A Tool for Enterprise Architecture Analysis using the PRM formalism

Markus Buschle, Johan Ullberg, Ulrik Franke, Robert Lagerström, and Teodor Sommestad

Industrial Information and Control Systems, KTH Royal Institute of Technology, Osquldas v. 12, SE-10044 Stockholm, Sweden
{markusb, johanu, ulrikf, robertl, teodors}@ics.kth.se,

Abstract. Enterprise architecture advocates model-based decision-making on enterprise-wide information system issues. In order to provide decision-making support, enterprise architecture models should not only be descriptive but also enable analysis. This paper presents a software tool, currently under development, for the evaluation of enterprise architecture models. In particular, the paper focuses on how to encode scientific theories so that they can be used for model-based analysis and reasoning under uncertainty. The tool architecture is described, and a case study shows how the tool supports the process of enterprise architecture analysis.

Keywords: Enterprise Architecture, Probabilistic relational Models, Software tool, Data Quality

1 Introduction

Over the last two decades, enterprise architecture has grown into an established approach for holistic management of information systems in organizations [1, 2]. A number of enterprise architecture initiatives have been proposed, such as The Open Group Architecture Framework (TOGAF) [3], the Zachman Framework [4], and military architectural frameworks such as DoDAF [5] and NAF [6]. The core concept of the enterprise architecture approach is the employment of models. Diagrammatic descriptions of IT systems and their environment are heavily used. However, enterprise architecture models are not limited to descriptive use only, but can also be employed to predict the behavior and effects of decisions. Rather than modifying enterprise information systems using trial and error, models allow predictions about the behavior of future architectures.

One prominent challenge to rational decision making is uncertainty. Therefore, a good enterprise architecture model should be able to capture uncertainties about assessment theory, system configuration or data quality, thus providing better decision support and risk management.

What constitutes a "good" enterprise architecture model is dependent on its purpose, i.e. the type of analysis it is intended to support [7]. For instance in

the case of analyzing data quality, the property of whether the data objects are accurate with respect to the real world they describe is of interest. This property however, is irrelevant for a number of other analyses, such as performance evaluation.

Several enterprise architecture software tools are available on the market, including Metis [8], System Architect [9] and Aris [10]. These tools generally focus on the modeling of an architecture whereas the analysis functionality is generally limited to performing an inventory or to sum costs over the modeled architecture. None of the mentioned tools has significant capabilities for system quality analysis based on an elaborated theory. Furthermore, these tools do not support the consideration of uncertainty as described above.

In this paper an enterprise architecture software tool is presented. This tool does not only provide functionality to model enterprise architectures, but also supports the analysis of them. In order to support enterprise architecture analysis as it has been outlined in [7] the tool consists of two main components. In the first component the theory relevant to analyze a certain system quality, such as data quality or modifiability, is modeled. One can consider this as the definition of a language tailored to describe a certain aspect, e.g. data quality. The second component supports the application of the theory to evaluate a specific enterprise architecture. This is done by modeling the "as-is" or "to-be" architecture of the enterprise. Based on the created models it is possible to determine how the architecture fulfills the requirements as they have been defined in the theory. The two-component architecture encourages the reuse of the developed theory as it is possible to use the same language to describe several architecture instances. The presented tool makes use of the Probabilistic Relation Models (PRM) formalism as it has been presented in [11] and can thereby manage the uncertainty aspects discussed above.

2 Enterprise Architecture Analysis

Enterprise architecture models have several purposes. Kurpjuweit and Winter [12] identify three distinct modeling purposes with regard to information systems, viz. (i) documentation and communication, (ii) analysis and explanation and (iii) design. The present article focuses on the analysis and explanation since this is necessary to make rational decisions about information systems [7]. An analysis-centric process of enterprise architecture is illustrated in Fig. 1. In the first step, assessment scoping, the problem is described in terms of one or a set of potential future scenarios of the enterprise and in terms of the assessment criteria with its theory (the PRM in the figure) to be used for scenario evaluation. In the second step, the scenarios are detailed by a process of evidence collection, resulting in a model (instantiated PRM, in the figure) for each scenario. In the final step, analysis, quantitative values of the models' quality attributes are calculated, and the results are then visualized in the form of e.g. enterprise architecture diagrams.

