Assessing System Availability Using an Enterprise Architecture Analysis Approach

Jakob Raderius, Per Närman, and Mathias Ekstedt

Department of Industrial Information and Control Systems,
Royal Institute of Technology (KTH), Stockholm, Sweden
raderius@kth.se, {pern,mek101}@ics.kth.se

Abstract. During the last decade, a model based technique known as enterprise architecture has grown into an established approach for management of information systems in organizations. The use of enterprise architecture primarily promotes good decision-making and communication between business and IT stakeholders. This paper visualizes a scenario where enterprise architecture models are utilized to evaluate the availability of an information system. The problem is approached by creating models based on metamodels tailored to the domain of enterprise architecture analysis. As the instantiated models are fed with data particular to the information system, one can deduce how the organization needs to act in order to improve the system´s availability.

Keywords: Enterprise architecture, architecture analysis, Bayesian networks, decision graphs, availability.

1 Introduction

The discipline of Enterprise Architecture (EA) advocates using models to capture IT-systems, business processes and their interrelations. The purpose of these models is to visualize the business and IT in a way that abstracts from the complexity inherent in modern enterprise information systems thus facilitating IT management and ensuring business-IT alignment [1].

When faced with architectural changes, decision-makers can use EA models from an enterprise analysis framework for decision-support. To facilitate decision-making, the analysis framework must preserve and explicitly display the uncertainty of the analysis. Uncertainty is frequently abundant when performing architecture analysis. To remedy this, a formalism with which one is able to perform quantitative architecture analysis with information which may be uncertain, such as interview answers, was presented in [4]. This formalism is called Extended Influence Diagrams (EID).

In addition to the ability of capturing the uncertainty of the world, the EIDs are able to model causal properties of the world. This entails that the result of the architecture analysis will not only show which decision alternative is better with respect to the goal at hand, but also why this is so, i.e. which factors make one alternative better than its competitors.

G. Feuerlicht and W. Lamersdorf (Eds.): ICSOC 2008, LNCS 5472, pp. 351–362, 2009.

Using an EID for architecture analysis imposes information requirements on the architecture models, i.e. the models and in particular their underlying metamodels must answer the questions posed by the EID.

This paper describes an EID for availability assessment, how it was used together with associated metamodels for analysis of a set of system properties according to the ISO 9126 standard [6][7]. Specifically, the paper describes the application of the approach in a case study of a specific system at a Swedish company. The system in question is an enterprise data warehouse whose initial goal was to create an infrastructure not limited to current functional requirements of data, but could accommodate future changes in information structure due to a generic and extensible nature. Although the case study analyzed the system properties maintainability, availability; interoperability and functional suitability space limitations only permits us treating the availability assessment. For a full account readers are referred to [22].

The remainder of this paper consists of six sections. Next is a related works chapter which treats other approaches to IT system availability assessment. Section 3 gives an overview of the analysis framework, and its two main components; EIDs and metamodels. Section 4 is dedicated to the concept of quality of service and the quality attributes. Section 5 describes the case study and Sections 6 and 7 present the analysis and conclusion respectively.

2 Related Works

There are many methods for availability analysis. Commonly used methods to estimate availability include reliability block diagrams and Monte Carlo simulations [23]. All of these approaches are unable to i) express and manage input data uncertainty, and ii) rely heavily on the building of intricate models of, primarily, hardware architectures. In this case study, decision-makers had to rely on vague and incomplete information collected mainly through interviews – due to the lack of proper documentation - and it was therefore crucial to be able to express the level of certainty of the analysis.

As for ii), the scope of the present study made the creation of elaborate models of the system's architecture prohibitively expensive. Especially since detailed topologies of networks, nodes and other infrastructure documentation was more or less unavailable.

Due to i) and ii), the approaches to availability analysis as presented in [23] and as further elaborated in [24], [25] and [26] were not applicable in this case. Queuing models [27] and Petri nets [23] also require exact and complete data and must be discarded for the same reasons.

Further, iii), it was hinted by the stakeholders that the problems concerning the availability of the system are at least partly a result of problems within the organization operating it, and not simply the result of an inferior implementation.

