suggestive. A granularity similar to the scenarios of the Enterprise Architecture Management Tool Survey 2005 [se05a] is envisaged.
3.1.1	Weitere Klärung: Nennen Sie bitte drei Elemente einer EA, mit denen Sie beschäftigt sind (z.B. Prozesse, Strategien und Ziele, Anwendungen, Infrastruktur, Projekte und Programme).	Further clarification: please state three kinds of elements of an EA you are concerned with (e.g. processes, strategies and goals, business applications, infrastructure, projects and programs)
3.1.2	Weitere Klärung: Nennen Sie drei Abteilungen, Rollen oder Gremien mit denen Sie im Rahmen von EA-Aufgaben zu tun haben.	Further clarification: please state three organizational units, roles or boards with which you interact in EA management
3.1.3	Weitere Klärung: Erläutern Sie die Art der Zusammenarbeit (Informationslieferant, Ergebnisinteressent,...) mit diesen Abteilungen, Rollen oder Gremien.	Further clarification: please state, how you interact with above organizational units, roles, or boards (information supplier, user of supplied information, ...)
3.2	Mit welche Problemstellungen werden Sie beim Management der EA und der Anwendungslandschaft konfrontiert?	Which are your concerns in managing the EA and the application landscape
3.3	Aufgreifen von ca. drei genannten Problemstellungen und versuchen folgende Fragen anhand dieser Problemstellungen zu beantworten:	Please give us further information about three of these concerns:

3.3.1	Adressieren Sie persönlich diese Problemstellungen? Wenn nein, wer dann?	Do you address the concerns in person? If not, who does?
3.3.2	Welche Bereiche (Fachabteilungen) sind in den Adressierungsprozess involviert (Fachabteilungen, Management, IT, ...)?	Which organizational units are involved in addressing the concerns (functional departments, management, IT, ...)?
3.3.3	Welche Rolle nehmen diese Abteilungen ein (Input-Geber, Output-Nehmer, Interessentreiber, Ausführung, Entwicklung der Methode)?	What are the roles of these organizational units (giving input, receiving information, executing tasks, developing approaches)?
3.3.4	Können Sie bei den Problemstellungen des EA Management Tendenzen, was die Erscheinungshäufigkeit anbelangt, erkennen?	Do you see trends regarding the concerns in EA management and their frequency?
3.3.5	Folgen Sie einer dokumentierten Vorgehensweise/Methodik zur Lösung dieser Problemstellungen? Darauf achten, dass die Methode den Nutzen der Darstellungen beschreibt und nicht nur die Erstellung? Wo liegt das Wissen über diese Vorgehensweise (dokumentiert, implizites Wissen einer oder mehrerer Personen)?	Do you use a documented procedure/method for addressing the concerns? The method description should not only focus on how views are created, but also on their benefit. Where is the knowledge about the methods (documented, implicit knowledge of one ore more persons)?
3.3.6	Werden dabei graphische Darstellungen/Visualisierungen/Reports zur Adressierung verwendet? Erläutern Sie diese kurz! Interviewer 2 skizziert den Viewpoint. Sind sie mit der Aktualität der Darstellung zufrieden? Wie werden diese Darstellungen genutzt?	Are graphical descriptions/visualizations/reports used in addressing the concerns? A short description of these should be given. The second interviewer could sketch the viewpoint. Are you content with the topicality of the diagrams? How are the diagrams used?
3.3.7	Müssen die beteiligten Personen über bestimmte Vorkenntnisse (z.B. UML), Kompetenzen oder Rollen verfügen, um die Adressierung durchführen zu können?	Are there specific skills, competencies or roles the involved persons need to have in order to address the concerns (e.g. UML knowledge)?
3.3.8	Wo liegen die Kernaufwände beim entsprechenden Ansatz?	Where are the core efforts connected to a specific approach?

3.3.9	Stimmt die Frequenz der Adressierung mit der Frequenz der Fragestellung überein? Wenn nein, welche Gründe gibt es für diese Diskrepanz?	Is a concern addressed as often as it arises? If not, are there reasons for the discrepancy?
3.3.10	Welche Abteilungen, Rollen oder Gremien erhalten Output aus der Bearbeitung der Problemstellung in Form von Vorgaben?	Which organizational units, roles, or boards receive guidelines output from addressing the concern?
3.3.11	Wie wird sichergestellt, dass diese Vorgaben auch eingehalten werden (Finanzierung, Projektleitung, ...)?	How is ensured, that these guidelines are obeyed (financing, project management, ...)?
3.3.12	Gibt es häufig auftretende Probleme bei dieser Vorgehensweise (z.B. unzureichende Dokumentation der Methodik; Input nur unzureichen verfügbar; Vorgaben (Output) werden nicht angenommen; Darstellungen werden nicht aktualisiert)?	Are there frequent problems with the approaches (e.g. insufficient documentation of methods; necessary input not available; output is not accepted; outdated diagrams)?
3.3.13	Worin sehen Sie die Stärken des von Ihnen gewählten Ansatzes zur Adressierung der Problemstellung?	What are the strengths of your approach?
3.3.14	Worin sehen Sie die Schwächen des von Ihnen gewählten Ansatzes zur Adressierung der Problemstellung?	What are the weaknesses of your approach?
3.3.15	Haben Sie weitere Problemstellungen, die sie gerne durch eine dokumentierte Vorgehensweise adressieren würden?	Are there additional concerns which you would like to address using a documented method?
4.	Cover up Fragen zu Eigenschaften von EA falls durch 3 noch nicht beantwortet	Cover up questions about aspects not yet covered in 3
4.1	Haben Sie ein explizites Informationsmodell und ist dieses in einem repository-basiertem Tool umgesetzt?	Do you have an explicit information model and is this implemented in a repository-based tool?
4.2	Welche Eigenschaften wollen Sie an ihrer aktuellen EA verbessern?	Which properties of your current EA do you want to improve?
4.3	Weitere Klärung: Nennen und Erläutern Sie die drei wichtigsten Qualitätsmerkmale für eine EA.	Further clarification: please state the three most important quality attributes for an EA.
4.3.1	Nutzen Sie Metriken in diesem Zusammenhang?	Do you use metrics in this context?

4.3.2	Können Sie sich vorstellen in diesem Zusammenhang Metriken zu verwenden?	Could you imagine using metrics in this context?
4.3.3	Warum nicht, bzw. was versprechen Sie sich davon?	If not, what is the reason, and if yes, what do you expect from using metrics?
4.4	Durch wen wird der EA Management Prozess in Ihrem Unternehmen getrieben?	Who drives EA management in your company?
5.	Fragen zu zukünftigen Entwicklung von EA Management	Questions regarding the future development of EA management
5.1	Was denken Sie wird sich bezüglich der von Ihnen beschriebenen Dinge (Problemstellungen, Aufgabenbereiche, Methodiken, ...) in Zukunft ändern?	Which changes do you expect in the things you described in the interview (concerns, fields of activities, methods, ...)
5.2	Gibt es Probleme/Fragestellungen, die durch EA Management endgültig gelöst werden können?	Are there problems which will be solved by EA management conclusively?
5.3	Welche Fragestellungen werden in 10 Jahren für ihr Unternehmen interessant sein?	Which concerns will be interesting for your company in 10 years?
5.4	Glauben Sie das SOA einen Einfluss auf das EA Management haben wird und wenn ja welchen?	Do you believe that SOA will influence EA management, and what will that influence be?
5.5	Gibt es weitere Aspekte ihrer Anwendungslandschaft, die sie gerne durch eine EA Management-Methodik adressieren würden?	Are there additional aspects of your application landscape you would like to address using an EA management-method?
6.	Fragen, die eventuell noch am Schluss gestellt werden könnten, um Beeinflussung auf sonst folgende Fragen zu vermeiden	Questions posed at the end of the interview, as they could influence the interviewed person
6.1	Was ist für Sie EA Management (eventuell aufgreifen der in 3.1 genannten Aufgabenstellungen und nach deren Beziehung zu EA Management fragen)?	What is EA management to you (possibly, the tasks described in 3.1. can be taken up again, asking what their relation to EA management is)?
6.2	Sehen Sie die Rolle von Geschäft und IT im Rahmen der EA Management-Problemstellungen als gleichberechtigt an, oder wird es von einer Seite stärker getrieben?	How do you perceive the role of business and IT in EA management? Are they equal partners, or is one side more active in driving the subject?

A.1.2. Interview Results Concerning Metrics

This section summarizes the metrics-related aspects of the EAMVS interviews, as described in Section 3.1.1. From some organizations, two persons have been interviewed in distinct interviews. Thus, interviews 14 and 17, 19 and 20, and 21 and 22 are from the same organization respectively.

Interview	General description	Usage purpose of metrics	Quality attributes, Unit of observation	Org. size	Business
1	Used metrics: Number of users, first tries regarding numbers of installations (data indicating more installations than users were an interesting hint regarding data quality)	Rethinking licensing issues (Are enough licenses available? Is it sensible to strategically focus on specific applications?)	Technical fit (extent to which a specific platform fits into the application landscape), considering support of open standards, influence on integrating applications, ease of administration and configuration, ease of development; Extent of support to business (redundant data, format conversions, reduction of manual process steps, business value, saved time) Business continuity (risk posed by application failures, recovery aspects)	large (division)	Manufacturing
2	Note: describes work at clients; Tried measuring a *degree of functional coverage* (to which extent is a process supported by applications), however not with "100% success"; Number of users; usage frequencies; cost aspects	The degree of functional coverage was meant to include detail information from low-level process chains into e.g. a process support map.	No standard catalog of quality attributes, but some points are sensible to look at in a first step (e.g. number of applications supporting a process, redundancy, ...) Evaluation of an (application landscape) architecture as a whole usually not done	large (division)	Professional, Scientific, and Technical Services
3	Use a standard framework for classifying applications in different portfolio models, considering both technical and functional criteria	Goal of classification: identifying need for action		large	Finance and Insurance
4	Measure IT efficiency (in development) and business value from IT in function points; Would like more approaches for estimating the business value of proposed changes, sees a problem in collecting information for metrics calculation	Tracking and controlling the evolution of the application landscape		large	Finance and Insurance

Table A.2.: Using metrics

Interview	General description	Usage purpose of metrics	Quality attributes, Unit of observation	Org. size	Business
5	Metrics usage rather limited: number of users, average number of concurrent users		Functional and technical dependencies: in a good application landscape, the technical infrastructure can untangle functional dependencies (e.g. using an integration broker)	large	Transportation and Warehousing
6	Application counts (used for sizing e.g. projects or business logic); Interface/coupling aspects	Planning, monitoring, and evaluating (original interview: "steuern")	Homogeneity of used technologies (original interview: "Wildwuchs vermeiden")	large	Finance and Insurance
7	Metrics used in maintenance department and development department, to measure developed software: errors/functionpoint, number of hotfixes, etc.; metrics values influence salaries			large	Finance and Insurance
8	Performance measurement of processes (e.g. how many contracting opportunities result in actual contracts); benchmarks; degree of functional coverage (of processes by applications); numbers of necessary and available licenses; business cases for functionality demanded by processes, or for important applications exist or are planned	For performance measurement: checking whether targets are met, if not, searching for the reasons; Benchmarking: how much does e.g. an SAP user cost in divisions in other countries, and what are the reasons for differences?		large (division)	Manufacturing
9	SoToGraph is used in specific projects; metrics are used for performance tuning of host applications; Less metrics usage regarding the application landscape as a whole (e.g. redundancy)	SoToGraph metrics: used for setting goals	Redundancy in an application landscape; applications should be exchangeable; changes regarding the frontend should be supported	large	Finance and Insurance

Table A.3.: Using metrics (continued)

Interview	General description	Usage purpose of metrics	Quality attributes, Unit of observation	Org. size	Business
10	First thoughts about metrics exist, however, necessary maturity is not believed to be reached. Necessary data quality possibly not achieved.			large	Transportation and Warehousing
11	Key performance indicators for architecture are currently introduced (e.g. number of architecture violations); Problem: in architecture, it is difficult to link metrics to a specific business value; the analyses would be expensive	Transparency, gaining and keeping improved overview	Complexity of the system landscape, which influences costs and efforts; Current goals: reduce complexity, reduce time to market, increase scalability; Achievement of the goals is currently rated via expert judgments; A current plan is to introduce about 10-15 criteria for goal achievement, to get more systematic judgments	medium (division)	Information (Telco)

Table A.4.: Metrics usage planned

Interview	General description	Unclear issues and problems with metrics	Envisaged usage purpose and quality attributes	Org. size	Business
12	Currently no approaches or ideas for using metrics for application landscapes	Sceptical regarding what to calculate, and the explanatory power of such calculations	Difficult, depends on business goals	large	Manufacturing
13	Currently no metrics for application landscapes; There are metrics for assessing projects ("Projektbewertungsprozess")	Do not know, what adequate metrics for assessing strategic developments would be		large	Finance and Insurance
14	Interviewed person would like to use function points, for estimating change projects: how many functionality does a project affect/change	Employees would feel controlled	Quality attributes exist	large	Transportation and Warehousing
15		Currently no knowledge about metrics which could be used for application landscapes	Quality attributes: standard conformity (e.g. for programming languages); interface counts	medium	Manufacturing
16	Some thoughts exist, also with respect to integrating application landscape aspects into the existent IT balanced scorecard	Unclear, what to measure (thoughts about numbers and complexity of interfaces exist); data collection would be a problem		medium	Manufacturing
17	Speaks only about data quality			large	Transportation and Warehousing
18	Work done by the architecture group should be measurable; Redundancy factors as KPIs; Metrics are used in operative areas (quality control), however, this is not really related to IT architecture; Quality is assessed for specific projects (punctuality, etc.) and for finished products (automated testing)		Reduced redundancy	large	Information (Telco)
19	KPIs (e.g. degree of functional coverage) are currently developed			large	Manufacturing

Table A.5.: Metrics possible, but many issues are not clear at the moment

Interview	General description	Problems and negative experience	Tried usage purpose of metrics	Org. size	Business
20	Relevant quality attributes include: flexibility and standardization; Metrics are used in business (e.g. number of warranty cases); Metrics for the structure of the application landscape, e.g. number of interfaces, are not used	No need for metrics measuring the structure of the application landscape		large	Manufacturing
21	KPIs are used in production (for monitoring and reporting): availability, throughput, utilization of applications, punctuality of projects	Complexity metrics were tried but turned out to be too difficult (resources for development and introduction not available)		medium	Finance and Insurance
22	Possible quality attributes: redundancy, length of "data supply paths" (number of applications via which data is sent from the application it originates to a specific using application)	Problem of complexity metrics: necessary information often not available, comparability of metrics often cannot be ensured; However, simple interface counts still seem to play a role	A usage purpose of the complexity metrics would have been controlling efficiency (e.g. controlling complexity vs. IT personnel)	medium	Finance and Insurance

Table A.6.: Metrics not possible, makes no sense

A.2. Online Questionnaire

This section presents excerpts from the EAMVS online questionnaire and its statistical evaluation, which are relevant to the environment analysis in Section 3.1.