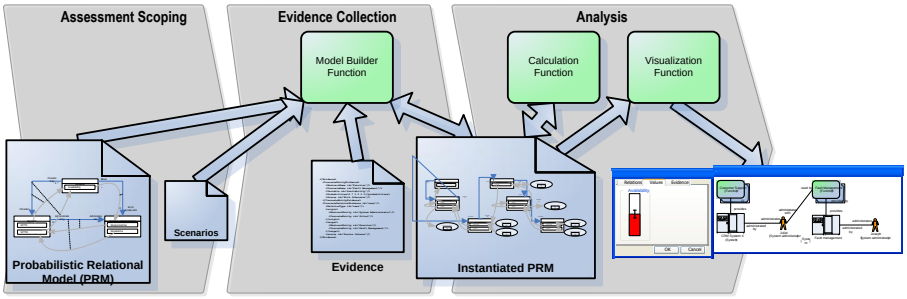

Fig. 1. The process of enterprise architecture analysis with three main activities: (i) setting the goal, (ii) collecting evidence and (iii) performing the analysis.

More concretely, assume that a decision maker in an electric utility is contemplating changes related to the maintainance of the power grid. The introduction of an maintenance management system would improve the quality of the maintenance process and allow more cost efficient grid maintenance. The question for the decision maker is whether this change is feasible or not.

As mentioned, in the first step, *assessment scoping*, the decision maker identifies the available decision alternatives, i.e. the enterprise information system scenarios. In this step, the decision maker also needs to determine how the scenario should be evaluated, i.e. the goal of the assessment. One such goal could be to assess the data quality of an information system. Other goals could be to assess the availability [13], interoperability [14] or security [15] of the proposed to-be architecture. Often several quality attributes are desirable goals. In this paper, without loss of generality, we simplify the problem to the assessment of data quality in the maintenance process.

Information about the involved systems and their organizational context is required for a good understanding of their data quality. For instance, it is reasonable to believe that a more precise data object attribute would increase the probability that the data quality of an information system is high. The impact of a certain data object attribute is thus one factor that can affect the data quality and should therefore be recorded in the scenario model. The decision maker needs to understand what information to gather, and also ensure that this information is indeed collected and modeled. Overall, the effort aims to understand which attributes causally influence the selected goal, viz. data quality. It might happen that the attributes identified do not directly influence the goal. If so, an iterative approach can be employed to identify further attributes causally affecting the attributes found in the previous iteration. This iterative process continues until all paths of attributes, and causal relations between them, have been broken down into attributes that are directly controllable for the decision maker [16].

In the second step, *collecting evidence*, the scenarios need to be detailed with actual information to facilitate analysis of them. Thus, once the appropriate

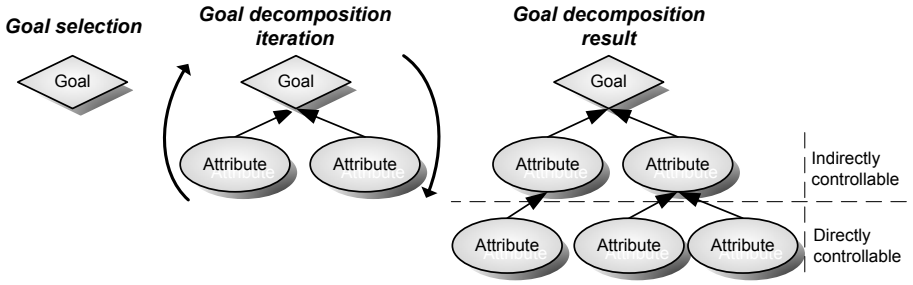

Fig. 2. . Goal decomposition method, from [16].

attributes have been set, the corresponding data is collected throughout the organization. In particular, it should be noted here that the collected data will not be perfect. Rather, it risks being incomplete and uncertain. The tool handles this, by allowing the user to enter the credibility of the evidence, depending on how large the deviations from the true value are judged to be.