The analysis approaches mentioned above are unable to perform any kind of organizational analysis, and are therefore insufficient with respect to iii).

3 A Quality of Service Evaluation Framework

This chapter will present the skeleton of a framework which can be used to create models that will answer a very specific set of questions about the quality of service of

enterprise information systems. The approach utilizes a model-based approach to achieve a level of abstraction sufficient to perform analyses concerning system qualities such as the functionality, availability, usability, performance, security and interoperability.

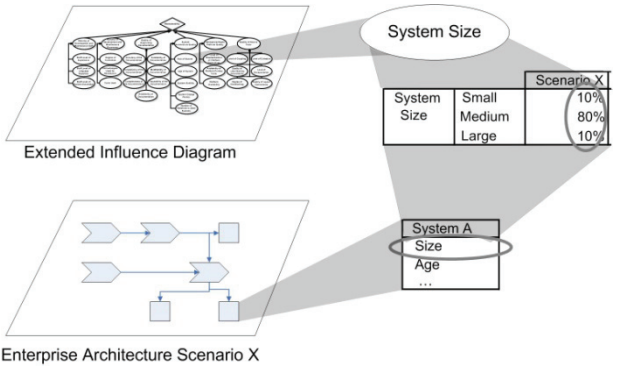

Fig. 1. To support architecture analysis, the architecture models must contain information specified by the analysis framework, here captured in the form of an extended influence diagram

Models are created with a specific goal in mind, such as increased system availability, and allow reasoning about the consequences of various scenarios with various modifications of variables. To guarantee that the models support system quality analyses, the underlying metamodel must contain at least the information required by the analysis frameworks for the respective system qualities. For example, if the goal is to analyze system maintainability the analysis framework might postulate that the analyst must determine the size of the system. If the analyst uses a model-based approach this means that the model must contain the property "system size". See Figure 1. Since the content of the architecture model is determined by their underlying metamodel, the model's metamodel must be closely coupled with the analysis frameworks.

In order to break down system characteristics into concrete and measurable questions suitable for analysis, we capture the characteristics using Extended Influence Diagrams (EID). These are graphic representations of decision problems and can be used to formally specify enterprise architecture analyses [4][7]. EIDs are an extension of influence diagrams, [18] [19] and feature the probabilistic inference engine taken from Bayesian networks [5][20], with which it is possible to reason about causal relations between different nodes through the means of Conditional Probability Tables.

Nodes of EIDs could in this context for example be "system coupling", or "system maintainability", and capture relations of the real world, such as "lower system coupling increases the system maintainability" through casual relationships. EIDs allow each node [7] to be broken down into sub-nodes until a sufficient level of tangibility adequate for data collection and analysis is achieved. To automate the

rather cumbersome probabilistic calculations of Bayesian networks there are several software tools available; see for instance GeNIe [21] which was used in this study.

Even though EIDs formally specify characteristics that influence a system and the causalities that exist between them they are in themselves insufficient to perform architecture analysis. To be able to create and measure different scenarios where those characteristics assume different states one needs a model better suited for analysis and use-case-specific instantiation. Class diagrams capture the entities of an architecture and their entity-relations and thus complement the EIDs that merely describe entity properties and their causal relations. The concept of *abstract models* [28] extends regular class diagram metamodels by layering an extended influence diagram on the metamodel. The nodes of the EIDs are represented as attributes of the metamodel and the causal relations between the attributes as *attribute relations*.

An abstract model thus contains the entities and attributes necessary to conduct analyses of a certain system. The entities can be extracted from EIDs and given attributes and relations based upon the stated causal relations [7].

3.1 Using the Framework

The analysis frameworks capture theories concerning architecture analysis. This theory is expressed in the abstract models comprising an EID and a metamodel. When using the approach, appropriate data is collected to create architecture models. The data collected is used to set the states of the attributes of the model entities. This information can be used as input to the Bayesian inference engine underlying the EIDs which makes the task of analysis straightforward. Values are propagated throughout the structure of the EIDs, and yield a quantitative result.