A.2.1. Excerpt from EAMVS Online Questionnaire

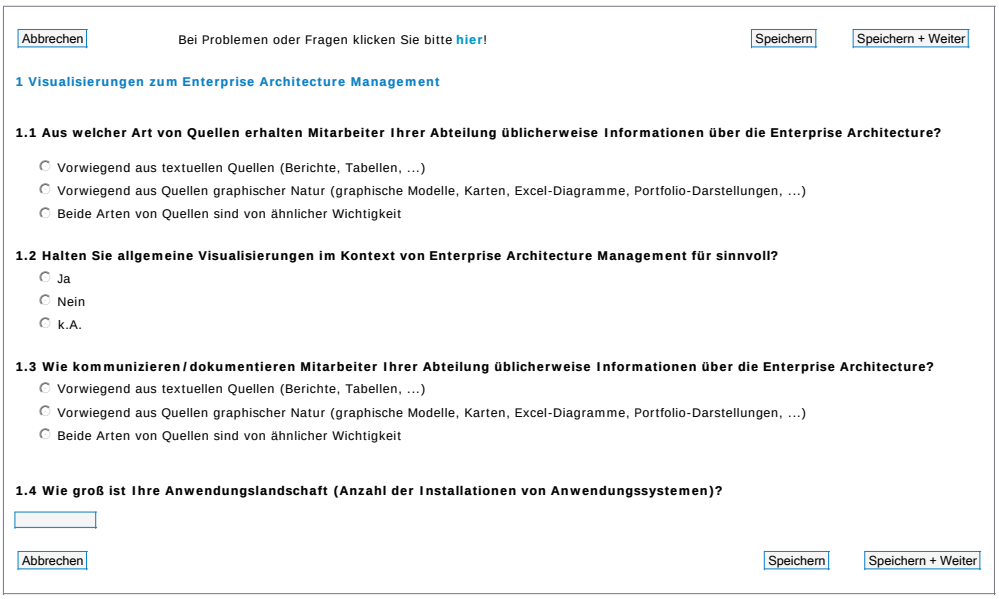
Subsequently excerpts from the online questionnaire are shown, to document how the online questionnaire responses have been collected.

A.2.1.1. Excerpt from EAMVS Online Questionnaire: Metrics-related Questions

This section presents the parts of the online questionnaire which contained questions used in the environment analysis in Section 3.1.

A.2.1.1.1. Questionnaire Section IIX: Concluding Questions (Abschließende Fragen)

Visualizations in EA management
(Visualisierungen zum Enterprise Architecture Management):



Best Practices and Trends (Best Practices and Trends):

Abbrechen Bei Problemen oder Fragen klicken Sie bitte hier! Speichern Speichern + Weiter

2 Best Practices und Trends

2.1 Wie würden Sie die Definition der Ziele für die Evolution der Anwendungslandschaft einschätzen?

- ○ Ziele sind nicht explizit dokumentiert und gesammelt, sondern verteilt, u.U. auch in den Köpfen entsprechender Mitarbeiter.
- ○ Klare Vorgaben existieren in einem dafür zuständigen Dokument.
- ○ Keine Vorgaben bzgl. Zielen vorhanden.
- ○ k.A.

2.2 Werden in Ihrem Unternehmen EA Frameworks eingesetzt?

- ○ Nein.
- ○ Ja, sie dienen als Ideengeber für den eigenen EAM Ansatz.
- ○ Ja, wir setzen ein angepasstes EA Framework ein
- ○ Ja, wir setzen ein EA Framework Out of the Box ein
- ○ k.A.

Falls ja, welches Framework?

2.3 Spielen in Ihrem Unternehmen Service-orientierte Architekturen eine Rolle?

- ○ Nein.
- ○ Es wird aktuell überlegt eine Service-orientierte Architektur einzuführen.
- ○ Eine Service-orientierte Architektur wird momentan in einem Bereich eingeführt.
- ○ Eine Service-orientierte Architektur wird momentan unternehmensweit eingeführt.
- ○ Das Unternehmen nutzt bereits eine Service-orientierte Architektur
- ○ k.A.

2.4 Wie werden in Ihrem Unternehmen momentan Informationen für das Management der Enterprise Architecture erhoben?

- ☐ Automatisch, z.B. durch Quellcodeanalyse, Crawler
- ☐ In periodischen Zyklen (jährlich, halbjährlich, monatlich, ...)
- ☐ Kontinuierlich, im Rahmen von Projekten
- ☐ Kontinuierlich durch z.B. Anwendungs- oder Prozessverantwortliche
- ☐ k.A.

2.5 Sind Sie mit der Datenqualität zufrieden?

- ☐ Nein, die Aktualität ist nicht ausreichend.
- ☐ Nein, die Korrektheit ist nicht ausreichend
- ☐ Nein, die Vollständigkeit ist nicht ausreichend
- ☐ Ja, die Datenqualität ist ausreichend
- ☐ k.A.

2.6 Werden in Ihrem Unternehmen Infrastrukturinformationen, z.B. aus einer CMDB Datenbank für das Management der Enterprise Architecture genutzt?

- ○ Nein.
- ○ Die Nutzung dieser Daten wird momentan geplant.
- ○ Infrastrukturinformationen werden bereits genutzt.
- ○ k.A.

2.7 Wer treibt in Ihrem Unternehmen das EAM?

- ○ Geschäft
- ○ IT
- ○ Geschäft und IT gemeinsam
- ○ k.A.

2.8 Ist in Ihrem Unternehmen das Geschäft ausreichend in das EAM eingebunden?

- ○ Ja
- ○ Nein
- ○ k.A.

2.9 Wo werden in Ihrer Organisation die Zielvorgaben für die zukünftige Evolution der Anwendungslandschaft aufgestellt?

- ○ Verschiedenste Personen/Gruppen versuchen solche Zielvorgaben aufzustellen, besitzen allerdings keine formalen Weisungsbefugnisse sie durchzusetzen
- ○ Eine Person/Gruppe (z.B. IT-Architekten) versuchen solche Zielvorgaben aufzustellen, besitzen allerdings keine formalen Weisungsbefugnisse sie durchzusetzen
- ○ Eine Person/Gruppe (z.B. IT-Architekten) mit den entsprechenden formalen Weisungsbefugnisse stellt solche Zielvorgaben auf)
- ○ Verschiedenste Personen/Gruppen mit den entsprechenden formalen Weisungsbefugnissen stellen solche Zielvorgaben auf.

Abbrechen Speichern Speichern + Weiter

Metrics and Measures (Metriken und Kennzahlen):

3 Metriken und Kennzahlen

3.1 Haben Sie (Ihre Organisation) bisher Metriken im Management der Anwendungslandschaft verwendet?

- ○ Ja
- ○ Nutzung vorstellbar
- ○ Nutzung nicht vorstellbar

3.2 Wie würden Sie den Umfang der Metriknutzung in Ihrer Organisation einschätzen?

- ○ Ein kleineres Vorhaben (einzelne Personen haben damit experimentiert)
- ○ Einige Mitarbeiter verwenden regelmäßig Metriken
- ○ Metriken sind im Management von Anwendungslandschaften weit verbreitet
- ○ Zentraler Bestandteil des Managements von Anwendungslandschaften

3.3 Was wäre/ist Ihre Aufgabe im Rahmen eines Metrikeinsatzes beim Management von Anwendungslandschaften?

- ○ Daten sammeln
- ○ Daten nachbearbeiten
- ○ Validieren/Kontrollieren von Daten und Metriken
- ○ Analysieren von Daten und Metriken
- ○ Präsentieren von Metriken
- ○ Einsatz der Daten, um Ziele für Projekte/die Anwendungslandschaft zu verfolgen
- ○ Planung des Metrikeinsatzes

3.4 Welche Merkmale der Anwendungslandschaft wollen Sie mit Metriken untersuchen?

3.4.1 Wartbarkeit: Die im Rahmen der technischen Entwicklung auftretenden Wartungsaufwände sollen gering sein.
1 (benötige ich nicht) bis 5 (brauche ich unbedingt)

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ k.A.

3.4.2 Flexibilität: Änderungen bzgl. der fachlichen Funktionalität (kunden- bzw. marktgetrieben) sollen schnell und kostengünstig umgesetzt werden können.
1 (benötige ich nicht) bis 5 (brauche ich unbedingt)

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ k.A.

3.4.3 Testbarkeit: Tests mit einer gegebenen Testabdeckung sollen mit möglichst geringem Aufwand durchgeführt werden.
1 (benötige ich nicht) bis 5 (brauche ich unbedingt)

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ k.A.

3.4.4 Performanz: Die Systeme der Anwendungslandschaft müssen in der Lage sein, bezüglich der Ausführung von Trades bestimmte SLAs unterstützen zu können.
1 (benötige ich nicht) bis 5 (brauche ich unbedingt)

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ k.A.

3.4.5 Skalierbarkeit: Die Anwendungslandschaft muss die Verteilung von Last auf neue Hardware unterstützen.
1 (benötige ich nicht) bis 5 (brauche ich unbedingt)

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ k.A.

3.4.6 Operationales Risiko: Das aus der Anwendungslandschaft erwachsende operationale Risiko soll gering gehalten werden.
1 (benötige ich nicht) bis 5 (brauche ich unbedingt)

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ k.A.

3.4.7 Kostenvorteile im Betrieb: Der Betrieb der Anwendungslandschaft, sowohl aus fachlicher als auch aus technischer Sicht, soll sich kostengünstig realisieren lassen.
1 (benötige ich nicht) bis 5 (brauche ich unbedingt)

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ k.A.

3.4.8 Installierbarkeit: Installationen müssen sich schnell und mit wenig Aufwand durchführen lassen.
1 (benötige ich nicht) bis 5 (brauche ich unbedingt)

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ k.A.

3.4.9 Funktionalität: Die Anwendungslandschaft muss das Geschäft durch die bestehende Funktionalität in seiner Geschäftstätigkeit bzw. der Ausführung der Prozesse unterstützten.
1 (benötige ich nicht) bis 5 (brauche ich unbedingt)

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ k.A.

3.5 Bezüglich welcher Betrachtungseinheiten würden Sie sich Metriken wünschen?

3.5.1 Metriken, die einen Wert für die gesamte Anwendungslandschaft berechnen
1 (benötige ich nicht) bis 5 (brauche ich unbedingt)

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ k.A.

3.5.2 Metriken, die einen Wert für eine bestimmte Domäne/Gruppierungen betrachten (z.B. „Lebensversicherungen", „Lagerhaltung", etc.)
1 (benötige ich nicht) bis 5 (brauche ich unbedingt)

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ k.A.

3.5.3 Metriken, die einem einzelnen Bestandteil der Anwendungslandschaft (einzelner Prozess, Service, betriebliche Anwendungen) einen Wert zuweisen
1 (benötige ich nicht) bis 5 (brauche ich unbedingt)

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ k.A.

3.6 Wie würden Sie die Aufgaben bei denen Sie Metriken einsetzen wollen, charakterisieren? Geben Sie für jede Art von Aufgabe an, welches Potential sie für einen Metrikeinsatz sehen?

3.6.1 Besseres Verstehen bestimmter Problemstellungen, um den Prozess der Lösungsfindung zu verbessern
1 (kein Potential) bis 5 (hohes Potential)

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ k.A.

3.6.2 Vorhersagen der Auswirkung bestimmter Maßnahmen treffen
1 (kein Potential) bis 5 (hohes Potential)

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ k.A.

3.6.3 Ziele vorgeben (als Sollwerte für Metriken) und die Zielerreichung kontrollieren (Operationalisierung von Architekturzielen)
1 (kein Potential) bis 5 (hohes Potential)

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ k.A.

3.6.4 Management der Verbesserung der Anwendungslandschaft: Status Quo, Verbesserungen und Verbesserungspotential aufzeigen
1 (kein Potential) bis 5 (hohes Potential)

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ k.A.

3.7 Was erwarten Sie sich von Metriken?

3.7.1 Ich kann gewisse Sachverhalte besser darstellen und kommunizieren
1 (Erwartung besteht nicht) bis 5 (Erwartung besteht auf jeden Fall)

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ k.A.

3.7.2 Ich kann gewisse Sachverhalte schneller verstehen
1 (Erwartung besteht nicht) bis 5 (Erwartung besteht auf jeden Fall)

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ k.A.

3.7.3 Ich bin besser informiert und kann fundiertere Entscheidungen treffen
1 (Erwartung besteht nicht) bis 5 (Erwartung besteht auf jeden Fall)

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ k.A.

3.7.4 Ich kann mir einen schnellen Überblick über bestimmte Sachverhalte schaffen
1 (Erwartung besteht nicht) bis 5 (Erwartung besteht auf jeden Fall)

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ k.A.

3.8 Welche Eigenschaften sind für Sie bei Metriken relevant?

3.8.1 Einfache Berechnungsvorschrift
1 (benötige ich nicht) bis 5 (brauche ich unbedingt)

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ k.A.

3.8.2 Mitarbeiter fühlen sich durch die Metrik kontrolliert
1 (benötige ich nicht) bis 5 (brauche ich unbedingt)

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ k.A.