In the third and final step, *performing the analysis*, the decision alternatives are analyzed with respect to the goal set (e.g. data quality). The mathematical formalism to be presented in section 2.1 plays a vital role in this analysis. Using conditional probabilities and Bayes' rule, it is possible to infer the values of the variables in the goal decomposition under different architecture scenarios [17]. By using the PRM formalism, the architecture analysis accounts for two kinds of potential uncertainties: that of the attribute values as well as that of the causal relations as such. Using this analysis framework, the pros and cons of the scenarios can be traded against each other in order to determine which alternative ought to be preferred.

2.1 Probabilistic Relational Models

A *probabilistic relational model* (PRM) [18] specifies a template for a probability distribution over an architecture model. The template describes the metamodel for the architecture model, and the probabilistic dependencies between attributes of the architecture objects. A PRM, together with an instantiated architecture model of specific objects and relations, defines a probability distribution over the attributes of the objects. The probability distribution can be used to infer the values of unknown attributes, given evidence of the values of a set of known attributes. PRMs are related to Bayesian Networks. As it is succinctly put in [11], PRMs "are to Bayesian networks as relational logic is to propositional logic".

A PRM model may be instantiated as a *relational skeleton*, σ_r, containing just objects, object relationships, and attributes. Furthermore, a *qualitative dependency structure* $\mathcal{S}$ defines the details of the attribute relationships, i.e. the sets of probabilistic parents influencing each attribute. Finally, the PRM is completed by the set of *parameters* $\theta_{\mathcal{S}}$ specifying the full conditional probabilistic dependencies between attributes in the form of numbers in Conditional Prob-

ability Matrices (CPM). The following expression thus defines the conditional probability of an instance $\mathcal{I}$, given σ_r, $\mathcal{S}$, and $\theta_{\mathcal{S}}$:

$$P(\mathcal{I}|\sigma_r, \mathcal{S}, \theta_{\mathcal{S}}) = \prod_{x \in \sigma_r} \prod_{A \in \mathcal{A}(x)} P(\mathcal{I}_{x.A}|\mathcal{I}_{Pa(x.A)})$$

$$= \prod_{X_i} \prod_{A \in \mathcal{A}(X_i)} \prod_{x \in \sigma_r(X_i)} P(\mathcal{I}_{x.A}|\mathcal{I}_{Pa(x.A)})$$

Compared to the standard chain rule for Bayesian networks, this equation is different in three ways: (i) the random variables are the attributes of a set of objects, (ii) the parents of a random variable depend on the model context of the object, and (iii) the parameters are shared between the attributes of objects in the same class. In other words, the variables in the dependency structure are the properties of the objects in the instantiated information model, and their causal relations are expressed by the CPM [11].

A PRM thus constitutes a formal machinery for calculating the probabilities of various architecture instantiations. This allows us to infer the probability that a certain attribute assumes a specific value, given some (possibly incomplete) evidence of the rest of the architecture instantiation. In addition to expressing and inferring uncertainty about attribute values as specified above, PRMs also provide support for specifying uncertainty about the structure of the instantiations.

PRMs further allow specializing classes through inheritance relationships. Classes can be related to each other using the subclass relation $\prec$, and each class X is associated with a finite set of subclasses C[X]. So if $Z, Y \in C[X]$, both Z and Y are subclasses of X. If $Z \prec Y$ then Z is a subclass of Y, and vice versa Y is a superclass of Z. A subclass Z always contains all dependencies and attributes of its superclass Y. PRMs further allow the dependencies and conditional probability distributions of inherited attributes to be specialized in subclasses.