4 Quality of Service

The term quality of service resembles and is based on software quality as defined in the first part of the ISO-specification 9126 - 1 [6]. Its software quality metrics roughly correspond to the quality attributes of quality of service. The metrics found in ISO 9126 are, however, too low level, requiring massive amounts of data which make them inapplicable for the kind of high-level analyses that are performed here.

Quality of Service is defined in terms of four are stand-alone main quality attributes and four sub-attributes of the fifth main attribute; functionality. The **availability** of an information system is a measure of how often it is ready to deliver its services to its users [8]. The **usability** is a measure of how easy it is for a user to interact with and perform his or her tasks in the system [9]. The **efficiency** represents the degree to which a software system can meet its objective in terms of scalability and responsiveness [10]. Further**,** the **maintainability** represents the ease with which a software system or component can be modified to correct faults, improve performance or other attributes, or adapt to a changed environment [11]. The **accuracy** is one of four sub-attributes that comprise functionality, and represents the degree to which a system, given correct input data, produces output data that is both

accurate and precise [12]. The **security** – another functionality sub-attribute – is the system´s ability to maintain data confidentiality, integrity and availability [13].The **interoperability** represents the ability of two or more systems or components to exchange information and to use that information [11]. Last of the functionality sub-attributes, the **suitability** of an information system is the degree to which the functionality of the system supports its intended work tasks (i.e. the match between the system and its functional requirements) [14][15]. We now proceed to describe the breakdown of the attribute availability [22], where each major area influencing the availability of an information system is given its separate sub-heading. See Figure 2 below for an overview EID for availability assessment.

4.1 Quality of the Maintenance Organization

The **quality of the maintenance organization** is crucial to the availability of a system and this quality is dependent, among other things, on the maintenance organization´s **service level management** – the understanding, enforcement management and monitoring of the expected requirements and service levels [16].

In order to successfully manage these requirements and to enable a system-evolution that continuously upholds them, they have to be clearly understood. Only then can an enterprise information system be managed purposefully as part of a larger concept [17].

Since it is in most systems´ nature to evolve, **change management** is of utter importance, dependent first and foremost upon a change management process and its grade of use. This process includes change requests, as well as approval processes for these requests to ensure that sufficient levels of financial control are exerted [17].

Problem management - incident and problem detection and handling – is, of course, fundamental to correct operation and high uptime. Availability of appropriate resources is a must, as well as quick responsiveness of those resources.

4.2 Reliability

Reliability is the ability to perform under stated conditions for a stated period of time. Dependencies consist of redundancy, the quality of fault prediction and avoidance, and the reliability of sub-components.

Redundancy is essentially a mix of different fail-over solutions. The extension to which software redundancy solutions such as separate memory management and process isolation have been deployed plays a significant part, as well as hardware solutions such as backup disks, networks or complete fail-over servers [16].

When it comes to the **quality of fault prediction and avoidance** in the case study-system, a special subsystem is responsible for active and pro-active defense against faulty and/or incomplete data as well as quality assurance of incoming data deliveries. It does this by analyzing and benchmarking data against validation rules and descriptions of each data supplier´s format and alerting personnel when errors are encountered, even sending data for manual correction when the need arises.

Most systems are divided into **sub-components**, containing data tiers or runtime code, for example. Those are mostly interdependent in the sense that the whole system will not be working correctly without all of them.

For example, although the data receiving component of the system will still be operational if the validation component stops working, the system seen from a customer point of view will not be able to deliver the agreed service levels (in this case, possibly inferior quality data). Thus, the system´s reliability depends strongly upon the reliability of its sub-components [16].

4.3 Recoverability

The **recoverability** of a system is its ability to bypass and recover from a failure [16]. It is dependent upon automatic mechanisms that can be sprung into action as soon as something bad happens (failure with data loss etc.), as well as mechanisms that pro-actively monitor the system and generate the appropriate warning and/or take the appropriate action as soon as possible. Thus, the **quality of automatic fault recognition** and the **fastness of automatic recovery techniques** play important roles.

If failure occurs and leads to corrupt data, speed is of the essence. While nightly jobs creating backups and scripts to automatically restore data when necessary are all valid and fairly standard recovery techniques in the realm of enterprise data warehousing, it is the **fastness** of the initiation of the rollback of transactions gone wrong or the restoration of corrupted databases that matter the most.