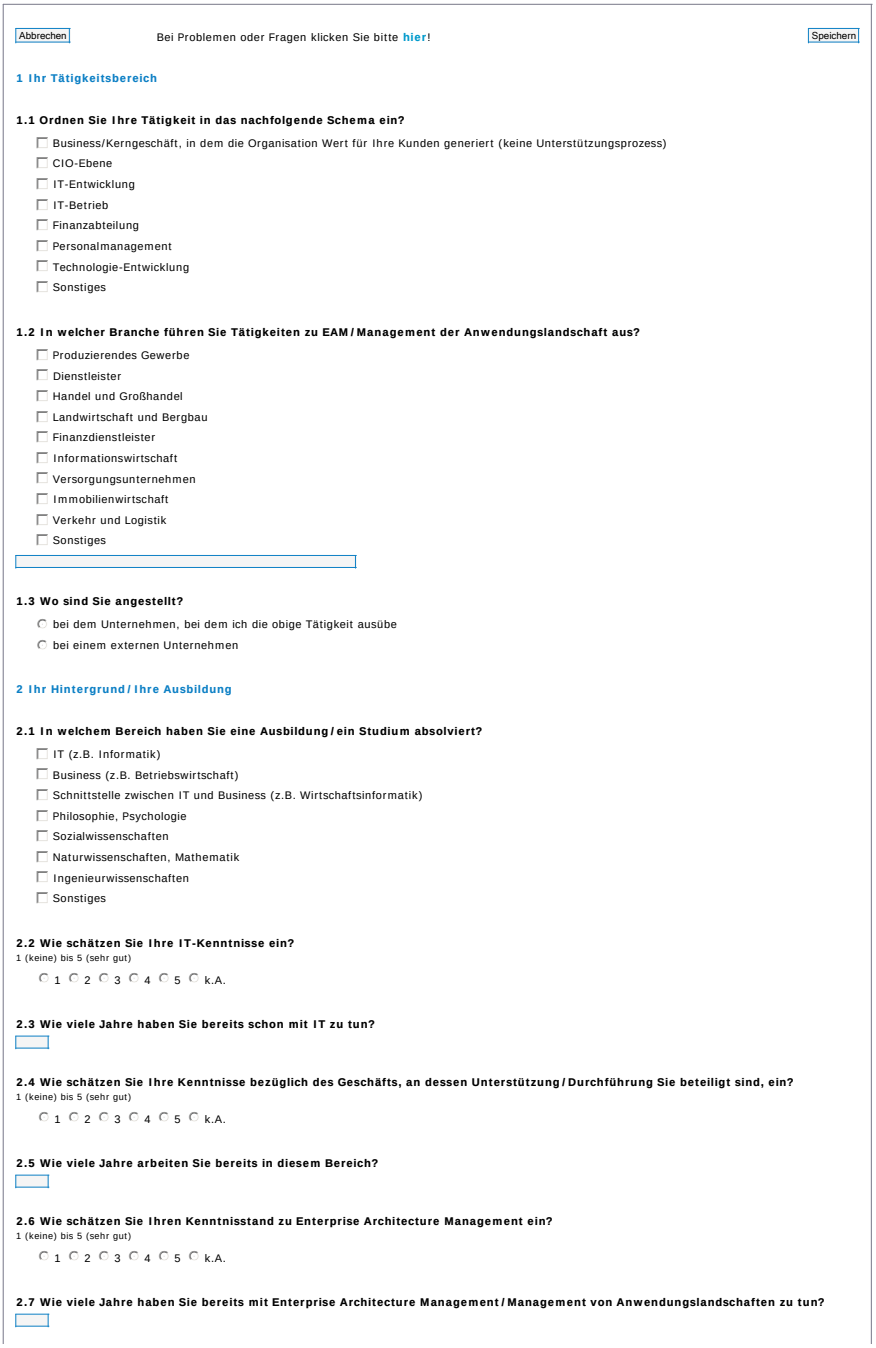
3.8.3 Die Metrik sollte fundiert und wohlverstanden sein
1 (benötige ich nicht) bis 5 (brauche ich unbedingt)

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ k.A.

3.8.4 Die Bedeutung der Metrik sollte sich leicht kommunizieren lassen
1 (benötige ich nicht) bis 5 (brauche ich unbedingt)

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ k.A.

A.2.1.1.2. Questionnaire Section IX: Personal Questions



Abbrechen Bei Problemen oder Fragen klicken Sie bitte hier! Speichern

1 Ihr Tätigkeitsbereich

1.1 Ordnen Sie Ihre Tätigkeit in das nachfolgende Schema ein?

- ☐ Business/Kerngeschäft, in dem die Organisation Wert für Ihre Kunden generiert (keine Unterstützungsprozess)
- ☐ CIO-Ebene
- ☐ IT-Entwicklung
- ☐ IT-Betrieb
- ☐ Finanzabteilung
- ☐ Personalmanagement
- ☐ Technologie-Entwicklung
- ☐ Sonstiges

1.2 In welcher Branche führen Sie Tätigkeiten zu EAM / Management der Anwendungslandschaft aus?

- ☐ Produzierendes Gewerbe
- ☐ Dienstleister
- ☐ Handel und Großhandel
- ☐ Landwirtschaft und Bergbau
- ☐ Finanzdienstleister
- ☐ Informationswirtschaft
- ☐ Versorgungsunternehmen
- ☐ Immobilienwirtschaft
- ☐ Verkehr und Logistik
- ☐ Sonstiges

1.3 Wo sind Sie angestellt?

- ○ bei dem Unternehmen, bei dem ich die obige Tätigkeit ausübe
- ○ bei einem externen Unternehmen

2 Ihr Hintergrund / Ihre Ausbildung

2.1 In welchem Bereich haben Sie eine Ausbildung / ein Studium absolviert?

- ☐ IT (z.B. Informatik)
- ☐ Business (z.B. Betriebswirtschaft)
- ☐ Schnittstelle zwischen IT und Business (z.B. Wirtschaftsinformatik)
- ☐ Philosophie, Psychologie
- ☐ Sozialwissenschaften
- ☐ Naturwissenschaften, Mathematik
- ☐ Ingenieurwissenschaften
- ☐ Sonstiges

2.2 Wie schätzen Sie Ihre IT-Kenntnisse ein?
1 (keine) bis 5 (sehr gut)

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ k.A.

2.3 Wie viele Jahre haben Sie bereits schon mit IT zu tun?

2.4 Wie schätzen Sie Ihre Kenntnisse bezüglich des Geschäfts, an dessen Unterstützung / Durchführung Sie beteiligt sind, ein?
1 (keine) bis 5 (sehr gut)

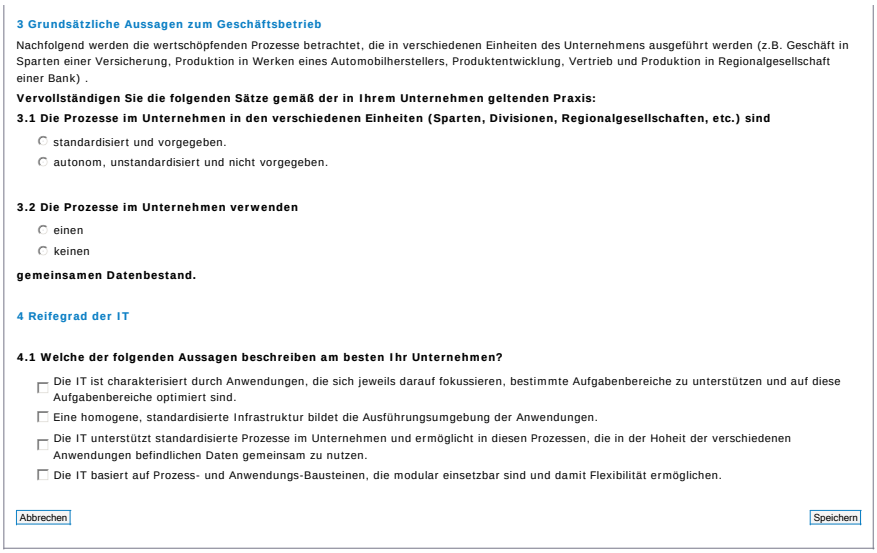
○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ k.A.

2.5 Wie viele Jahre arbeiten Sie bereits in diesem Bereich?

2.6 Wie schätzen Sie Ihren Kenntnisstand zu Enterprise Architecture Management ein?
1 (keine) bis 5 (sehr gut)

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ k.A.

2.7 Wie viele Jahre haben Sie bereits mit Enterprise Architecture Management / Management von Anwendungslandschaften zu tun?



3 Grundsätzliche Aussagen zum Geschäftsbetrieb

Nachfolgend werden die wertschöpfenden Prozesse betrachtet, die in verschiedenen Einheiten des Unternehmens ausgeführt werden (z.B. Geschäft in Sparten einer Versicherung, Produktion in Werken eines Automobilherstellers, Produktentwicklung, Vertrieb und Produktion in Regionalgesellschaft einer Bank) .

Vervollständigen Sie die folgenden Sätze gemäß der in Ihrem Unternehmen geltenden Praxis:

3.1 Die Prozesse im Unternehmen in den verschiedenen Einheiten (Sparten, Divisionen, Regionalgesellschaften, etc.) sind

- ○ standardisiert und vorgegeben.
- ○ autonom, unstandardisiert und nicht vorgegeben.

3.2 Die Prozesse im Unternehmen verwenden

- ○ einen
- ○ keinen

gemeinsamen Datenbestand.

4 Reifegrad der IT

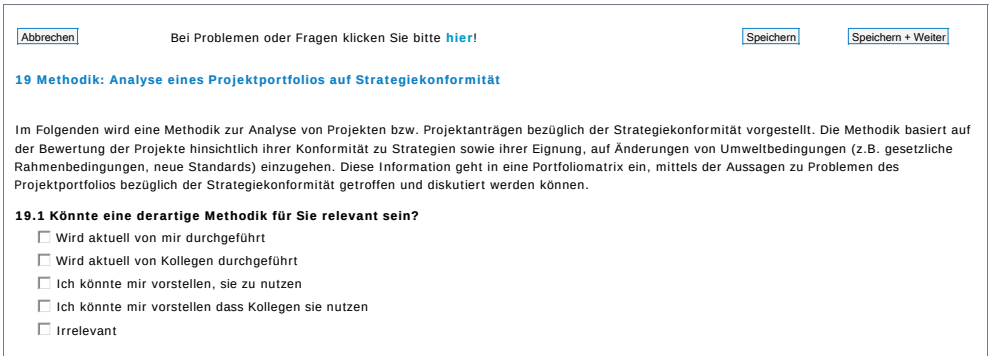
4.1 Welche der folgenden Aussagen beschreiben am besten Ihr Unternehmen?

- ☐ Die IT ist charakterisiert durch Anwendungen, die sich jeweils darauf fokussieren, bestimmte Aufgabenbereiche zu unterstützen und auf diese Aufgabenbereiche optimiert sind.
- ☐ Eine homogene, standardisierte Infrastruktur bildet die Ausführungsumgebung der Anwendungen.
- ☐ Die IT unterstützt standardisierte Prozesse im Unternehmen und ermöglicht in diesen Prozessen, die in der Hoheit der verschiedenen Anwendungen befindlichen Daten gemeinsam zu nutzen.
- ☐ Die IT basiert auf Prozess- und Anwendungs-Bausteinen, die modular einsetzbar sind und damit Flexibilität ermöglichen.

Abbrechen Speichern

A.2.1.2. Excerpt from EAMVS Online Questionnaire: Exemplary Methodology

Subsequently, an exemplary questionnaire form for a methodology (see Section 3.1.1.2.3 for an overview of the total structure of the online questionnaire) is shown, to demonstrate, how the majority of the online questionnaire was made up:



Abbrechen Bei Problemen oder Fragen klicken Sie bitte hier! Speichern Speichern + Weiter

19 Methodik: Analyse eines Projektportfolios auf Strategiekonformität

Im Folgenden wird eine Methodik zur Analyse von Projekten bzw. Projektanträgen bezüglich der Strategiekonformität vorgestellt. Die Methodik basiert auf der Bewertung der Projekte hinsichtlich ihrer Konformität zu Strategien sowie ihrer Eignung, auf Änderungen von Umweltbedingungen (z.B. gesetzliche Rahmenbedingungen, neue Standards) einzugehen. Diese Information geht in eine Portfoliomatrix ein, mittels der Aussagen zu Problemen des Projektportfolios bezüglich der Strategiekonformität getroffen und diskutiert werden können.

19.1 Könnte eine derartige Methodik für Sie relevant sein?

- ☐ Wird aktuell von mir durchgeführt
- ☐ Wird aktuell von Kollegen durchgeführt
- ☐ Ich könnte mir vorstellen, sie zu nutzen
- ☐ Ich könnte mir vorstellen dass Kollegen sie nutzen
- ☐ Irrelevant

Nachfolgend stellen wir Ihnen einen Viewpoint (Darstellung) vor, die eventuell geeignet ist, im Rahmen der Methodik genutzt zu werden.
Betrachten Sie bitte den Viewpoint, ohne dabei Details der graphischen Darstellung zu bewerten (spezielle Icons, Farbwahl, etc.).

19.2 Problemstellung: Sind die Maßnahmen, die Änderungen an der Anwendungslandschaft bewirken an den Strategien ausgerichtet? Dabei sollten auch z.B. finanzielle Aspekte und Vorgaben (z.B. Gesetze) berücksichtigt werden.

19.2.1 Inwieweit ist diese Problemstellung für Sie relevant?
1 (benötige ich nicht) bis 5 (brauche ich unbedingt)

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ k.A.

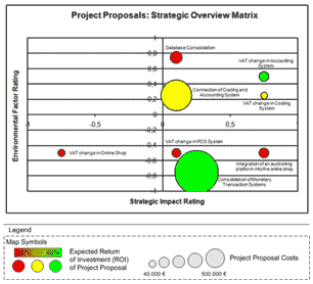
19.2.2 Führen Sie entsprechende Analysen aktuell durch?

- ○ Ja, bei Bedarf
- ○ Ja, täglich
- ○ Ja, wöchentlich
- ○ Ja, monatlich
- ○ Ja, quartalsweise
- ○ Ja, jährlich
- ○ Nein, Nutzung vorstellbar
- ○ Nein, Nutzung nicht vorstellbar
- ○ k.A.

19.2.3 Inwieweit eignet sich der Viewpoint 60 zur Beantwortung der Problemstellung?
1 (keine Hilfe) bis 5 (sehr große Hilfe)

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ k.A.

Viewpoint 60



19.3 Viewpoint 60 enthält auch Informationen zur Projektkosten („Investitionen in das Projekt") sowie der entsprechenden finanziellen Rendite („ROI").

19.3.1 Inwieweit ist diese Fragestellung für Sie relevant?
1 (benötige ich nicht) bis 5 (brauche ich unbedingt)

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ k.A.

19.4 Auf welchem Medium werden bzw. würden Sie diese Viewpoints verwenden?

- ☐ Diagrammanalyse allein am Bildschirm
- ☐ Ausdruck zur Weitergabe an Kollegen
- ☐ Großformatiger Ausdruck als Arbeitsunterlage für Meetings (Zeichnen im Ausdruck)
- ☐ Interaktive Bearbeitung am Bildschirm (Meeting, Webcast)
- ☐ Powerpoint-Präsentationen
- ☐ Publikation im Web
- ☐ k.A.

19.5 Welchen Aufwand erzeugt bzw. würde die Pflege der entsprechenden Daten bei Ihnen erzeugen? Unter anderem wären folgende Daten und Beziehungen zu pflegen:

- **Strategien / strategische Richtlinien**
- **Einflussfaktoren aus der Umwelt der Organisation (z.B. Gesetze, Richtlinien)**
- **Bewertung von Projekten / Projektvorschlägen im Bezug auf Strategien und Umwelteinflüsse**

Geben Sie an, welchen Aufwand sie zusätzlich zu anderen Datenpflege-Aktivitäten generiert.

- ○ vernachlässigbar klein < 1 Manntag/Jahr
- ○ bis 5 Manntage/Jahr
- ○ bis 10 Manntage/Jahr
- ○ bis 1 Mannmonat/Jahr
- ○ sehr hoch > 1 Mannmonat/Jahr
- ○ k.A.

Abbrechen | Speichern | Speichern + Weiter

A.2.2. Questionnaire Analysis Results

This Section shows detailed statistical evaluations performed on the online questionnaire data, which are referenced by the environment analysis in Section 3.1.2.

A.2.2.1. Analyzing Influences on the Attitude Towards Metrics

A.2.2.1.1. Influence of Education on MetricsUsage

Two analyses of the influence of the education received by the practitioners on their inclination to use metrics were conducted using the following variables, with their respective mapping to the questions in Table 3.2. For all nominal 1/0 variables which map to yes or no, yes is coded as 1, no as 0.