3 Architecture of the Tool

The presented tool is implemented in Java based on a Model-View-Controller architecture [19] [20]. Where the model represents the knowledge and considered data that in this case are PRM respectively instantiated PRM. The view is in charge of the visualization of the model for the tool user and finally the controller links the user to the application making the program react on the users input. Thereby a decoupling of data access, program logic, data presentation, and user interaction can be ensured. This facilitates the extension of the implementation and eliminates potential error sources as the program code is structured and functionality grouped according to its purpose.

The data model for PRMs and instantiated PRMs, is specified in XSD and stored in XML files [21] this is done through an application of the Castor library [22]. As XML is a widespread format created models can be imported into other applications and data does not need to be captured a second time. The user interface is build upon the NetBeans Visual Library [23] with usage of the JApplication framework. This library provides a set of reusable pieces,

called widgets, and can be applied to create visualization. The widgets can be aggregated and related to each other thereby reflecting the creation of models intuitively. These models are drawn on a special canvas that in the NetBeans terminology is called scene. Besides the modeling capabilities the user interface provides the one applying the tool with support functionalities such as filtering, tagging, and export. These well-defined tasks are performed by corresponding tailored services that are implemented following the singleton pattern to ensure data consistency. The architecture described is depicted in Fig. 3.

The tool is separated into two units, one supporting the modeling of the PRM, the other one makes the tool user able to instantiate and analyze this defined structure. These parts have to be used sequentially, reflecting the method that has been described in section 2, starting with the modeling of classes and their attributes as well as the relationships and dependencies between them. Thereby the focus of the analysis is set, as the defined classes reflect the domain of interest. The second component of the EAT allows the instantiation of the PRM. Thereby one or several scenarios of interests are modeled according to constraints defined in the dependency structure for the PRM. Afterwards the analysis is performed. Therefore the instantiated PRM gets translated into a Bayesian network understandable by the Smile library [24]. This library performs the evaluation of the network. Finally the calculated values are written back to the instantiated PRM and visualized for the tool user.

The person applying the tool can than compare the modeled scenarios and their contained classes by consideration of the probabilities that attributes of the classes are in a certain state; thereby the identification of the configuration that qualifies best is made possible.

4 Example Tool Application

This section will illustrate the application of the tool through a case study performed at an electric grid operator in Sweden [25]. The case study focused on assessing data quality in a maintenance management system. In this paper a reduced version of the study is presented, and in the correspondingly reduced theory *data quality* is defined as the *quality of the content* and the *quality of the values* for the data objects used by the system. Quality of content is defined in terms of *relevance*, the degree to which the information objects have a purpose for the users, and *precision*, i.e. that the information objects are detailed enough for the application using them. Turning to *quality of values*, this is defined in terms of *accuracy* and *completeness* of the data objects, or more precisely the attributes contained within a data object. Accuracy is measured as the degree to which the values found in a system correspond to the actual values they represent, whereas completeness is measured as the amount of values stored in the system compared to the domain they represent.

Based on the theory outlined above the PRM for data quality analysis can be constructed. The PRM has four classes, firstly the *Information System* for which the assessment is performed. This system uses a set of *Data Objects* that