4.4 Serviceability and Manageability

Serviceability is defined as the ability to perform effective problem determination, diagnosis and repair [16]. It is dependent upon the **quality of the system documentation** [16][17]. The serviceability of a system is highly dependent upon concise, up-to-date documentation, especially since the maintenance organization changes, and with it the personnel responsible for risen issues. When breaking down the quality of the system documentation into its constituents, one will find dependencies such as the availability, completeness and accuracy of said documentation, among other things.

No less important is the ability of the maintenance personnel to be able to understand what has gone wrong and where. **Understandability of error messages** is of paramount importance, as no incident management can take place if the nature of the incident is unknown [16].

Finally, **manageability**; the ability to create and maintain control over an environment in which the system works correctly, has a highly important dependability upon correct specifications on how the system works and behaves [16]. The continuous management of a system will no doubt see it evolve, and the organization has to ensure that it evolves in a controlled fashion.

An EID for availability analysis containing all the relevant aspects as listed above is shown in Figure 2.

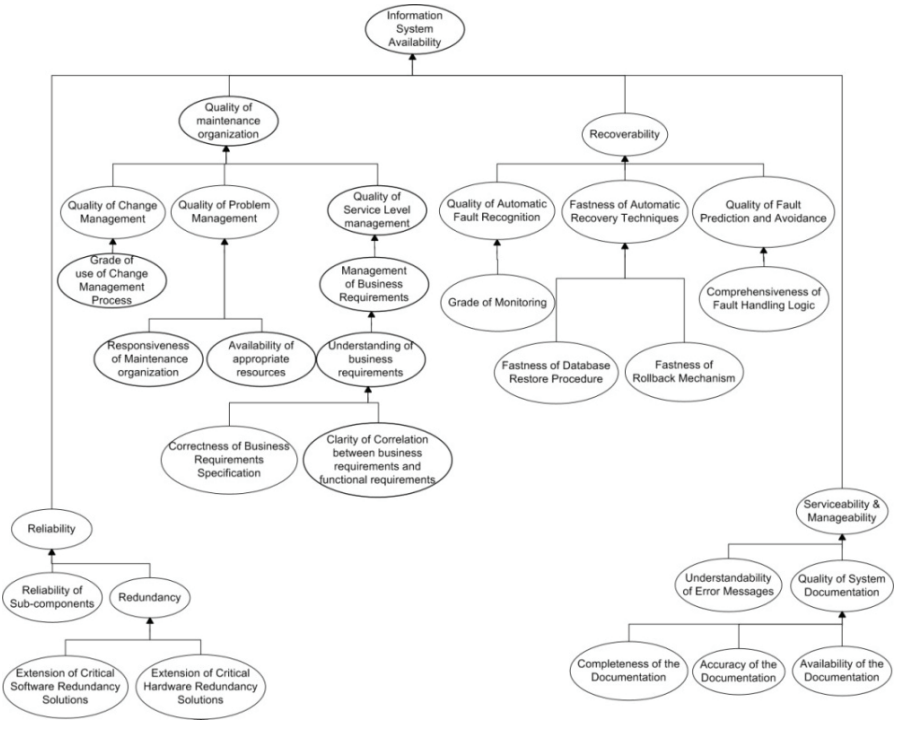

Fig. 2. An availability Extended Influence Diagram

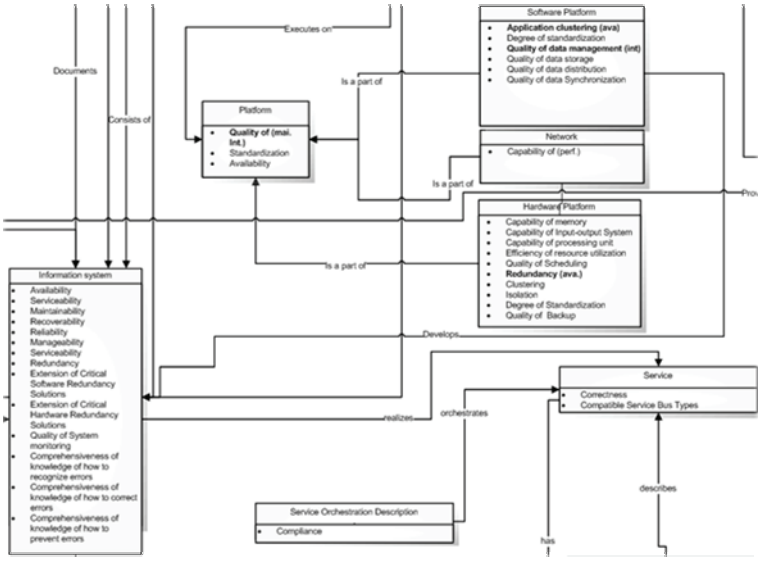

Fig. 3. Part of the metamodel for availability assessment

4.5 A Metamodel for Availability Assessments