Variable	Level of measurement	Factor	Answer	Question
MetricsUsage	Ordinal	1	Usage not possible	IIX 3.1
		2	Usage possible	
		3	Yes	
EdIT	Nominal	1/0	IT (e.g., Computer Science) (Yes/No)	IX 2.1
EdBusiness	Nominal	1/0	Business (e.g., Business Administration) (Yes/No)	
EdBusinessIT	Nominal	1/0	Between Business and IT (e.g., Information Systems) (Yes/No)	
EdNat	Nominal	1/0	Natural Sciences or Mathematics (Yes/No)	
EdIng	Nominal	1/0	Engineering (Yes/No)	

The remaining options of question IX 2.1 (*Philosophy or Psychology*, *Social Sciences*, *Other*) have not been checked by any participating practitioner.

Full ordinal regression model, analyzing the influence of all kinds of education considered (see question IX 2.1) on *MetricsUsage* according to question IIX 3.1 (n=27):

Parameter Estimates

		Estimate	Std. Error	Wald	df	Sig.	95% Confidence Interval	
							Lower Bound	Upper Bound
Threshold	[MetricsUsage = 1.00]	-.226	3.175	.005	1	.943	-6.448	5.997
	[MetricsUsage = 2.00]	3.761	3.247	1.342	1	.247	-2.603	10.125
Location	[EdIT=.00]	3.143	1.491	4.442	1	.035	.220	6.066
	[EdIT=1.00]	0[a]	.	.	0	.	.	.
	[EdBusiness=.00]	-2.664	1.406	3.589	1	.058	-5.421	.092
	[EdBusiness=1.00]	0[a]	.	.	0	.	.	.
	[EdBusinessIT=.00]	2.438	1.545	2.490	1	.115	-.590	5.465
	[EdBusinessIT=1.00]	0[a]	.	.	0	.	.	.
	[EdNat=.00]	3.438	1.724	3.975	1	.046	.058	6.818
	[EdNat=1.00]	0[a]	.	.	0	.	.	.
	[EdIng=.00]	-1.904	1.672	1.297	1	.255	-5.181	1.373
	[EdIng=1.00]	0[a]	.	.	0	.	.	.

Link function: Logit.

a. This parameter is set to zero because it is redundant.

Warning in analysis: *There are 15 (50,0%) cells (i.e., dependent variable levels by combinations of predictor variable values) with zero frequencies.*

Reduced model, considering only the factors which appeared significant above (n=27):

Parameter Estimates

		Estimate	Std. Error	Wald	df	Sig.	95% Confidence Interval	
							Lower Bound	Upper Bound
Threshold	[MetricsUsage = 1.00]	-.278	1.271	.048	1	.827	-2.769	2.212
	[MetricsUsage = 2.00]	3.351	1.512	4.909	1	.027	.386	6.315
Location	[EdIT=.00]	2.549	1.021	6.236	1	.013	.548	4.550
	[EdIT=1.00]	0[a]	.	.	0	.	.	.
	[EdBusiness=.00]	-1.541	1.022	2.274	1	.132	-3.545	.462
	[EdBusiness=1.00]	0[a]	.	.	0	.	.	.
	[EdNat=.00]	2.727	1.239	4.845	1	.028	.299	5.156
	[EdNat=1.00]	0[a]	.	.	0	.	.	.

Link function: Logit.

a. This parameter is set to zero because it is redundant.

Warning in analysis: *There are 6 (33,3%) cells (i.e., dependent variable levels by combinations of predictor variable values) with zero frequencies.*

The warnings highlight, that the results have to be treated carefully. This is here done by using them more in an exploratory than in a confirmatory way.

A.2.2.1.2. Influence of Experience on MetricsUsage

Two analyses target the influence of the practitioners' experience on their inclination to use metrics. In addition to the variable *MetricsUsage*, defined as in Appendix A.2.2.1.1, the following variables are used (knowledge ratings surveyed on a five-point scale are treated as a metric variable):

- *ITKnowledge*: represents question IX 2.2. from Table 3.2 as a metric variable
- *BusinessKnowledge*: represents question IX 2.4 from Table 3.2 as a metric variable

- *EAMKnowledge*: represents question IX 2.6 from Table 3.2 as a metric variable
- *ITYears*: represents question IX 2.3 from Table 3.2 as a metric variable
- *BusinessYears*: represents question IX 2.5 from Table 3.2 as a metric variable
- *EAMYears*: represents question IX 2.7 from Table 3.2 as a metric variable

Ordinal regression model analyzing influence of knowledge on *MetricsUsage* (n=26):

Parameter Estimates

		Estimate	Std. Error	Wald	df	Sig.	95% Confidence Interval	
							Lower Bound	Upper Bound
Threshold	[MetricsUsage = 1.00]	-1.486	4.902	.092	1	.762	-11.095	8.122
	[MetricsUsage = 2.00]	1.792	4.943	.131	1	.717	-7.896	11.481
Location	ITKnowledge	-1.487	.902	2.717	1	.099	-3.256	.281
	BusinessKnowledge	1.198	.619	3.741	1	.053	-.016	2.412
	EAMKnowledge	.797	.534	2.228	1	.136	-.250	1.844

Link function: Logit.

Warning in analysis: *There are 27 (56,3%) cells (i.e., dependent variable levels by combinations of predictor variable values) with zero frequencies.*

Ordinal regression model analyzing the influence of years of experience on *MetricsUsage* (n=27):

Parameter Estimates

		Estimate	Std. Error	Wald	df	Sig.	95% Confidence Interval	
							Lower Bound	Upper Bound
Threshold	[MetricsUsage = 1.00]	-2.319	1.169	3.932	1	.047	-4.610	-.027
	[MetricsUsage = 2.00]	.793	1.034	.588	1	.443	-1.233	2.819
Location	ITYears	-.063	.081	.613	1	.434	-.222	.095
	BusinessYears	.206	.096	4.577	1	.032	.017	.395
	EAMYears	-.137	.139	.969	1	.325	-.410	.136

Link function: Logit.

Warning in analysis: *There are 54 (66,7%) cells (i.e., dependent variable levels by combinations of predictor variable values) with zero frequencies.*

Again, the warnings highlight, that the results have to be treated carefully. This is here done by using them more in an exploratory than in a confirmatory way.

A.2.2.1.3. Influence of Data Quality on MetricsUsage

The ordinal regression model for analyzing the influence of data quality on the inclination to use metrics relies on the variables for eduction *EdIt*, *EdBusiness*, *EdNat*, and the variable *MetricsUsage* as defined in Appendix A.2.2.1.1, together with a variable used for capturing satisfaction with data quality: *DataQualityComplaints*.

DataQualityComplaints is defined based on question IIX 2.5 from Table 3.2. With this question, the practitioners were able to check three kinds of complaints about data quality: *"No, they are not topical enough"*, *"No, correctness is insufficient"*, and *"No, they*

are not complete enough". DataQualityComplaints counts, how many of these complaints a practitioner indicated to apply. The variable is treated as ordinal.

Ordinal regression model analyzing the influence of education (question IX 2.1) and data quality on *MetricsUsage* (n=27):

Parameter Estimates

		Estimate	Std. Error	Wald	df	Sig.	95% Confidence Interval	
							Lower Bound	Upper Bound
Threshold	[MetricsUsage = 1.00]	.583	1.814	.103	1	.748	-2.971	4.138
	[MetricsUsage = 2.00]	4.434	2.079	4.550	1	.033	.360	8.508
Location	[EdIT=.00]	3.063	1.131	7.337	1	.007	.847	5.280
	[EdIT=1.00]	0[a]	.	.	0	.	.	.
	[EdBusiness=.00]	-1.102	1.169	.889	1	.346	-3.392	1.189
	[EdBusiness=1.00]	0[a]	.	.	0	.	.	.
	[EdNat=.00]	3.900	1.604	5.914	1	.015	.757	7.043
	[EdNat=1.00]	0[a]	.	.	0	.	.	.
	[DataQuality Complaints=.00]	2.509	2.952	.722	1	.395	-3.277	8.294
	[DataQuality Complaints=1.00]	-.933	1.174	.632	1	.427	-3.234	1.368
	[DataQuality Complaints=2.00]	-.778	1.291	.363	1	.547	-3.308	1.752
	[DataQuality Complaints=3.00]	0[a]	.	.	0	.	.	.

Link function: Logit.

a. This parameter is set to zero because it is redundant.

Warning in analysis: *There are 27 (56,3%) cells (i.e., dependent variable levels by combinations of predictor variable values) with zero frequencies.*

Thus, the data is used merely in an exploratory way.

A.2.2.1.4. Influence of the Application Landscape Size on MetricsUsage

The ordinal regression model for analyzing the influence of application landscape size on the inclination to use metrics relies on the variables for eduction *EdIt*, *EdBusiness*, *EdNat*, and the variable *MetricsUsage* as defined in Appendix A.2.2.1.1, together with a variable used for capturing application landscape size: *size*.

size is based on question IIX 1.4 from Table 3.2, aggregated to size classes as described in Section 3.1.1.2.2. Thus, *size* can take three values: 1 signifies small, 2 signifies medium, 3 signifies large. Due to this aggregation, *size* is treated as an ordinal variable.

Ordinal regression model analyzing the influence of *size* on *MetricsUsage* (n=20):

Parameter Estimates

		Estimate	Std. Error	Wald	df	Sig.	95% Confidence Interval	
							Lower Bound	Upper Bound
Threshold	[MetricsUsage = 1.00]	-3.615	2.330	2.407	1	.121	-8.181	.951
	[MetricsUsage = 2.00]	1.036	2.307	.202	1	.653	-3.486	5.558
Location	[EdIT=.00]	3.780	1.632	5.365	1	.021	.582	6.979
	[EdIT=1.00]	0[a]	.	.	0	.	.	.
	[EdBusiness=.00]	-2.635	1.564	2.838	1	.092	-5.702	.431
	[EdBusiness=1.00]	0[a]	.	.	0	.	.	.
	[EdNat=.00]	2.629	1.682	2.445	1	.118	-.666	5.925
	[EdNat=1.00]	0[a]	.	.	0	.	.	.
	[size=1.00]	-3.254	2.073	2.464	1	.116	-7.317	.809
	[size=2.00]	-3.483	2.390	2.124	1	.145	-8.167	1.201
	[size=3.00]	0[a]	.	.	0	.	.	.

Link function: Logit.

a. This parameter is set to zero because it is redundant.

Warning in Analysis: *There are 22 (61,1%) cells (i.e., dependent variable levels by combinations of predictor variable values) with zero frequencies.*

Thus, the data is used merely in an exploratory way.

A.2.2.1.5. Influence of Stakeholder Power and Number on MetricsUsage

The ordinal regression model for analyzing the influence of the number and power of stakeholders on the inclination to use metrics relies on the variables for eduction *EdIt*, *EdBusiness*, *EdNat*, and the variable *MetricsUsage* as defined in Appendix A.2.2.1.1, together with three dummy variables used for capturing stakeholder power and number according to question IIX 2.9 from Table 3.2. Thereby the dummy variables map to the answers possible with question IIX 2.9 as follows:

- *Different persons/groups try to set goals, but have no formal power for enforcing them*: StkhMultPowNo
- *A person/group (e.g., IT Architects) tries to set goals, but has no formal power for enforcing them*: StkhOnePowNo
- *A person/group with the necessary power for enforcing them sets goals*: StkhOnePowYes
- *Different persons/groups with the necessary formal power set goals*: Reference category

Ordinal regression model analyzing the influence of how many stakeholder try to set goals and have the power to set goals on MetricsUsage (n=27):

Parameter Estimates

		Estimate	Std. Error	Wald	df	Sig.	95% Confidence Interval	
							Lower Bound	Upper Bound
Threshold	[MetricsUsage = 1.00]	-.129	3.132	.002	1	.967	-6.267	6.009
	[MetricsUsage = 2.00]	3.871	3.270	1.402	1	.236	-2.538	10.280
Location	[EdIT=.00]	2.500	1.062	5.536	1	.019	.417	4.582
	[EdIT=1.00]	0[a]	.	.	0	.	.	.
	[EdBusiness=.00]	-1.484	1.108	1.793	1	.181	-3.656	.688
	[EdBusiness=1.00]	0[a]	.	.	0	.	.	.
	[EdNat=.00]	2.013	1.438	1.958	1	.162	-.806	4.832
	[EdNat=1.00]	0[a]	.	.	0	.	.	.
	[StkhMultPowNo=.00]	1.160	1.571	.545	1	.460	-1.919	4.239
	[StkhMultPowNo=1.00]	0[a]	.	.	0	.	.	.
	[StkhOnePowNo=.00]	.889	1.377	.417	1	.518	-1.810	3.588
	[StkhOnePowNo=1.00]	0[a]	.	.	0	.	.	.
	[StkhOnePowYes=.00]	-.984	1.399	.495	1	.482	-3.725	1.757
	[StkhOnePowYes=1.00]	0[a]	.	.	0	.	.	.

Link function: Logit.

a. This parameter is set to zero because it is redundant.

Warning in Analysis: *There are 24 (53,3) cells (i.e., dependent variable levels by combinations of predictor variable values) with zero frequencies.*

Thus, the data is used merely in an exploratory way.

A.2.2.1.6. Influence of Goal Explication on MetricsUsage

Examining the influence of goal explication on *MetricsUsage* introduces the variable *Goals*, which is defined based on question IIX 2.1 from Table 3.2. Thereby, the possible answers to question IIX 2.1 are represented by *Goals*, which is treated as an ordinal variable, as follows:

- *No goals are set*: $Goals = 1$
- *Goals are not explicitly documented, but distributed, possibly also in the heads of the respective employees*: $Goals = 2$
- *Precise goals exist in a document*: $Goals = 3$

The data did not allow an ordinal regression (due to a possible complete separation in the data), thus a simple table is presented here to support exploratory analysis, which here hints to an importance of having explicit goals when using metrics:

MetricsUsage * Goals Crosstabulation

Count

		Goals			
		1.00	2.00	3.00	Total
MetricsUsage	usageNotPossible	1	2	0	3
	usagePossible	0	10	3	13
	usingMetrics	0	2	8	10
Total		1	14	11	26

A.2.2.1.7. Influence of the Maturity in IT Usage on MetricsUsage

The ordinal regression model for analyzing the influence of maturity in IT usage on the inclination to use metrics relies on the variables for eduction *EdIt*, *EdBusiness*, *EdNat*, and the variable *MetricsUsage* as defined in Appendix A.2.2.1.1, together with a variable used for capturing maturity in IT usage: *Maturity*.