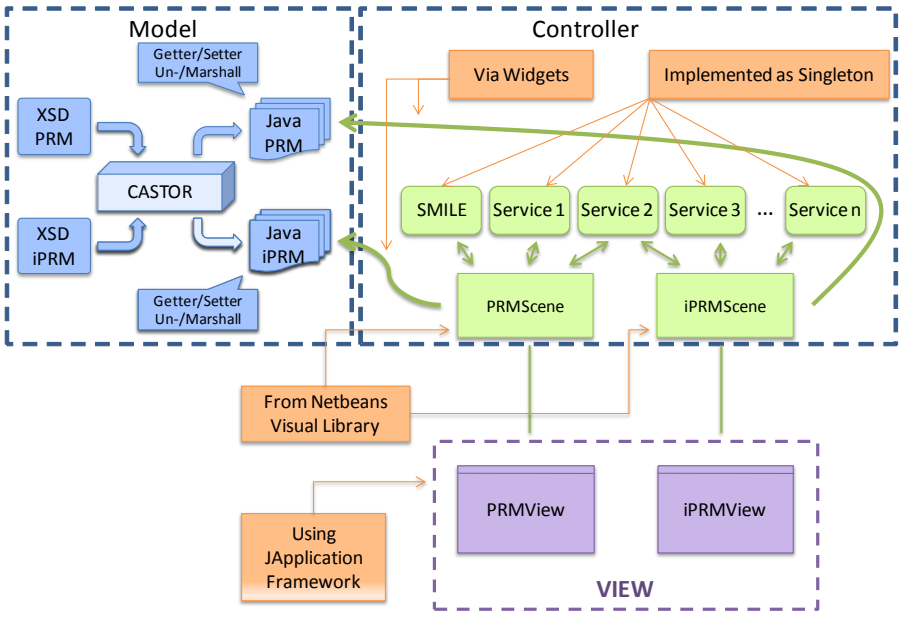

Fig. 3. High level architecture of the tool, illustrating the main components. "iPRM" is shorthand notation for instantiated PRM.

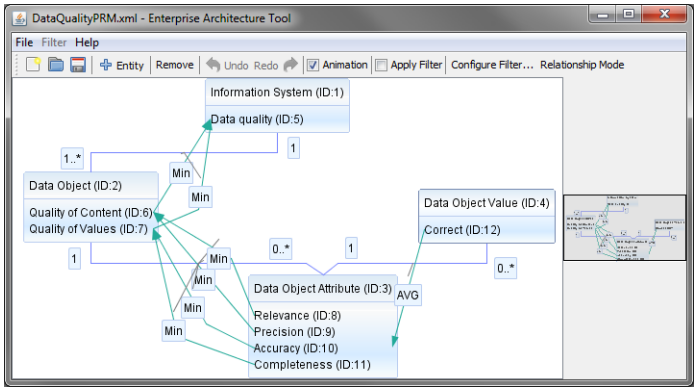

Fig. 4. The PRM for data quality analysis showing classes and attributes relevant for the analysis.

constitute the abstract information model employed by the system. Each object contains one or more *Data Object Attributes* and in the operational system these attributes are instantiated to *Data Object Values*. Turning to the attributes of the PRM, each of the concepts defined in the theory, e.g. relevance and completeness, corresponds to one attribute. This can be seen in Fig. 4, for instance the attribute

Data quality is associated to the class *Information System* and depends on the attributes *quality of values* and *quality of content* of the class *Data Object*. The conditional probabilistic dependencies between attributes were also defined, for details, see [25].

The data in this case study was collected primarily through interviews but for the acurracy of the data objects direct observations were made and compared with values in the system, i.e comparing the data object values with the actual items they represent to assess the accuracy of an attribute. The instantiated PRM is shown in Fig. 5 where the maintenance system, two data objects, three attributes and a set of values of these attributes are modeled. Based on (i) the model and (ii) the values of the descriptive attributes in the PRM (as found during the data collection), the assessment can be performed. The data quality of the maintenance system, measured in percent of complete fulfilment of data quality requirements, was found to be 62 percent.