A metamodel was derived based on the EID, the metamodel is very similar to the one presented in [7]. This metamodel is generic, containing classes such as **information system** and **hardware platform**. A small part of the metamodel for the case study system is shown in Figure 3. The metamodel can be found in its entirety in [22].

5 Case Study – Evaluating the Availability of a Data Warehouse

In order to analyze the availability of the enterprise data warehouse, data from the system and the organization was collected. Sources include documentation, logged statistics and interviewees. Questions are based on the theory captured in the EID presented in the previous section and the frequencies of the answers were mapped to discrete states to be useful for the Bayesian inference engine of the EID.

Figure 4 depicts a scenario where interviewees with varying degree of certainty and credibility (see [29] for an account of how to estimate source credibility) are asked questions related to change management, to facilitate an estimation of the grade of use of the change management process.

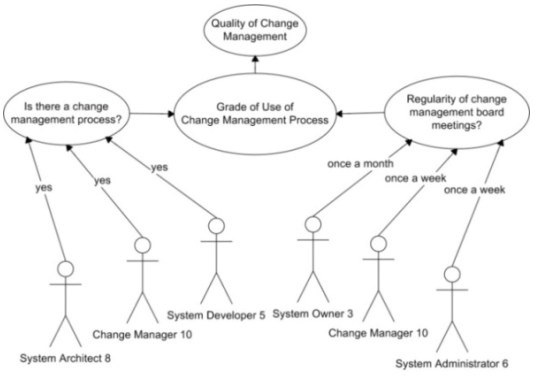

Fig. 4. Collecting data through interviews

As this data, shaped from answer frequencies and aptly modified for their source's credibility is then used in the Bayesian network tool GeNIe to perform the analysis. Collecting data using statistics or documentation is done in a similar fashion.

5.1 Instantiating the Models

By instantiating the metamodel into models it is possible to establish a detailed view of the system, its environment and the links between different parts of the platform and functions within the organization.

With such a model, one is able to reason clearly about how different entities in the system are coupled, and gather insight into how changes in variables will impact the system and the organization. Classes are instantiated using a notation similar to the