Maturity is based on question IX 4.1 from Table 3.2, where each answer is assigned a number:

- *IT is characterized by applications focused on supporting specific areas, for which it is optimized*: 1
- *A homogeneous, standardized infrastructure is the execution environment of the applications*: 2
- *IT supports standardized processes in the organization and enables these processes to use data owned by the different applications*: 3
- *IT is based on process- and application-components, which can be used in a modular way and thus provide flexibility*: 4

Maturity is then defined as the maximum number of the answers checked by the respective practitioner.

Ordinal regression model analyzing the influence of *Maturity* on *MetricsUsage* (n=26):

Parameter Estimates

		Estimate	Std. Error	Wald	df	Sig.	95% Confidence Interval	
							Lower Bound	Upper Bound
Threshold	[MetricsUsage = 1,00]	-2,991	2,164	1,911	1	,167	-7,232	1,250
	[MetricsUsage = 2,00]	,864	2,202	,154	1	,695	-3,452	5,179
Location	[EdIT=,00]	2,197	1,074	4,184	1	,041	,092	4,303
	[EdIT=1,00]	0[a]	.	.	0	.	.	.
	[EdBusiness=,00]	-1,752	1,121	2,442	1	,118	-3,950	,445
	[EdBusiness=1,00]	0[a]	.	.	0	.	.	.
	[EdNat=,00]	2,347	1,379	2,895	1	,089	-,356	5,051
	[EdNat=1,00]	0[a]	.	.	0	.	.	.
	[Maturity=1,00]	-2,107	1,523	1,915	1	,166	-5,092	,878
	[Maturity=2,00]	-2,980	1,860	2,567	1	,109	-6,624	,665
	[Maturity=3,00]	-2,300	1,577	2,128	1	,145	-5,390	,790
	[Maturity=4,00]	0[a]	.	.	0	.	.	.

Link function: Logit.

a. This parameter is set to zero because it is redundant.

Warning in Analysis: *There are 23 (51,1%) cells (i.e., dependent variable levels by combinations of predictor variable values) with zero frequencies.*

Thus, the data is used merely in an exploratory way.

A.2.2.1.8. Correlations between Maturity Level, Goal Explication and Application Landscape Size

To examine correlations between the variables *Maturity* (as introduced in Appendix A.2.2.1.7), *Goals* (as introduced in Appendix A.2.2.1.6), and *size* (as introduced in Appendix A.2.2.1.4), three analyses based on Spearman's rank correlation coefficient (Spearman's rho) are calculated. Spearman's rho is chosen, as the variables under consideration are ordinal.

Correlations

			Maturity	size
Spearman's rho	Maturity	Correlation Coefficient	1.000	.110
		Sig. (2-tailed)	.	.644
		N	27	20
	size	Correlation Coefficient	.110	1.000
		Sig. (2-tailed)	.644	.
		N	20	20

Correlations

			Maturity	Goals
Spearman's rho	Maturity	Correlation Coefficient	1.000	.138
		Sig. (2-tailed)	.	.500
		N	27	26
	Goals	Correlation Coefficient	.138	1.000
		Sig. (2-tailed)	.500	.
		N	26	26

Correlations

			Goals	size
Spearman's rho	Goals	Correlation Coefficient	1.000	.310
		Sig. (2-tailed)	.	.197
		N	26	19
	size	Correlation Coefficient	.310	1.000
		Sig. (2-tailed)	.197	.
		N	19	20

The low correlation coefficients, which are not indicated to be significant by the analyses, do not imply correlations between the three variables, especially not between maturity and goals, and between maturity and size, where such a correlation might be expected.

A.2.2.2. Analyzing Influences on the Relevant Granularity Level of Metrics

Three linear regression analyses examine the influence of the potential role of the answering practitioner (question IIX 3.3) and the size of the application landscape (question IIX 1.4) on the importance with which the three different granularity levels in question IIX 3.5.1 - 3.5.3 have been rated.

Thereby, the ratings for these granularity levels appear as the dependent variable in the respective regression model, interpreted as metric variables:

- *ObjLandscape*: question IIX 3.5.1 from Table 3.2
- *ObjDomain*: question IIX 3.5.2 from Table 3.2
- *ObjComponent*: question IIX 3.5.3 from Table 3.2

Four dummy variables were introduced to represent the potential role of the answering practitioner (question IIX 3.3):

- *Validating and checking data and metrics*: RoleValidation
- *Presenting metrics*: RolePresenting
- *Using the data for monitoring goals for projects/the application landscape*: RoleControlling
- *Planning metrics usage*: RolePlanning
- *Analysing data and metrics*: Reference category

The remaining answers to question IIX 3.3 were not considered, as none of the participating practitioners has chosen them.

The variable *size*, as introduced in Appendix A.2.2.1.4, is here represented by two dichotomous variables, in order to consider that *size* is no metric variable: *size2=1* indicates *size=2*, while *size3=1* indicates *size=3*.

Application landscape as a whole (ObjLandscape, n=14):

Coefficients[a]

Model		Unstandardized Coefficients		Standardized Coefficients	t	Sig.
		B	Std. Error	Beta		
1	(Constant)	3.500	.515		6.791	.000
	RoleValidation	-1.000	.893	-.287	-1.120	.295
	RolePresenting	-1.000	1.190	-.211	-.840	.425
	RoleControlling	-1.000	.893	-.287	-1.120	.295
	size2	1.000	.893	.287	1.120	.295
	size3	1.500	.787	.555	1.905	.093

a. Dependent Variable: ObjLandscape

Domains/Subgroups (ObjDomain, n=14):

Coefficients[a]

Model		Unstandardized Coefficients		Standardized Coefficients	t	Sig.
		B	Std. Error	Beta		
1	(Constant)	4.833	1.065		4.537	.003
	RoleValidation	-1.333	1.202	-.692	-1.109	.304
	RolePresenting	-1.167	1.324	-.446	-.881	.408
	RoleControlling	-.833	1.202	-.433	-.693	.510
	RolePlanning	-.500	.964	-.371	-.519	.620
	size2	-.333	.909	-.203	-.367	.725
	size3	.333	.642	.223	.519	.620

a. Dependent Variable: ObjDomain

Specific components of the application landscape (ObjComponent, n=15):

Coefficients[a]

Model		Unstandardized Coefficients		Standardized Coefficients	t	Sig.
		B	Std. Error	Beta		
1	(Constant)	4.750	.906		5.244	.001
	RoleValidation	-1.250	1.027	-.597	-1.217	.258
	RolePresenting	-1.500	1.152	-.526	-1.302	.229
	RoleControlling	.250	1.027	.119	.243	.814
	RolePlanning	-.500	.839	-.350	-.596	.567
	size2	-.250	.765	-.140	-.327	.752
	size3	.750	.523	.466	1.434	.189

a. Dependent Variable: ObjComponent

A.2.2.3. Analyzing Influences on which Kinds of Usage Scenarios are Relevant

A.2.2.3.1. Influence of Maturity

Below four regression analyses examine the influence of the potential role of the answering practitioner (question IIX 3.3) and the maturity in IT usage (question IX 4.1) on the importance with which the four different usage scenarios in questions IIX 3.6.1 - 3.6.4 have been rated.

Thereby, the ratings for these usage scenarios appear as the dependent variable in the respective regression model, interpreted as the following metric variables:

- *ScUnderstanding*: question IIX 3.6.1 from Table 3.2
- *ScPredicting*: question IIX 3.6.2 from Table 3.2
- *ScSetting*: question IIX 3.6.3 from Table 3.2
- *ScShowing*: question IIX 3.6.4 from Table 3.2

The remaining variables used in the analyses are the ones introduced in Appendix A.2.2.2, except *Maturity2*, *Maturity3* and *Maturity4*.

Maturity2, *Maturity3* and *Maturity4* are introduced to represent the maturity of IT , thereby considering the ordinal nature of this concept as measured by question IX 4.1. They are three dichotomous variables, linked to *Maturity* from Appendix A.2.2.1.7 as follows: *Maturity2=1* indicates *Maturity=2*, while *Maturity3=1* indicates *Maturity=3*, and *Maturity4=1* indicates *Maturity=4*.

Improve understanding of certain problems, to aid in finding a solution (ScUnderstanding, n=21):

Coefficients[a]

Model		Unstandardized Coefficients		Standardized Coefficients	t	Sig.
		B	Std. Error	Beta		
1	(Constant)	4.423	.791		5.594	.000
	RoleValidation	-1.423	1.118	-.440	-1.273	.225
	RolePresenting	-.703	1.044	-.217	-.673	.513
	RoleControlling	-.609	.952	-.224	-.640	.533
	RolePlanning	-.194	.763	-.102	-.254	.804
	Maturity2	-.423	1.370	-.095	-.309	.762
	Maturity3	-.828	.809	-.411	-1.024	.325
	Maturity4	-.441	.715	-.210	-.617	.548

a. Dependent Variable: ScUnderstanding

Predicting the effects of specific measures (ScPredicting, n=21):

Coefficients[a]

Model		Unstandardized Coefficients		Standardized Coefficients		
		B	Std. Error	Beta	t	Sig.
1	(Constant)	4.077	.645		6.316	.000
	RoleValidation	-1.077	.913	-.362	-1.180	.259
	RolePresenting	-.105	.852	-.035	-.123	.904
	RoleControlling	-.429	.777	-.172	-.552	.590
	RolePlanning	-.422	.623	-.241	-.677	.511
	Maturity2	.923	1.118	.225	.826	.424
	Maturity3	.713	.661	.385	1.079	.300
	Maturity4	.056	.583	.029	.096	.925

a. Dependent Variable: ScPredicting

Setting goals (as target values for specific metrics) and monitoring goal achievement (operationalizing architectural goals) (ScSetting, n=21):

Coefficients[a]

Model		Unstandardized Coefficients		Standardized Coefficients		
		B	Std. Error	Beta	t	Sig.
1	(Constant)	4.308	.550		7.833	.000
	RoleValidation	-.308	.778	-.124	-.396	.699
	RolePresenting	-.459	.726	-.185	-.632	.539
	RoleControlling	-.408	.662	-.196	-.617	.548
	RolePlanning	-.263	.531	-.181	-.496	.628
	Maturity2	.692	.953	.202	.727	.480
	Maturity3	.775	.563	.502	1.377	.192
	Maturity4	.302	.497	.187	.607	.554

a. Dependent Variable: ScSetting

Managing application landscape optimization: showing the status quo and potential for improvement (ScShowing, n=21):

Coefficients[a]

Model		Unstandardized Coefficients		Standardized Coefficients		
		B	Std. Error	Beta	t	Sig.
1	(Constant)	4.308	.505		8.528	.000
	RoleValidation	-1.308	.714	-.485	-1.831	.090
	RolePresenting	-.228	.667	-.085	-.342	.738
	RoleControlling	.746	.608	.330	1.226	.242
	RolePlanning	.198	.488	.125	.407	.691
	Maturity2	.692	.875	.186	.791	.443
	Maturity3	.237	.517	.141	.458	.655
	Maturity4	-.160	.456	-.091	-.350	.732

a. Dependent Variable: ScShowing

Above analysis might imply an influence of *RoleValidation* on *ScShowing*, however, such a result is discarded here: only one practitioner checked this category.

A.2.2.3.2. Influence of Application Landscape Size

Four linear regression analyses examine the influence of the potential role of the answering practitioner (question IIX 3.3) and the size of the application landscape (question IIX 1.4) on the relevance of different usage scenarios for metrics (question IIX 3.6.1-3.6.4). The variables are the ones introduced with Section A.2.2.3.1 and A.2.2.2 (*size2* and *size3*).

Improve understanding of certain problems, to aid in finding a solution (ScUnderstanding, n=15):

Coefficients[a]

Model		Unstandardized Coefficients		Standardized Coefficients	t	Sig.
		B	Std. Error	Beta		
1	(Constant)	3.750	1.089		3.445	.009
	RoleValidation	-.750	1.234	-.290	-.608	.560
	RolePresenting	-1.333	1.385	-.378	-.963	.364
	RoleControlling	-.750	1.234	-.290	-.608	.560
	RolePlanning	-1.000	1.008	-.567	-.992	.350
	size2	.250	.920	.114	.272	.793
	size3	1.583	.628	.796	2.519	.036

a. Dependent Variable: ScUnderstanding

Predicting the effects of specific measures (ScPredicting, n=15):

Coefficients[a]

Model		Unstandardized Coefficients		Standardized Coefficients	t	Sig.
		B	Std. Error	Beta		
1	(Constant)	5.500	1.966		2.798	.023
	RoleValidation	-2.500	2.229	-.719	-1.122	.295
	RolePresenting	-1.167	2.501	-.246	-.466	.653
	RoleControlling	-2.000	2.229	-.576	-.897	.396
	RolePlanning	-1.500	1.820	-.633	-.824	.434
	size2	-1.000	1.661	-.339	-.602	.564
	size3	-.333	1.135	-.125	-.294	.776

a. Dependent Variable: ScPredicting

Setting goals (as target values for specific metrics) and monitoring goal achievement (operationalizing architectural goals) (ScSetting, n=15):

Coefficients[a]

Model		Unstandardized Coefficients		Standardized Coefficients	t	Sig.
		B	Std. Error	Beta		
1	(Constant)	5.250	1.185		4.431	.002
	RoleValidation	-1.250	1.343	-.539	-.930	.379
	RolePresenting	-1.167	1.508	-.369	-.774	.461
	RoleControlling	-1.750	1.343	-.754	-1.303	.229
	RolePlanning	-.500	1.097	-.316	-.456	.661
	size2	-.750	1.001	-.380	-.749	.475
	size3	-.083	.684	-.047	-.122	.906

a. Dependent Variable: ScSetting

Managing application landscape optimization: showing the status quo and potential for improvement (ScShowing, n=15):

Coefficients[a]

Model		Unstandardized Coefficients		Standardized Coefficients	t	Sig.
		B	Std. Error	Beta		
1	(Constant)	5.000	1.161		4.305	.003
	RoleValidation	-2.000	1.317	-.782	-1.519	.167
	RolePresenting	-1.167	1.478	-.335	-.789	.453
	RoleControlling	-1.70E-015	1.317	.000	.000	1.000
	RolePlanning	-.500	1.075	-.287	-.465	.654
	size2	-.500	.982	-.230	-.509	.624
	size3	.167	.671	.085	.249	.810

a. Dependent Variable: ScShowing

APPENDIX B

Details of Models and Theories Underlying the Metrics

B.1. Metrics Relying on the Basic Information Model

B.1.1. Failure Propagation Structure of an Application Landscape

Section 4.3.1.1.1 introduces a set of Functions $F_{oi}(b)$, to represent the failure propagation structures in an application landscape. $F_{oi}(b)$ is here defined by providing an algorithm which gives for a specific Interface *oi* and the information which BusinessApplications have failed, and which not, (Vector <boolean> b, which contains for every position i whether BusinessApplication i is operational), whether *oi* is operational. The algorithm relies on the information model introduced in I-90 (Section 4.3.1).