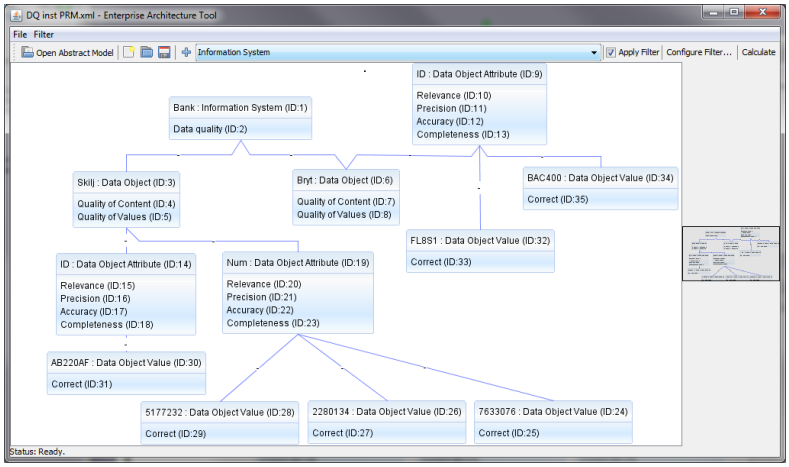

Fig. 5. Instantiated PRM for data quality analysis containing one system, two data objects and three attributes with values attached.

5 Discussion and Future works

This paper presents a tool which supports enterprise architecture analysis with the use of the PRM formalism. While providing a powerful mechanism for the use of discrete variables in an analysis, the PRM formalism in its initial form has a few weaknesses that deserve further studies.

Several system qualities are typically analyzed through the usage of continuous variables e.g. in [26] continuous variables are used for performance analysis. In order to perform those evaluations with support of the presented tool, it is

necessary to discretize all continuous variables. At the moment we are investigating how the PRM formalism can be extended so that it can be used with combinations of continuous and discrete variables, so called hybrid networks [27], as well as a corresponding tool implementation.

Another weakness of the PRM formalism is that it does not provide any means to query the models for structural information such as "given an information system, how many elements does the set of related data objects contain?" The Object Constraint Language (OCL) [28] is a formal language developed to describe constraints on UML models. OCL provides a means to specify such constraints and perform queries on the models in a formal language. OCL in its original form is side effect free, but currently an imperative version of OCL is being added to the tool. Thereby the analysis functionality can be extended to consider the structure of the PRM instantiation more comprehensively.

Besides the two mentioned shortcomings of the used formalism there are some improvements with respect to usability. Regarding the user interface of the tool, we are planning to make the models more intuitive and the information provided easier to understand. Enterprise architecture models are more graspable if they only depict the interesting parts of the model (in a goal-sense). Therefore, the tool should be extended to support views and viewpoints, e.g. as presented in [26]. Additionally we plan the support of iconic visualization of typical enterprise architecture elements, such as applications or data objects, to present the models in an easily understandable way.

Finally we are planning to integrate the support of predefined model components. As models based on the same metamodel are likely to have common parts, the modeling process can be sped up if common building blocks are offered by the metamodel provider and used by the person that creates a certain model.

6 Conclusion

In this paper a tool and method for analysis of enterprise architecture scenarios was presented. To fulfill this purpose the tool consist of two separate parts, one for defining analysis theory and one for enterprise architecture modeling, and makes use of the PRM formalism for specifying theory. Applying this formalism allows for the consideration of uncertainty, an aspect that so far is uncommon in the field of enterprise architecture analysis. The paper describes the PRM formalism as well as the underlying architecture of the tool briefly.

In the paper an example of data quality assessment was outlined, but the tool supports the analysis of various quality attributes, such as maintainability, information security, and interoperability. The tool supports information system decision making as it allows the comparison of several scenarios with regard to a system quality. Thereby the "as-is" as well as several "to-be" architecture of an enterprise can be compared quantitatively in order to find the one that best satisfies decision maker requirements.