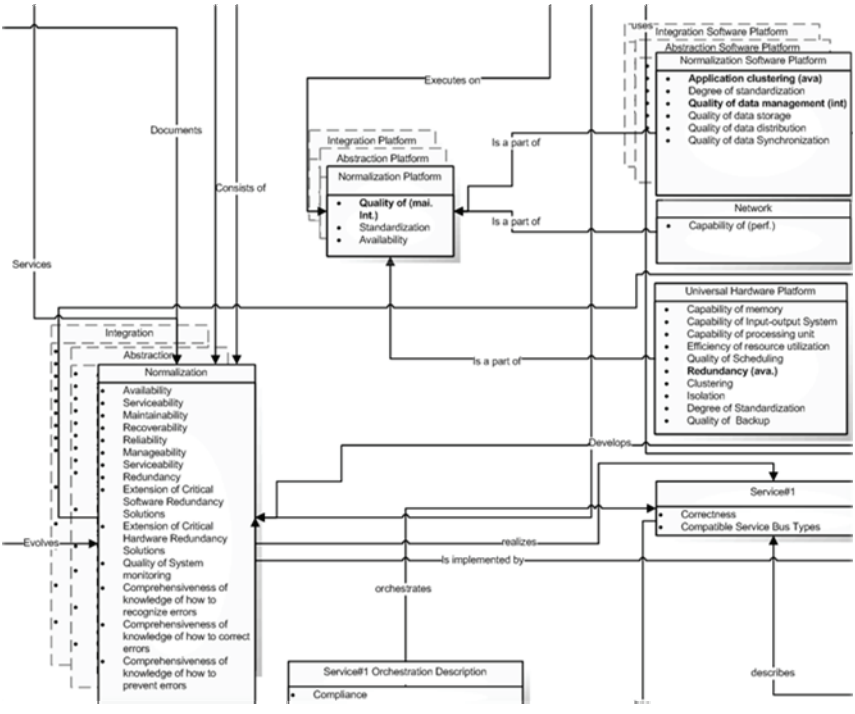

Fig. 5. Part of the model based on the availability metamodel

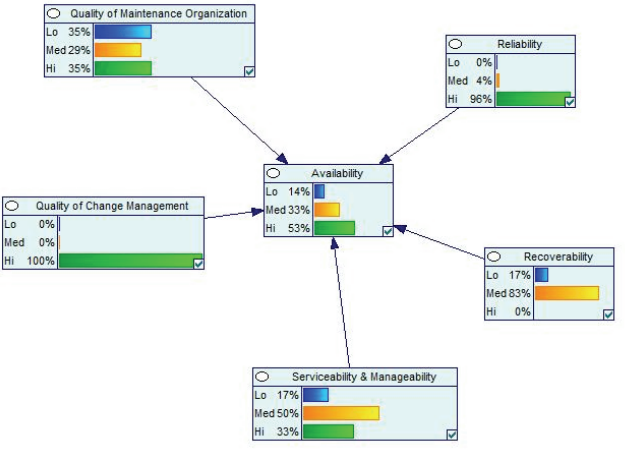

Fig. 6. Availability results

one used for the generic metamodel, though all instances except the top one use a crosshatched line to mark the border of the box representing it, see Figure 5 for an excerpt of the model of the evaluated system.

The instantiated model contains class instances for the three **layers** of the data warehouse, as well as separate instance for the **platforms** and **software platforms** used.

5.2 Results

Figure 6 shows the results for the availability assessment of the data warehousing solution. It should be interpreted as the probabilities that availability reaches a certain state. In this case, it is high with 53%, medium with 33% and low with 14% probability. As the intergroup weights of the different influences for this particular system are not known, weighted sampling distribution is used to distribute the proportionality of the influences in the GeNIe model evenly.

6 Analysis

According to the results displayed in Figure 6, one can say with 53% certainty that the availability of the case study system is high. High in this context translates to the 99.8-99.9% uptime necessary to uphold the organizations´ established service level agreements with customers. Thus, this quality attribute could certainly use some improvement.

The availability causalities found in the EID for availability (Figure 2) state that the quality of the maintenance organization, the Recoverability and the Serviceability & Manageability of the system are main influences and the GeNIe model states that they are 35%, 17% and 17% low respectively.

In the case of the maintenance organization one can tell from the GeNIe model that the low states originate in the lack of a business requirements specification and clarity of the correlation between business requirements and system functional requirements. Hence, a clear link between business concerns and IT should improve availability.

The recoverability is another source of concern, mainly because of a low quality of the fault prediction and avoidance system. This in turn is caused by a very limited set of fault handling logic implemented.

Suggestions for improvement include developing a monitoring and message handling application to be able to respond to incidents more effectively and efficiently, and implementing a sub-system that monitors the system and is able to handle software fatalities by running different command scripts. When it comes to Serviceability & Manageability, efforts to improve system documentation would greatly improve availability.

7 Conclusion

This paper has presented a case study where availability was assessed using an extended influence diagram – a form of Bayesian network - combined with an enterprise architecture metamodel. The metamodel makes it possible to create models with precisely the information that is needed for quantitative system quality analysis.