The algorithm traverses the net made up by the failure propagation structure by recursively visiting the respective nodes. Thereby, OR-nodes are considered by evaluating the subnet of each alternative branch of an OR-node as a different *curEvalAlt* (currently evaluated alternative). The recursion is stopped if it reaches an Interface that has already been considered in the currently processed *curEvalAlt*. The algorithm starts by a call to *startCheck*:

```
startCheck(Interface oi, Vector<boolean> b){

  intialize checkedInterfaces with an empty set for each Interface;
  checkInterface(oi, b, checkedInterfaces, ''root'');

}
```

```
boolean checkInterface(Interface oi, Vector<boolean> b,
      Vector<Set> checkedInterfaces, String curEvalAlt){

  checkedInterfaces[oi].add(curEvalAlt);

  if(b[oi.offeringApplication]==false)
    return false;
  else if(oi.failureTree==null)
    return true;
  else
    return checkNode(oi.failureTree, b, checkedInterfaces, curEvalAlt);

}


booelan checkNode(Node n, Vector b<boolean>,
      Vector<Set> checkedInterfaces, String curEvalAlt){
  if(n instanceof Connector){
    if(checkedInterfaces[n.uses].contains(curEvalAlt)
      return true;
    else
      return checkInterface(n.uses);
  }
  else if (n instanceof OperatorNode){
    if(n.type==AND){
      for(toCheck:n.children)
        if(!checkNode(toCheck, b, checkedInterfaces, curEvalAltSet))
          return false

      return true;
    }
    else{  //default case OR
      for(toCheck:n.children){
        //unique Id for the currently checked branch of the OR-node
        String newCurEvalAltSet = generateUniqueId(n,toCheck);
        if(checkNode(toCheck, b, checkedInterfaces, newCurEvalAltSet))
          return true;
      }
      return false;
    }
  }
}
```

B.1.2. avSupportAvailability Formalized

Formalized, *fullSupportAvailability* of a BusinessProcess *bp* is the sum of the probabilities of all combinations of failing BusinessApplications, in which all Interfaces used by *bp* are operational:
$fullSupportAvailability(bp) =$
$\sum_{e \in \{b \in \mathbb{B}^{AppNr} | \bigwedge_{oi \in bp.processSupport} F_{oi}(b)=true\}} A^{|e|}(1-A)^{AppNr-|e|}$

Thereby, F_{oi} is the function formalized in Appendix B.1.1.

B.2. Metrics Relying on the Extended Information Model

B.2.1. serviceAvailability in the Extended Information Model

In order to consider the additional information introduced by using the extended information model (I-91), the definition of F_{oi} is adapted:

$F_{oi} : \mathbb{B}^{AppNr} \times \mathbb{B}^{ConnNr} \times \mathbb{B}^{IntNr} \times \mathbb{B}^{DepNr} \mapsto \mathbb{B}$

Thereby, the arguments of $F_{oi}(b, c, t, p)$ signify:

$b \in \mathbb{B}^{AppNr}$ is defined as with the basic information model: It assigns to each BusinessApplication i, whether it is operational or has failed. $AppNr$ is the number of BusinessApplications in the application landscape.

$c \in \mathbb{B}^{ConnNr}$ assigns to each Connector, whether it is operational ($c_i = true$) or has failed ($c_i = false$). $ConnNr$ is the number of Connectors in the application landscape.

$t \in \mathbb{B}^{IntNr}$ indicates for each Interface, whether it propagates a failure in case it is down and called by another BusinessApplication ($t_i = true$), or whether the failure is not propagated ($t_i = false$). $IntNr$ is the number of Interfaces in the application landscape.

$p \in \mathbb{B}^{DepNr}$ indicates for each Dependency in an application-internal failure tree, whether it propagates a failure ($p_i = true$), or not ($p_i = false$). $DepNr$ is the number of Dependencies in all BusinessApplications of the application landscape.

Now, $F_{oi}(b, c, t, p)$ is defined by an extension of the algorithm in Appendix B.1.1:

```
startCheck(Interface oi, Vector<boolean> b, Vector<boolean> c,
  Vector<boolean> t, Vector<boolean> p){
  intialize checkedInterfaces with an empty set for each Interface;
  checkInterface(oi, b, c, t, p, checkedInterfaces, ''root'');
}
```

```
boolean checkInterface(Interface oi, Vector<boolean> b,
      Vector<boolean> c, Vector<boolean> t, Vector<boolean> p
      Vector<Set> checkedInterfaces, String curEvalAlt){

  checkedInterfaces[oi].add(curEvalAlt);
  if(b[oi.offeringApplication]==false)
    return t[oi]==false;
  else if(oi.failureTree!=null&&
              checkNode(oi.failureTree, b, c, t, p, curEvalAlt)==false)
    return t[oi]==false;
  else
    return true;
}

booelan checkNode(Node n, Vector<boolean> b, Vector<boolean> c,
      Vector<boolean> t, Vector<boolean> p
      Vector<Set> checkedInterfaces, String curEvalAlt){

  if(n instanceof Connector){
    if(checkedInterfaces[n.uses].contains(curEvalAlt)
      return c[n];
    else
      return checkInterface(n.uses, b, c, t, p,
          checkedInterfaces, curEvalAlt)&&c[n];
  }
  else if (n instanceof OperatorNode){
    if(n.type==AND){
      for(toCheck:n.dependency)
        if(!checkNode(toCheck.children, b, c, t, p,
              checkedInterfaces, curEvalAltSet)&&p[toCheck])
          return false
      return true;
    }
    else{  //default case OR
      for(toCheck:n.dependency){
        //unique Id for the currently checked branch of the OR-node
        String newCurEvalAltSet = generateUniqueId(n,toCheck);
        if(checkNode(toCheck.children, b, c, t, p,
                  checkedInterfaces,newCurEvalAltSet)||!p[toCheck])
          return true;
      }
      return false;
    }
  }
}
```

When using this version of F_{oi} to define serviceAvailability for the extended information model, the equation has to be adapted to consider the occurrence probability of the respective combination of b,c,t and p. Assuming the respective events (Dependency propagates a failure or not, Interface propagates a failure or not, BusinessApplication fails, Connection fails) occur independently, this can be achieved as follows:

$$serviceAvailability(oi) = \sum_{(b,c,t,p)\in\{(b,c,t,p)\in\mathbb{B}^{AppNr}\times\mathbb{B}^{ConnNr}\times\mathbb{B}^{IntNr}\times\mathbb{B}^{DepNr}|F_{oi}(b,c,t,p)=true\}} prob(b,c,t,p)$$

Thereby, the function $prob(b,c,t,p)$ gives the occurrence probability of b,c,t and p, and is defined based on several helper functions:

- As already in Section 4.3.1.1.1, $|b|$ is here defined as $|b| = |\{i|b_i = true\}|, b \in \mathbb{B}^{AppNr}$, which is the number of positions b_i with $b_i = true$
- $true : \mathbb{B}^n \mapsto \mathcal{P}(\mathbb{N})$, with $true(x) = \{i|x_i = true\}$, which gives all indices, at which the respective Vector has a *true*-element
- Likewise, $false : \mathbb{B}^n \mapsto \mathcal{P}(\mathbb{N})$, with $false(x) = \{i|x_i = false\}$ gives all indices, at which the respective Vector has a *false*-element.

Subsequent helper functions give the probabilities of a failure event represented by Vectors b, c, t or p occurring. They do this by multiplying the respective probabilities. The attributes accessed to get these probabilities are from I-91 (Section 4.3.2).[1]

- $probB(b) = A^{|b|} \cdot (1-A)^{AppNr-|b|}$
- $probC(c) = \left(\prod_{i\in true(c)} i.connectionAvailability\right) \cdot \left(\prod_{i\in false(c)} (1 - i.connectionAvailability)\right)$
- $probT(t) = \left(\prod_{i\in true(t)} i.interfaceRealization.failurePropagationProbability\right) \cdot \left(\prod_{i\in false(t)} (1 - i.interfaceRealization.failurePropagationProbability)\right)$
- $probP(p) = \left(\prod_{i\in true(p)} i.propagationProbability\right) \cdot \left(\prod_{i\in false(p)} (1 - i.propagationProbability)\right)$

Now, $prob(b,c,t,p) = probB(b) \cdot probC(c) \cdot probT(t) \cdot probP(p)$ holds.

[1] The equations access these attributes using the respective indices which represent these objects in the corresponding Vectors. This notation is shorthand, a call to a function assigning the respective object to the index is left out to increase readability.

B.2.2. failureImpact in the Extended Information Model

Defining failureImpact in the extended information model relies first of all on an adapted "conditional serviceAvailability" (see Section 4.3.1.1.2 for the conditional serviceAvailability in the basic information model):

$$p(oi\ working \mid ba\ failed) = \frac{p(oi\ working \wedge ba\ failed)}{p(ba\ failed)} =$$

$$\frac{\sum_{(b,c,t,p)\in\{(b,c,t,p)\in\mathbb{B}^{AppNr}\times\mathbb{B}^{ConnNr}\times\mathbb{B}^{IntNr}\times\mathbb{B}^{DepNr}\mid F_{oi}(b,c,t,p)=true\wedge b_{ba}=false\}} prob(b,c,t,p)}{\sum_{(b,c,t,p)\in\{(b,c,t,p)\in\mathbb{B}^{AppNr}\times\mathbb{B}^{ConnNr}\times\mathbb{B}^{IntNr}\times\mathbb{B}^{DepNr}\mid b_{ba}=false\}} prob(b,c,t,p)}.$$

Interpreting, similar as with the basic information model, $\frac{\sum_{oi\in Interface} serviceAvailability(oi)}{|Interface|}$ as the expected share of operational Interfaces in the application landscape, and $\frac{\sum_{oi\in Interface} p(oi\ working \mid ba\ failed)}{|Interface|}$ as the share of operational Interfaces assuming *ba* has failed, failureImpact can be defined as:

$$failureImpact(ba) = \frac{\sum_{oi\in Interface}(serviceAvailability(oi) - p(oi\ working \mid ba\ failed))}{|Interface|}.$$

B.2.3. failureLoss Formalized

The formal definition of the *failureLoss* of a BusinessApplication *ba* is built on the LossFunctions and the conditional serviceAvailability as defined for the extended information model (see Appendix B.2.2):

$$failureLoss(ba) = \sum_{bp\in BusinessProcess}\sum_{ubp\in bp.usageByProcess}\Big[$$
$$ubp.failureLoss.getLossForFailedTime(1 - ubp.processSupport.p(working \mid ba\ failed) -$$
$$ubp.failureLoss.getLossForFailedTime(1 - ubp.processSupport.serviceAvailability)\Big].$$

APPENDIX C

Software Used for Metrics Calculation and Visualization

C.1. Exploring Options of Distributing Domains on Platforms: Proposal Generator Details

The Proposal Generator, as described in Section 5.3.3.1, walks through a the set of possible proposals distributing n Domains on k platforms, evaluating a sample of these proposals.

The Proposal Generator walks through the possible proposals by equivalence classes, which share the numbers of Domains deployed on the different platforms, e.g. for a four-platform proposal: one Domain on the first platform, three Domains on the second, five Domains on the third, and five Domains on the fourth.

These equivalence classes are generated as an array of length k, *platformSizes*, which stores on the i-th position the number of Domains deployed on the i-th platform, as follows:

1. *platformSizes* is initialized with zeros
2. *platformSizes[k-1]* is incremented
3. *platformSizes* is checked at each position i, starting at position k-1
   - if *platformSizes[i]* $> n$, and i==*0*, the algorithm terminates
   - if *platformSizes[i]* $> n$, and $i > 0$, *platformSizes[i-1]* is incremented
4. it is checked
   - whether each platform has received at least one Domain
   - whether the sum over all positions of platformSizes is exactly n (i.e. whether all Domains are planned to be deployed)
   - whether the $platformSizes[i] \leq platformSizes[i+1]$ holds for $i = 0...n-2$; This is checked to avoid duplicate equivalence classes, which differ only in the labels of the respective classes.[1]

   If all three checks are passed, the proposals in the respective equivalence class are generated.
5. resume at step 2

[1] An equivalence class (1, 2, 3, 4) can be transformed into one described by (2, 1, 3, 4) by just relabeling two platforms.

Generating all proposals in an equivalence class is achieved as follows:

```
platformAssignments is an array of size n, and is initialized with zeros;
  //Each position of the array signifies a Domain, and the   value at the
  // respective position indicates, on which platform this Domain
  //is deployed.
random is a random variable, which is true with
  probability sampleSize/totalNumberOfProposals

while(true){
  platformAssignments[n-1]++
  //check for overflows
  for(int position=n-1; position >=0; position--){
    if(platformAssignments[position] >= platformNumber){
      if(position==0)
        exit; //terminate the outermost loop
      else{
        platformAssignments[position]=0;
        platformAssignments[position-1]++;
      }
    }
  }
  //check, whether the assignment is valid in
    //the current equivalence class
  if not (number of assignments to each platform fit)
    continue;
  //avoid duplicate proposals, i.e. proposals which
  //can be transformed into each other by re-labeling platforms
  //this relies on the fact that the algorithm for
  //generating the equivalence classes supplies the platform
  //sizes with increasing platform size, and
  //transforming a platform into another by just re-labeling
  //relies on platforms of identical size
  for all series of equally sized platforms{
    for(int i=begin of series; i < end of series; i++){
      String lower=concatenation of Domain ids of platform i;
      String upper=concatenation of Domain ids of platform i+1;
      if(lower >= upper)
        continue with next proposal;
    }
  }
  //A valid proposal has been found
  //check whether it is taken into the sample
  if(random.nextRealization==true)
    processCurrentProposal;
}
```

In evaluating the proposals, the metrics calculator described in Section 5.2.2, configured to derive *serviceAvailability*, is used.

C.2. Calculating Failure Distributions and Modules at Risk

C.2.1. Runtime Complexity of Fully Calculating a Failure Distribution

Fully deriving a failure distribution, as done manually for a small example in Section 4.3.3.1.2, can be achieved by an algorithm which iterates over the classes of failure events with $i = 0..n$ Applications failing, and fully processes the failure events in each class:

```
n=number of Applications in the application landscape;
avy=assumed availability per Application;
affected[] contains for each Application the set of Modules failing
    if the Application fails;
moduleNumber is the number of Modules in the application landscape;
failureEventProbabilities[moduleNumber] is initialized with zeroes;

for(int i = 0; i <= n; i++){
  for(failureEvent in failureEvents with exactly i Applications failing){
    failingModules = emptyset;
    for(int j = 0; j < i; j++){
      failingModules = setUnification(failingModules,
          affected[failureEvent.getFailingApp(j)]);
          //getFailingApp(j) gets the j-th
          //failing Application of a failure event
    }
    failureEventProbabilities[|failingModules|] +=
        ((1-avy)^i)(avy^(n-i)); //probability of failure event
  }
}
```

The analysis of the runtime complexity of this algorithm relies on the equation giving the number of basic operations, here set unifications: $\sum_{i=0}^{n} \left(\binom{n}{i} i\right)$. Thereby, $\binom{n}{i}$ is the number of failure events in the class having i failing Applications, i is the number of set unification necessary to process such an event.