References

1. Ross, J.W., Weill, P., Robertson, D.: Enterprise Architecture As Strategy: Creating a Foundation for Business Execution. Harvard Business School Press (August 2006)
2. Winter, R., Fischer, R.: Essential layers, artifacts, and dependencies of enterprise architecture. Journal of Enterprise Architecture **Volume 3, Number 2** (2007) 7–18
3. The Open Group: TOGAF 2007 edition. Van Haren Publishing, Zaltbommel, Netherlands (2008)
4. Zachman, J.A.: A framework for information systems architecture. IBM Syst. J. **26**(3) (1987) 276–292
5. Department of Defense Architecture Framework Working Group: DoD Architecture Framework, version 1.5. Technical report, Department of Defense, USA (2007)
6. NAF: NATO C3 Technical Architecture. (2005)
7. Johnson, P., Ekstedt, M.: Enterprise Architecture – Models and Analyses for Information Systems Decision Making. Studentlitteratur, Sweden (2007)
8. Troux Technologies: Metis. `http://www.troux.com/products/` (March 2010)
9. IBM: System Architect. `http://www-01.ibm.com/software/awdtools/systemarchitect/productline/` (March 2010)
10. Scheer, A.: Business process engineering: Reference models for industrial enterprises. Springer-Verlag New York, Inc. Secaucus, NJ, USA (1994)
11. Getoor, L., Friedman, N., Koller, D., Pfeffer, A., Taskar, B.: Probabilistic relational models. In Getoor, L., Taskar, B., eds.: An Introduction to Statistical Relational Learning. MIT Press (2007)
12. Kurpjuweit, S., Winter, R.: Viewpoint-based meta model engineering. In: Enterprise Modelling and Information Systems Architectures (EMISA 2007) (2007)
13. Franke, U., Johnson, P., König, J., Marcks von Würtemberg, L.: Availability of enterprise IT systems – an expert-based bayesian model. In: Proc. Fourth International Workshop on Software Quality and Maintainability (WSQM 2010), Madrid. (March 2010)
14. Ullberg, J., Lagerström, R., Johnson, P.: A framework for service interoperability analysis using enterprise architecture models. In: IEEE International Conference on Services Computing. (July 2008)
15. Sommestad, T., Ekstedt, M., Johnson, P.: A probabilistic relational model for security risk analysis. Computers & Security (February 2010) Accepted.
16. Lagerström, R., Saat, J., Franke, U., Aier, S., Ekstedt, M.: Enterprise meta modeling methods – combining a stakeholder-oriented and a causality-based approach. In: Enterprise, Business-Process and Information Systems Modeling, Lecture Notes in Business Information Processing. Volume 29., Springer Berlin Heidelberg (June 2009) 381–393 ISSN 1865-1348.
17. Jensen, F.V.: Bayesian Networks and Decision Graphs. Springer-Verlag New York, Inc., Secaucus, NJ, USA (2001)
18. Friedman, N., Getoor, L., Koller, D., Pfeffer, A.: Learning probabilistic relational models. In: Proc. of the 16th International Joint Conference on Artificial Intelligence, Morgan Kaufman (1999) 1300–1309
19. Reenskaug, T.: Models-views-controllers. Technical note, Xerox PARC (1979)
20. Sun Microsystems: Design Pattern: Model-View-Controller. `http://java.sun.com/blueprints/patterns/MVC.html` (2002)
21. Bray, T., Paoli, J., Sperberg-McQueen, C., Maler, E., Yergeau, F.: Extensible markup language (XML) 1.0. W3C recommendation **6** (2000)

22. ExoLab Group: The Castor Project. http://www.castor.org/ (March 2010)
23. NetBeans: NetBeans Visual Library. http://graph.netbeans.org (March 2010)
24. Decision Systems Laboratory of the University of Pittsburgh: SMILE. http://genie.sis.pitt.edu/ (March 2010)
25. Magnusson, S., Udin, S.: Assessing data quality on the maintenance sytem at svenska kraftnät. Technical report, Industrial Information and Control Systems, KTH (2009)
26. Lankhorst, M.: Enterprise architecture at work : modelling, communication, and analysis. Springer (2005)
27. Lauritzen, S.: Propagation of probabilities, means, and variances in mixed graphical association models. Journal of the American Statistical Association **87**(420) (1992) 1098–1108
28. Object Management Group: Object Constraint Language specification, version 2.0 formal/06-05-01. Technical report (2006)