Furthermore, the paper discussed how a model for availability can be built from a theoretical foundation and then injected with empirical data in order to find the actual

availability of a data warehouse solution. The analysis provided suggestions for how improvement of certain aspects of the data warehouse and its surrounding organization can improve the system's availability.ï

References

1. Minoli, D.: Enterprise Architecture A to Z. s.l.: Auerbach (2008)
2. Zachman, J.A.: Concepts of the Framework for Enterprise Architecture (1992)
3. The Open Group Architecture Framework. TOGAF 8 Enterprise Edition. The Open Group, http://www.opengroup.org/togaf/
4. Johnson, P., et al.: Enterprise Architecture Analysis with Extended Influence Diagrams. Information System Frontiers, vol. 9. Springer, Netherlands (2007)
5. Jensen, F.: Bayesian Networks and Decision Graphs. Springer, New York (2001)
6. ISO/IEC. 9126-1 Software Engineering - Product Quality - Quality Model (2001)
7. Närman, P., Johnson, P., Nordström, L.: Enterprise Architecture: A Framework Supporting System Quality Analysis. In: Proceedings of the 11th International EDOC Conference (2007)
8. Hawkins, M., Piedad, F.: High Availability: Design,Techniques and Processes. Prentice Hall, Upper Saddle River (2001)
9. Nielsen, J.: Usability Engineering. Academic Press, San Diego (1993)
10. Smith, C.U., Williams, L.G.: Performance Solutions: A Practical Guide to Creating Responsive, Scalable Software. Pearson Education, Indianapolis (2001)
11. IEEE. IEEE Standard Glossary of Software Engineering Terminology (1990)
12. Redman, T.: Data Quality for the Information Age. Artech House, Norwood (1996)
13. Stoneburner, G.: Underlying Technical Models for Information Technology Security. National Institute of Standards and Technology, Gaithersburg (2001)
14. ISO/IEC. 9126-2 Technical Report - Software Engineering – Product Quality - Part 2: External Metrics (2003)
15. Sommeville, I., Sawyer, P.: Requirements Engineering. Wiley, Chichester (2004)
16. Johnson, P., Ekstedt, M.: Enterprise Architecture - Models and Analyses for Information Systems Decision Making. s.l.: Studentlitteratur (2007) ISBN 9789144027524
17. Addy, R.: Effective Service Management - To ITIL and Beyond. s.l. Springer, Heidelberg (2007)
18. Shachter, R.: Evaluating influence diagrams. Operations Research, vol. 34(6). Institute for Operations Research and the Management Sciences, Hanover Maryland (1986)
19. Howard, R.A., Matheson, J.E.: Decision Analysis. Influence Diagrams, vol. 2(3). Institute for Operations Research and the Management Sciences, Hanover Maryland (2005)
20. Neapolitan, R.: Learning Bayesian Networks. Prentice-Hall, Inc., Upper Saddle River (2003)
21. GeNIe & SMILE. GeNIe Website (2008), http://genie.sis.pitt.edu
22. Raderius, J.: Assessing the quality of service of an enterprise data warehouse. ICS, KTH, Stockholm (2008)
23. Trivedi, K., et al.: Achieving and Assuring High Availability. LNCS. Springer, Heidelberg (2008)
24. Sahner, R.A., Trivedi, K.S., Puliafito, A.: Performance and Reliability Analysis of Computer Systems. Kluwer Academic Press, Dordrecht (1996)
25. Trivedi, K.S.: Probability & Statistics with Reliability, Queueing and Computer Science Applications, 2nd edn. John Wiley, New York (2001)

26. Zhou, L., Held, M., Sennhauser, U.: Connection availability analysis of span-restorable mesh networks. Springer Science, Heidelberg (2006)
27. Grønbæk, J., et al.: Client-Centric Performance Analysis of a High-Availability Cluster (2007)
28. Johnson, P., et al.: A Tool for Enterprise Architecture Analysis. In: Proceedings of the 11th International EDOC Conference (2007)
29. Gammelgård, et al.: Architecture Scenario Analysis – Estimating the Credibility of the Results. In: Proceedings of the Seventeenth Annual International Symposium of The International Council on Systems Engineering (2007)