From there, the runtime complexity analysis makes the following approximation, giving an upper limit for the number of set unifications:

$\sum_{i=0}^{n} \left(\binom{n}{i} i\right) \leq n \sum_{i=0}^{n} \binom{n}{i} = n2^n$

Thus, it follows that for $u(n)$, the number of set unifications necessary to process an application landscape with n Applications, $u(n) \in O(n2^n)$ holds.

C.2.2. Details of Monte Carlo Simulation for Deriving a Failure Distribution

The Monte Carlo simulation, which derived the failure distributions used in the case study in Section 5.4.3, estimated the probabilities of a failure event being in one of 20 classes, each class being five per cent of the total Modules wide (see e.g. Figure 5.20).

The Monte Carlo simulation started with 500,000 randomly drawn sets of failing Applications. Then, the simulation kept adding batches of 500,000 randomly drawn sets, until it detected that this does no longer considerably change the estimated probabilities.

Formally, the stop criterion for the simulation sums up the absolute values of the changes a new batch of 500,000 sets of failing Applications makes to the probabilities estimated for the 20 classes. If this sum is 0.00001 or smaller, the algorithm stops. Thus, the stop criterion after adding the k-th batch is:

$\sum_{class=1}^{20} |oldProbabilities[class] - newProbabilities[class]| \leq 0.00001$, with oldProbabilities being the estimations after the $(k-1)$-th step, and newProbabilities the estimations after the k-th step.

Bibliography

[Ab95] Abreu, F. B.: *The MOOD Metric Set.* In *Proceedings of the ECOOP'95 Workshop on Metrics.* Springer, Aarhus, Denmark, 1995.

[AD05] Aier, S.; Dogan, T.: *Indikatoren zur Bewertung der Nachhaltigkeit von Unternehmensarchitekturen.* In (Ferstl, O. et al., Ed.): *7. Internationale Tagung Wirtschaftsinformatik*, pp. 607–626, Physica-Verlag, Bamberg, 2005.

[Ai07] Aier, S. et al.: *Deriving SOA Evaluation Metrics in an Enterprise Architecture Context.* In (Ahrens, M. et al., Ed.): *2nd International SeMSoC Workshop: Business Oriented Aspects concerning Semantics and Methodologies in Service-oriented Computing at the Fifth International Conference on Service Oriented Computing*, Springer, Vienna, Austria, 2007.

[An03] Andrews, T. et al.: *Business Process Execution Language for Web Services Version 1.1, 2003.* Technical Report, BEA Systems, International Business Machines Corporation, Microsoft Corporation, SAP AG, Siebel Systems, 2003.

[AT06] ATLAS group at LINA & INRIA:
ATL: Atlas Transformation Language. ATL User Manual, version 0.7. http://www.eclipse.org/m2m/atl/doc/ATL_User_Manual[v0.7].pdf (cited 2008-05-03). 2006.

[Ba98] Balzert, H.: *Lehrbuch der Software-Technik: Software-Management, Software-Qualitätssicherung, Unternehmensmodellierung.* Spektrum, Heidelberg, 1998. ISBN 3-8274-0065-1.

[Ba07] Baker, M. J.: *Marketing Strategy & Management.* 4th Edition, Palgrave Macmillan, Houndmills, 2007. ISBN 978-1-4093-8627-6.

[BD02] Bansiya, J.; Davis, C.: *A Hierarchical Model for Object-Oriented Design Quality Assessment. IEEE Transactions on Software Engineering*, Vol. 28(1), pp. 4–17, 2002.

[Be98] Bengtsson, P. O.: *Towards Maintainability Metrics on Software Architecture: An Adaptation of Object-Oriented Metrics.* In *Proceedings of the First Nordic Workshop on Software Architecture (NOSA'98)*, IEEE Computer Society, Ronneby (Sweden), 1998.

[Be04] Beyer, N.: *Kennzahlen zur Beschreibung von Anwendungslandschaften und ihre Visualisierung auf Softwarekarten.* Bachelor's thesis, Department of Informatics, Technische Universität München, 2004.

[Be08] Bergstrom, M. et al.: *Radical Cartography.* Website, 2008. http://www.radicalcartography.net/ (cited 2008-01-30).

[BL06] Bichler, M.; Lin, K.-J.: *Service-Oriented Computing: Composition and QoS Issues in Business Services Networks. IEEE Computer Society*, Vol. 39(3), pp. 99 – 101, 2006.

[BM03] Braun, P.; Marschall, F.: *BOTL - The Bidirectional Object Oriented Transformation Language.* Technical Report, Department of Informatics, Technische Universität München, 2003.

[BP00] Black, J. S.; Porter, L. W.: *Management – Meeting New Challenges.* Prentice Hall, Upper Saddle River, 2000. ISBN 0-321-01407-3.

[Br07] Braatz, B. et al.: *An approach using formally well-founded domain languages for secure coarse-grained IT system modelling in a real-world banking scenario.* In (Toleman, M.; Cater-Steel, A.; Roberts, D., Ed.): *18th Australasian Conference on Information Systems (ACIS'07)*, pp. 386–395, The University of Southern Queensland, Toowoomba, Australia, 2007.

[Br08] Brewer, C.: *ColorBrewer Intro – Selecting Good Color Schemes for Maps.* Website, 2008. http://www.personal.psu.edu/cab38/ColorBrewer/ColorBrewer_intro.html (cited 2008-01-30).

[BT00] Byrd, T.; Turner, D.: *Measuring the Flexibility of Technology Infrastructure: Exploratory Analysis of a Construct. Journal of Management Information Systems*, Vol. 17(1), pp. 167–208, 2000.

[Bu07a] Buckl, S. et al.: *A Pattern based Approach for constructing Enterprise Architecture Management Information Models.* In (Oberweis, A. et al., Ed.): *8. Internationale Tagung Wirtschaftsinformatik (Band 2)*, pp. 145–162, Universitätsverlag Karlsruhe, Karlsruhe, 2007.

[Bu07b] Buckl, S. et al.: *Generating Visualizations of Enterprise Architectures using Model Transformations (extended version). Enterprise Modelling and Information Systems Architectures - An International Journal*, Vol. 2(2), pp. 3–13, 2007.

[Bu08a] Buckl, S. et al.: *Enterprise Architecture Management Pattern Catalog (Version 1.0, February 2008)*. Technical Report TB 0801, Chair for Informatics 19 (sebis), Technische Universität München, 2008.

[Bu08b] Buckl, S. et al.: *Enterprise Architecture Management Trend Report*. Technical Report TB 0802, Chair for Informatics 19 (sebis), Technische Universität München, 2008 (in preparation).

[BW82] Basili, V. R.; Weiss, D. M.: *A Methodology for Collecting Valid Software Engineering Data*. Technical Report TR-1235, University of Maryland, 1982.

[BW05] Braun, C.; Winter, R.: *A Comprehensive Enterprise Architecture Metamodel and Its Implementation Using a Metamodeling Platform*. In (Desel, J.; Frank, U., Ed.): *Enterprise Modelling and Information Systems Architectures*, Vol. LNI P-75, pp. 64–79, Köllen Druck+Verlag, Klagenfurt, 2005.

[Ca08] Carden, T.: *Travel Time Tube Map*. Website, 2008. http://www.tomcarden.co.uk/p5/tube_map_travel_times/applet/ (cited 2008-01-31).

[CK94] Chidamber, S.; Kemerer, C.: *A Metrics Suite for Object Oriented Design*. *IEEE Transactions on Software Engineering*, Vol. 20(6), pp. 476–493, 1994.

[Cl02] Clements, P. et al.: *Documenting Software Architectures: Views and Beyond*. Addison Wesley, Boston, 2002. ISBN 0-2017-0372-6.

[CM06] CMMI Product Team: *CMMI® for Development, Version 1.2*. Technical Report CMU/SEI-2006-TR-008, Carnegie Mellon Software Engineering Institute, 2006.

[CM07] CMMI Product Team: *CMMI® for Acquisition, Version 1.2*. Technical Report CMU/SEI-2007-TR-017, Carnegie Mellon Software Engineering Institute, 2007.

[Cu05] Cukic, B.: *Guest Editor's Introduction: The Promise of Public Software Engineering Data Repositories*. *IEEE Software*, Vol. 22(6), pp. 20–22, 2005.

[De89] Dewey, M. et al.: *Dewey Decimal Classification and Relative Index*. Forest Press, a division of OCLC Online Computer Library Center, Albany, 1989. ISBN 0-910608-37-7.

[De06] Dern, G.: *Management von IT-Architekturen*. 2nd Edition, Vieweg, Wiesbaden, 2006. ISBN 978-3-528-15816-3.

[DFH03] Disterer, G.; Fels, F.; Hausotter, A., Ed. *Taschenbuch der Wirtschaftsinformatik*. Carl Hanser Verlag, München, 2003.

[DL05] Ducasse, S.; Lanza, M.: *The Class Blueprint: Visually Supporting the Understanding of Classes*. *IEEE Transactions on Software Engineering (TSE)*, Vol. 31(1), pp. 75 – 90, 2005.

[Do08] Dorling, D. et al.: *Worldmapper: The world as you've never seen it before – Total Population.* Website, 2008. http://www.worldmapper.org/display.php?selected=2 (cited 2008-01-30).

[Dr54] Drucker, P. F.: *The Practice of Management.* Harper & Brothers, New York, 1954.

[Dr77] Drucker, P. F.: *People and Performance.* Butterworth Heinemann, Oxford, 1977. ISBN 9-780750-625029.

[Dr99] Drukarczyk, J.: *Finanzierung.* 8th Edition, Lucius & Lucius, Stuttgart, 1999. ISBN 3-8252-1229-7.

[Du05] Dunsire, K. et al.: *The ABACUS Architectural Approach to Computer-Based System and Enterprise Evolution.* In *ECBS '05: Proceedings of the 12th IEEE International Conference and Workshops on the Engineering of Computer-Based Systems (ECBS'05)*, pp. 62–69, IEEE Computer Society, Washington, DC, USA, 2005.

[EE00] El Emam, K.: *A Methodology for Validating Software Product Metrics.* Technical Report NRC 44142, National Research Council of Canada, 2000.

[EGW07] Ende, B.; Gomber, P.; Wranik, A.: *An Order-Channel Management Framework for Institutional Investors.* In (Oberweis, A. et al., Ed.): *8. Internationale Tagung Wirtschaftsinformatik (Band 2)*, pp. 705–722, Universitätsverlag Karlsruhe, Karlsruhe, 2007.

[Ei87] Eichhorn, W. E.: *Measurement in Economics.* Physica-Verlag, Heidelberg, 1987.

[Ei89] Eisenhardt, K. M.: *Building Theories from Case Study Research. Academy of Management Review*, Vol. 14(4), pp.532–550, 1989.

[ELW06] Ernst, A.; Lankes, J.; Wittenburg, A.: *Tool Support for Enterprise Architecture Management - Strengths and Weaknesses.* In *The Tenth IEEE International EDOC Conference (EDOC 2006)*, pp. 13–22, IEEE Computer Society, Hong Kong, China, 2006.

[En08] Engels, G. et al.: *Quasar Enterprise. Anwendungslandschaften serviceorientiert gestalten.* Dpunkt Verlag, Heidelberg, 2008. ISBN 978-3-89864-506-5.

[EO99] Executive Office of the President, Office of Management and Budget: *North American Industry Classification System (NAICS).* Jist Works, 1999. ISBN 1-56370-537-0.

[ER03] Endres, A.; Rombach, D.: *A Handbook of Software and Systems Engineering. Empirical Observations, Laws and Theories.* Pearson Education, Harlow, England, 2003.

[Er06] Ernst, A. et al.: *Using Model Transformation for Generating Visualizations from Repository Contents - An Application to Software Cartography.* Technical Report TB 0601, Chair for Informatics 19 (sebis), Technische Universität München, 2006.

[ES90] ESRI: *ESRI Map Book. Volume Fourteen.* ESRI Press, Redlands, California, 1990.

[ES01] ESRI: *ESRI Map Book. Volume 16.* ESRI Press, Redlands, California, 2001.

[Fa99] Fahrmeir, L. et al.: *Statistik. Der Weg zur Datenanalyse.* Springer, Berlin, 1999.

[FHT96] Fahrmeir, L.; Hamerle, A.; Tutz, G.: *Multivariate statistische Verfahren.* 2nd Edition, Walter de Gruyter, Berlin, 1996.

[FP97] Fenton, N.; Pfleeger, S. L.: *Software Metrics. A Rigorous & Practical Approach.* Chapman & Hall, London, 1997.

[Fr02] Frank, U.: *Multi-Perspective Enterprise Modeling (MEMO) - Conceptual Framework and Modeling Languages.* In *35th Annual Hawaii International Conference on System Sciences (HICSS'02)*, Vol. 3, pp 1258–1267, IEEE Computer Society, Honolulu, 2002.

[Ge89] Gertsbakh, I.: *Statistical Reliability Theory.* Marcel Dekker, Inc, New York, 1989.

[GKC06] Garg, A.; Kazman, R.; Chen, H.-M.: *Interface descriptions for enterprise architecture. Science of Computer Programming*, Vol. 61(1), 2006.

[GL04] Gläser, J.; Laudel, G.: *Experteninterviews und qualitative Inhaltsanalyse als Instrument rekonstruierender Untersuchungen.* VS Verlag für Sozialwissenschaften, Wiesbaden, 2004.

[He04] Hevner, A. R. et al.: *Design Science in Information Systems Research. MIS Quarterly*, Vol. 28(1), pp. 75–105, 2004.

[He08] Hewlett-Packard Development Company, L.P.: *HP Discovery and Dependency Mapping software.* Hewlett-Packard Development Company, L.P., Website, 2008.
https://h10078.www1.hp.com/cda/hpms/display/main/hpms_-content.jsp?zn=bto&cp=1-11-15-25%5E767_4000_100__ (cited 2006-03-28).

[HGM02] Hake, G.; Grünreich, D.; Meng, L.: *Kartographie.* 8th Edition, Walter de Gruyter, Berlin, New York, 2002. ISBN 3-11-016404-3.

[HV99] Henderson, J.; Venkatraman, N.: *Strategic alignment: Leveraging information technology for transforming organizations. IBM Systems Journal*, Vol. 38(2 & 3), 1999.

[Ia05] Iacob, M.-E. et al.: *Architecture Analysis*. In (Lankhorst, M., Ed.): *Enterprise Architecture at Work*. Springer, Berlin, Heidelberg, New York, 2005.

[In06] Infosys Technologies Limited: *Infosys Enterprise Architecture Survey 2005*. http://www.infosys.com/IT-services/architecture-services/ea-survey/EA-survey-results.pdf (cited 2008-05-20), 2006.

[Is90] Ishikawa, K.: *Introduction to Quality Control*. Chapman & Hall, London, 1990.

[IS01] ISO/IEC: *Software engineering – Product quality – Part 1: Quality model*. Technical Report, ISO/IEC, 2001.

[IS03a] ISO/IEC: *Software engineering – Product quality – Part 2: External metrics*. Technical Report, ISO/IEC, 2003.

[IS03b] ISO/IEC: *Software engineering – Product quality – Part 3: Internal metrics*. Technical Report, ISO/IEC, 2003.

[IS04] ISO/IEC: *Software engineering – Product quality – Part 4: Quality in use metrics*. Technical Report, ISO/IEC, 2004.

[IT05] IT Governance Institute: *CobiT 4.0 - Control Objectives for Information and related Technology*. Information Systems Audit and Control Association, Rolling Meadows, USA, 2005. ISBN 1-933284-37-4.

[Je96] Jensen, K.: *Coloured Petri nets: Basis concepts, analysis methods and practical use*. Springer, Berlin, Heidelberg, 1996.

[Ju04] Junginger, M.: *Wertorientierte Steuerung von Risiken im Informationsmanagement*. Phd thesis, University of Hohenheim, 2004.

[Kü03] Kütz, M. E.: *Kennzahlen in der IT: Werkzeuge für Controlling und Management*. Dpunkt Verlag, Heidelberg, 2003.

[Kü05] Kütz, M.: *IT-Controlling für die Praxis: Konzeption und Methoden*. Dpunkt Verlag, Heidelberg, 2005.

[Ke02] Keller, W.: *Enterprise Application Integration*. Dpunkt Verlag, Heidelberg, 2002. ISBN 3-89864-186-4.

[Ke06] Keller, W.: *IT-Unternehmensarchitektur*. Dpunkt Verlag, Heidelberg, 2006. ISBN 3-89864-419-7.

[KM99] Klein, H. K.; Myers, M. D.: *A Set of Principles for Conducting and Evaluating Interpretative Field Studies in Information Systems. MIS Quarterly*, Vol. 23(1), pp. 67–94, 1999.

[KN91] Kaplan, R. S.; Norton, D. P.: *The Balanced Scorecard–Measures That Drive Performance. Harvard Business Review*, Vol. 70, pp. 71–79, 1991.

[KNS92] Keller, G.; Nüttgens, M.; Scheer, A.-W.: *Semantische Prozeßmodellierung auf der Grundlage "Ereignisgesteuerter Prozeßketten (EPK)"*. Technical Report Volume 89, Instituts für Wirtschaftsinformatik (IWi), Universität des Saarlandes, 1992.

[Kr71] Krantz, D. H. et al.: *Foundations of Measurement, Volume 1, Additive and Polynomial Representation*. Academic Press, New York, 1971.

[Kr05] Krcmar, H.: *Informationsmanagement*. 4th Edition, Springer, Berlin, 2005.

[La05] Lankhorst, M. E.: *Enterprise Architecture at Work*. Springer, Berlin, Heidelberg, New York, 2005. ISBN 3-540-23271-2.

[LD03] Lanza, M.; Ducasse, S.: *Polymetric Views - A Lightweight Visual Approach to Reverse Engineering. In IEEE Transactions on Software Engineering (TSE)*, Vol. 29(9), pp. 782–795. September 2003.

[Le07] Leitel, J.: *Entwicklung und Anwendung von Beurteilungskriterien für Enterprise Architecture Frameworks*. Master's thesis, Department of Informatics, Technische Universität München, 2007.

[Li92] Lindemann, C.: *Performance Analysis of Complex systems by deterministic and stochastic Petri Net models*. PhD thesis, Technische Universität Berlin, 1992.

[Li06] Livolsi, D. et al.: *Guided Architecture-based Design Optimisation of CBSs*. In *ECBS '06: Proceedings of the 13th Annual IEEE International Symposium and Workshop on Engineering of Computer Based Systems (ECBS'06)*, pp. 247–258, IEEE Computer Society, Washington, DC, USA, 2006.

[LKO97] Li-Thiao-Té, P.; Kennedy, J.; Owens, J.: *Mechanisms for Interpretation of OO Systems Design Metrics*. In *TOOLS '97: Proceedings of the Technology of Object-Oriented Languages and Systems-Tools - 24*, pp. 221–231, IEEE Computer Society, Beijing, China, 1997.

[LLS06] Laudon, K. C.; Laudon, J. P.; Schoder, D.: *Wirtschaftsinformatik – Eine Einführung*. Pearson Studium, München, 2006.

[LMW05a] Lankes, J.; Matthes, F.; Wittenburg, A.: *Softwarekartographie: Systematische Darstellung von Anwendungslandschaften*. In (Ferstl, O. K. et al., Ed.): *Wirtschaftsinformatik 2005*, pp. 1443–1462, Physica-Verlag, Bamberg, Germany, 2005.

[LMW05b] Lankes, J.; Matthes, F.; Wittenburg, A.: *Architekturbeschreibung von Anwendungslandschaften: Softwarekartographie und IEEE Std 1471-2000*. In (Liggesmeyer, P.; Pohl, K.; Goedicke, M., Ed.): *Software Engineering 2005*, Vol. P-64 of *Lecture Notes in Informatics (LNI)*, pp. 43–54. Köllen Druck+Verlag, Essen, Germany, 2005.

[Lo08] Longstreet Consulting Inc.: *Function Points Analysis Training Course*. http://www.softwaremetrics.com/Function Point Training Booklet New.pdf (cited 2008-05-05). 2008.

[LS02] Lewerentz, C.; Simon, F.: *Metrics-based 3D Visualization of Large Object-Oriented Programs.* In *Proceedings of the First IEEE International Workshop on Visualizing Software for Understanding and Analysis (VISSOFT 2002)*, pp. 70 – 77, IEEE Computer Society, 2002.

[LS08] Lankes, J.; Schweda, C. M.: *Using Metrics to Evaluate Failure Propagation in Application Landscapes.* In (Bichler, M. et al., Ed.): *Multikonferenz Wirtschaftsinformatik (MKWI) 2008*, pp. 1827–1838, GITO-Verlag, Berlin, Munich, 2008.

[LTC02] Lindvall, M.; Tesoriero, R.; Costa, P.: *Avoiding Architectural Degeneration: An Evaluation Process for Software Architecture.* In *Proceedings of the Eighth IEEE Symposium on Software Metrics (METRICS'02)*, pp. 77–86, IEEE Computer Society, Los Alamitos, CA, USA, 2002.

[LW04] Langenberg, K.; Wegmann, A.: *Enterprise Architecture: What Aspects is Current Research Targeting?* Technical Report EPFL Technical Report IC/2004/77, Ecole Polytechnique Fédérale de Lausanne, Laboratory of Systemic Modeling, 2004.

[Ma05a] Masak, D.: *Moderne Enterprise Architekturen.* Springer, Berlin, 2005.

[Ma05b] Maxwell, C. et al.: *Heuristic-Based Architecture Generation for Complex Computer System Optimisation.* In *12th IEEE International Conference and Workshops on the Engineering of Computer-Based Systems*, pp. 70–78, IEEE Computer Society, Greenbelt, USA, 2005.

[Ma08] Matthes, F. et al.: *Enterprise Architecture Management Tool Survey 2008.* Chair for Informatics 19 (sebis), Technische Universität München, 2008.

[Mc76] McCabe, T.: *A Complexity Measure. IEEE Transactions on Software Engineering*, Vol. 2(4), pp. 308–320, 1976.

[ME02] META Group: *Enterprise Architecture Desk Reference.* META Group, Inc., 2002.

[MS95] March, S. T.; Smith, G. F.: *Design and natural science research on information technology. Decision Support Systems*, Vol. 15, pp. 251–266, 1995.

[Mu05] Murer, S.: *Managed Evolution Grosser Informatiksysteme.* Ringvorlesung ETH/Uni Zürich, 2005.

[OGC00a] Office of Government Commerce (OGC): *ITIL - Service Delivery.* IT Infrastructure Library (ITIL). The Stationery Office, Norwich, UK, 2000. ISBN 0-11-330017-4.

[OGC00b] Office of Government Commerce (OGC): *ITIL - Service Support.* IT Infrastructure Library (ITIL). The Stationery Office, Norwich, UK, 2000. ISBN 0-11-330015-8.

[OM05a] OMG: *Unified Modeling Language: Infrastructure, version 2.0, formal/05-07-05.* Object Management Group, 2005.

[OM05b] OMG: *Unified Modeling Language: Superstructure, version 2.0, formal/05-07-04*. Object Management Group, 2005.

[OM06a] OMG: *Meta Object Facility (MOF) Core Specification, Version 2.0, formal/06-01-01*. Object Management Group, 2006.

[OM06b] OMG: *Object Constraint Language, version 2.0, formal/06-05-01*. Object Management Group, 2006.

[On07] O'Neill, T.: *PhD Thesis about ABACUS approach*. E-Mail Correspondence with Dr. Tim O'Neill, Avolution Pty. Ltd. 2007.

[Pr08] Price, M.: *Troux 7.1*. E-Mail Correspondence with Troux Technologies, Inc., 2008.

[Ro95] Robinson, A. H. et al.: *Elements of Cartography*. 6th Edition, John Wiley & Sons, Hoboken, USA, 1995. ISBN 0-471-55579-7.

[Ro02] Rosenkranz, F.: *Geschäftsprozesse – Modell- und computergestützte Planung*. Springer, Berlin, 2002.

[Ro03] Ross, J. W.: *Creating a strategic IT Architecture Compentency: Learning in Stages*. *MIS Quarterly Executive*, Vol. 2(1), 2003.

[RSD07] Rud, D.; Schmietendorf, A.; Dumke, R.: *Resource Metrics for Service-Oriented Infrastructures*. In (Lübke, D., Ed.): *Proceedings of the SEMSOA Workshop 2007 on Software Engineering Methods for Service-Oriented Architecture*, pp. 90–98, Leibniz Universität Hannover, FG Software Engineering, Hannover, Germany, 2007.

[RWJ02] Ross, S.; Westerfield, R.; Jaffe, J.: *Corporate Finance*. 6th Edition, McGraw-Hill, Boston, 2002.

[RWR06] Ross, J. W.; Weill, P.; Robertson, D. C.: *Enterprise Architecture as Strategy – Creating a Foundation for Business Execution*. Harvard Business School Press, Boston, Massachusetts, 2006. ISBN 978-1-591-39839-4.

[Se92] Sedgewick, R.: *Algorithmen in C++*. Addison Wesley, Boston, 1992. ISBN 3-89319-462-2.

[se05a] sebis: *Enterprise Architecture Management Tool Survey 2005*. Chair for Informatics 19 (sebis), Technische Universität München, 2005.

[Se05b] Sekatzek, P.: *Visualisierung von IT-Bebauungsplänen in Form von Softwarekarten - Konzeption und prototypische Umsetzung*. Diploma thesis, Department of Informatics, Technische Universität München, 2005.

[SK04] Schmid-Kleemann, M.: *Balanced Scorecard im IT-Controlling: Ein Konzept zur Operationalisierung der IT-Strategie bei Banken*. Schweizerische Kammer der Bücher-, Steuer- und Treuhandexperten, Zürich, 2004.

[Sl05] Slocum, T. A. et al.: *Thematic Cartography and Geographic Visualization.* 2nd Edition, Pearson Prentice Hall, Upper Saddle River, NJ, 2005. ISBN 0-13-035123-7.

[SS03] Sneed, H. M.; Sneed, S. H.: *Web-basierte Systemintegration – So überführen Sie bestehende Anwendungssysteme in eine moderne Webarchitektur.* Vieweg, Braunschweig, 2003. ISBN 3-528-05837-4.

[SS07] Schelp, J.; Stutz, M.: *A Balanced Scorecard Approach to Measure the Value of Enterprise Architecture.* In (Lankhorst, M.; Johnson, P., Ed.): *Proceedings of the Second Workshop on Trends in Enterprise Architecture Research (TEAR 2007)*, pp. 5–11, Via Nova Architectura, St. Gallen, Switzerland, 2007.

[St83] Stockman, N.: *Antipositivist theories of the sciences.* D. Reidel Publishing Company, Dordrecht, 1983.

[St99] Stäger, C.: *Multi Channel Management – Mehrdimensionale Optimierung der Kundenbeziehung zur nachhaltigen Steigerung der Profitabilität im Retail Banking.* Phd thesis, University of Zurich, 1999.

[St06] Street, N.: *TimeContours: Using isochrone visualisation to describe transport network travel cost.* PhD thesis, Imperial College London, 2006.

[SW05] Schwinn, A.; Winter, R.: *Entwicklung von Zielen und Messgrößen zur Steuerung der Applikationsintegration.* In *7. Internationale Tagung Wirtschaftsinformatik*, pp. 587–606, Physica-Verlag, Bamberg, 2005.

[TOG02] The Open Group: *TOGAF "Enterprise Edition" Version 8.1.* The Open Group, 2002. http://www.opengroup.org/architecture/togaf8-doc/arch/ (cited 2006-08-10).

[Tu90] Tufte, E. R.: *Envisioning Information.* Graphics Press, Cheshire, Connecticut, 1990.

[Tu01] Tufte, E. R.: *The Visual Display of Quantitative Information.* Graphics Press, Cheshire, Connecticut, 2001.

[USC00] Center for Software Engineering, USC.: *COCOMO II – Model Definition Manual.* Technical Report, Center for Software Engineering, USC, 2000.

[Ve05] Verlage, M. et al.: *Überwachung der Qualität der Architektur einer Software-Produktlinie am Beispiel eines web-basierten Wertpapierinformationssystems.* In (Liggesmeyer, P.; Pohl, K.; Goedicke, M., Ed.): *Software Engineering 2005.* Köllen Druck+Verlag, Essen, Germany, 2005.

[Wa03] Walter, M.: *Quantitative Bewertung hochverfügbarer Systeme.* PhD thesis, Technische Universität München, Munich. 2003.

[We08] Weisstein, E. W.: *"Stirling Number of the Second Kind." From MathWorld–A Wolfram Web Resource.* http://mathworld.wolfram.com/StirlingNumberoftheSecondKind.html (cited 2008-04-28). 2008.

[Wf05] The Workflow Management Coalition: *Workflow Management Coalition Workflow Standard: Process Definition Interface – XML Process Definition Language.* Technical Report WFMC-TC-1025, The Workflow Management Coalition, 2005.

[Wi07] Wittenburg, A.: *Softwarekartographie: Modelle und Methoden zur systematischen Visualisierung von Anwendungslandschaften.* Phd thesis, Technische Universität München, 2007.

[WM07] Wittenburg, A.; Matthes, F.: *Building an integrated IT governance platform at the BMW Group. International Journal of Business Process Integration and Management*, Vol. 2(4), pp. 327–337, 2007.

[Wo02] Woolridge, M. J.: *An introduction to multiagent systems.* John Wiley & Sons, Chichester, 2002.

[YZL07] Yu, T.; Zhang, Y.; Lin, K.-J.: *Efficient algorithms for Web services selection with end-to-end QoS constraints. ACM Transactions on the Web*, Vol. 1(1), 2007.